INTEGRATED DEVELOPMENT ENVIRONMENT USING CLOUD COMPUTING (IDECC)

A major project report submitted in partial fulfilment of the requirement for the award of degree of

Bachelor of Technology

In

Computer Science & Engineering/Information Technology

Submitted by

SANSKAR RAI (201256)

RISHABH KESARWANI (201532)

Under the guidance & supervision of

DR. NISHANT SHARMA

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology, Waknaghat, Solan – 173234 (India)

CERTIFICATE

This is to certify that the work which is being presented in the project report titled **"Integrated Development Environment using Cloud Computing (IDECC)"** in partial fulfilment of requirements for the award of the degree of B.Tech in Computer Science & Engineering / Information Technology and submitted to the Department of Computer Science & Engineering And Information Technology, Jaypee University of Information Technology, Solan is an authentic record of work carried out by Sanskar Rai (201256) and Rishabh Kesarwani (201532) during the period from August 2023 to May 2024 under the supervision of Dr. Nishant Sharma (Assistant Professor(SG), Department of Computer Science & Engineering And Information Technology)

Sanskar Rai
(201256)

Rishabh Kesarwani
(201532)

The above statement made is correct to the best of my knowledge

Dr. Nishant Sharma
Assistant Professor (SG)
Computer Science & Engineering and Information Technology
Jaypee University of Information Technology, Solan

DECLARATION

We hereby declare that the work presented in this report entitled **“Integrated Development Environment using Cloud Computing (IDECC)”** in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering / Information Technology** submitted in the Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat is an authentic record of our own work carried out over a period from August 2023 to May 2024 under the supervision of Dr. Nishant Sharma (Assistant Professor(SG), Department of Computer Science & Engineering And Information Technology)

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Sanskar Rai	**Rishabh Kesarwani**
(201256)	**(201532)**

This is to certify that the above statement made by the candidates is true to the best of my knowledge.

Dr. Nishant Sharma
Assistant Professor (SG)
Computer Science & Engineering and Information Technology
Jaypee University of Information Technology, Solan

ACKNOWLEDGEMENT

Firstly, we express our heartiest thanks and gratefulness to almighty God for His Divine Blessing makes us possible to complete the project work successfully. We are really grateful and wish our profound indebtedness to our Supervisor **Dr. Nishant Sharma, Assistant Professor (SG)**, Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Solan. Deep Knowledge & keen interest of my supervisor in the field of **"Cloud Computing"** to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stages have made it possible to complete this project.

We would like to express our heartiest gratitude to **Dr. Nishant Sharma,** Department of Computer Science & Engineering and Information Technology, for his kind help to finish our project. We would also like to generously welcome each one of those individuals who have helped us straightforwardly or in a roundabout way in making this project a win. In this unique situation, we might want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated our undertaking.

Finally, we must acknowledge with due respect the constant support and patience of our parents.

Sanskar Rai	**Rishabh Kesarwani**
(201256)	**(201532)**

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

This project report is about the development of IDECC, **Integrated Development Environment using Cloud Computing (IDECC)**. Every developer's nightmare is creating a development environment for every small or big project which includes all the dependencies used in the project, maintaining these dependencies and then installing them again if some error occurs, and making the environment again. IDECC aims to remove this hassle for every developer by introducing Browser-based Integrated Development Environment, by removing the need of creating a development environment on PC, IDECC provides us with a list of programmable languages and Frameworks which you can launch in IDE just by a single click, in your browser. No need to install dependencies on your PC and maintain a development environment, IDECC aims to increase the learning, by removing unwanted hassles, mostly daunting for new coders. IDECC will improve collaboration between team members working on any project and increase each other's knowledge. The frontend of the system will be built using React and the backend using Django. The Development Environments will be containerised in Cloud Storage, which when needed will go and run up a virtual machine with the same configurations, and will be ready to use by the developer. Using Azure, Scalability of the Virtual Machines will also be easy. IDECC aims to revolutionise the way development is done in the Software Development community, opening doors to new boundaries.

CHAPTER 1: INTRODUCTION

1.1.INTRODUCTION

In the dynamic world of software development, innovation is all about how smoothly one can build their code, collaboratively and without the complications from intricate setup processes. Developers sometimes find themselves caught up in the complications of establishing development environments, varying from one pc to another, from one type of OS to another, a procedure that not only wastes a lot of time but also creates a barrier to fluid coding. Recognizing this issue our main project embarks on a transformative journey to reinvent the fundamental nature of software development.

At its core, our major project strives to address difficulties that poses a difficulty for every developer during the earliest stages of project - the work of setting up development environments. This issue is not only time consuming but also often leads to compatibility concerns, which limits the exchange of code and also the natural flow of project activity. Our objective is to create a cloud based Integrated Development Environment (IDE) platform that not only frees developers from the hassles of setting up development environment but also helps them with collaborative coding environment.

1.1.1 UNDERSTANDING THE CHALLENGES WE CONFRONT

The foundation of this project comes in the fact that developers irrespective of their experience, become tangled with the difficulties of customizing development environments. This procedure is loaded with tiresome manual setups and compatibility concerns, which works as a speed breaker in the lifecycle of software development. The time spent on setting up development environment immediately affects our time in the project. Moreover, the disparities in development environment between the team members can lead to sluggish project efforts.

Our initiative seeks to not only remove this difficulty from the life of individual developers but also from the team and collaborative members as a whole. Our project acknowledges this issue and wants to provide the answer – a cloud-based IDE platform that not only removes the challenges of setting up a development environment but also assures a harmonized environment allowing the developers to code easily between team members.

1.1.2 EMPOWERING DEVELOPERS: THE PROJECT'S OBJECTIVES AND GOALS

By developing a cloud-based IDE platform, we seek to simplify environment setup. The platform attempts to deliver pre-configured environments optimized for a number of programming languages and frameworks. This seeks to ease the load of laborious setup, enabling the developers to leap straight into coding with minimal friction. The platform also strives to standardise development environments. The cloud-based IDE will strive to provide a common development environment across the team, reducing compatibility difficulties and creating a uniform coding ecosystem. All this will lead to an increase in productivity and collaboration: By addressing the limitations given by the environment arrangement, our project aims to boost productivity and aiding develop a culture of invention and collaboration withing the team.

1.1.3 NAVIGATING THE TECHNOLOGICAL FRONTIER: THE TOOLS AND TECHNOLOGIES

The implementation of our concept demands a hand-picked selection of cutting-edge technologies. Our cloud-based IDE platform harnesses the power of cloud-computing, offering developers with on-demand access to scalable and dynamic environment. Containers play the key function in our project, encapsulating the entire development and maintaining consistency throughout different phases of the software development lifecycle. Continuous Integration/Continuous Deployment (CI/CD) is also being evaluated to employ to provide seamless testing and deployment procedures to boost the entire development workflow.

Moreover, the user interface is made as friendly as possible, making it easy for the user to browse through the website. All the famous languages and frameworks will be integrated through the website for the user to use. User will have the option to visit their prior projects, as well as make new ones.

The pre-loaded environments will be containerised using Docker. The website containing the Docker containers are expected to be placed on Azure Cloud Service supplied by Microsoft to allow latency free communication between the user and website experience.

Our project intends to emerge as an easy-to-use experience for every developer. By introducing a cloud-based IDE platform, we aim to envision a future where developers are not bounded by the shackles of manual configuration of development environment, a platform providing seamless collaboration, and a platform where developers are free to test to try on anything without having to worry about anything. As we set sail on this adventure, the objective is clear – a transformed software development landscapes that thrives on efficiency, collaboration and the unrestrained creativity of developers.

1.2.PROBLEM STATEMENT

Every programming language and/or framework require very intricacies sets of dependencies, libraries for the language to be developed properly and that too in a maintained development environment which comes out as a problem not only for rookie programmers but also for seasoned ones and is quite a hassle to set it up each and every time a new project is to be made by the developer, even if the project is only a Hello World. Setting up the development environment in IDEs has been a long-time headache in the software development community.

1.3. OBJECTIVES

The objectives of this project include features that every developer whether it be a new programmer or a seasoned developer, everyone will find it appealing to work on.

1.3.1 CREATING AN IDEAL PLATFORM

Our main objective stands to develop an ideal platform for every user so that even a newbie will find it easy and appealing to work on the application. These may include generating a user-friendly website, easy and appealing for everyone. Keeping the process smooth and fast for the user, because no one likes too much buffering time.

All the while providing every feature that is important and necessary for the application to work.

1.3.2 COORDINATING JOINT VENTURE ACTIVITIES

No developer works alone, they always work in a team. Keeping this in mind we want to provide our user(s) with proper collaborative features so that any team can work on their project easily while fully collaborating with each other.

1.4. SIGNIFICANCE AND MOTIVATION OF THE PROJECT WORK

1.4.1 EXPLORING THE IMPORTANCE

The primary objective of this program is to radically reinvent the way developers approach their profession by lowering the harsh hurdles connected with setting up development environments. This project is designed to alter the industry by providing an all-encompassing platform that stimulates innovation, boosts efficiency, and streamlines operations.

1.4.2 GIVING DEVELOPERS MORE AUTHORITY

The most important thing is to allow developers, whether novice and experienced, to concentrate their attention on what actually matters: code. By making the difficult process of developing environments easier, this project frees developers from the shackles of setup concerns. This helps developers to unleash their creative potential and push the boundaries of software engineering.

1.4.3 INNOVATIVE COLLABORATION TECHNIQUES

Furthermore, the effort has the potential to radically transform the collaborative operations of development teams. Creating uniform development environments for team members increases communication, lowers compatibility difficulties, and speeds up project timeframes. This will develop a climate receptive to new ideas and amiable teamwork, resulting in higher-quality software products.

1.4.4 THE INSPIRATION FOR THE PROJECT

This project was prompted by a desire to make complex things easier and procedures more efficient. The idea derives from a common desire to alleviate developers of the stress of complex setup processes so that they may focus on generating code. It's an invitation to cultivate an environment in which innovation thrives, creativity soars, and cooperation knows no bounds.

1.5. ORGANIZATION OF PROJECT REPORT

1.5.1 CHAPTER 1: INTRODUCTION

The opening chapter of our project report, contains the overview of the whole project. The Introduction chapter includes the complete overview of the project, the opening chapter also contains the problem statement, it also contains the main usage of the project, its advantages, use cases. The motivation for the team to work on this project is also mentioned in the Introduction chapter highlighting its significance as well.

1.5.2 CHAPTER 2: LITERATURE RESEARCH

Every project work is started by doing proper research firsthand to map out the necessary technology that is going to be used in the project, to scope out the necessary precautions, the help that can be found by someone who has already encountered it and all of the things. This chapter includes all the relevant literature papers that our team has studied and deemed it usable for our project, it includes all the different technological stacks that are to be used, some papers on similar project, understanding the deployment and different things necessary for the completion and usage of this project.

1.5.3 CHAPTER 3: SYSTEM DEVELOPMENT

The required technological stacks, deployment servers, programming languages etc are covered in this chapter. This chapter covers the major infrastructure details of the system, including the hardware, software, and networking components. It also provides a pseudo code of the system's main functions and a diagram of the database structure. The necessary images needed for understanding the flow of data, networking etc. are displayed in this chapter of the report. It gives us the in depth understanding of the whole technological need of the project.

1.5.4 CHAPTER 4: TESTING

This chapter covers the testing details of the system, including the methodology used and the results of the tests.

1.5.5 CHAPTER 5: RESULTS AND EVALUATION

The results of the project are shown in this section of the report. It contains all the findings and the progress of the report. In this section we have also compared our findings and work with already existing applications to give an outline of what else needs to be done in the project.

1.5.6 CHAPTER 6: CONCLUSIONS AND FUTURE SCOPE

In this part of the report the conclusion that can be derived from the work are discussed, all the findings and results are discussed here. The future scope of this project discusses what else is left to be done and the improvements that can be done in the already created project.

CHAPTER 2: LITERATURE SURVEY

2.1 OVERVIEW OF RELEVANT LITERATURE

A.

Title	Microsoft azure: Cloud platform for application service deployment [1]
Author	VP Desai, KS Oza, PP Shinde, Dr. PG Naik
Year	2021
Summary	This research paper presents an in-depth assessment of Microsoft Azure's web application deployment and administrative features. The paper gives a thorough assessment of a number of Azure services, stressing their appropriateness for various application scenarios. Azure App Service, Azure Cloud Services, and Azure Virtual Machines are among the services provided. Microsoft Azure Application Framework Azure App Service is a fully managed Platform as a Service (PaaS) solution that simplifies web application setup, scalability, and management. The article describes the core features of Azure App Service, as well as how many operating systems, programming languages, and frameworks it supports. It also stresses how scalable and user-friendly Azure App Service is, making it a fantastic solution for both small and large-scale apps. Azure Web Services Azure Cloud Services, a more typical IaaS (Infrastructure as a Service) offering, provides more control over the underlying infrastructure. The paper not only describes the complicated capabilities of Azure Cloud Services, such as load balancing and auto scaling, but also demonstrates how to implement them. It emphasizes how well-suited Azure Cloud Services are for applications that demand more exact infrastructure control.

	Azure Virtual Machines Azure Virtual Machines offer a completely configurable IaaS solution for applications that require utmost control and flexibility. The document describes the process of constructing and administering virtual machines, as well as the different types of Azure Virtual Machines that are available, such as Windows and Linux instances. It highlights how Azure Virtual Machines may be expanded and adapted to meet the demands of high-performance applications. Comparative Analysis The report compares and contrasts Azure App Service, Azure Cloud Services, and Azure Virtual Machines, highlighting the merits and drawbacks of each. It supports readers in identifying the appropriate deployment choice for their unique application demands and preferences. This research paper is a wonderful resource for learning and using Microsoft Azure's web application deployment and administration features. This research report will be very beneficial as we develop our services on Azure.

Table: 2.1 Review of Azure and its services

B.

Title	Automated Dockerisation of Python-based Web Apps [2]
Author	SK Shreekar
Year	2020
Summary	The author of the paper discusses the issues of containerizing Python applications and offers an automatic Dockerisation method created exclusively for Python-based projects. The aforementioned Python framework automates the production of Docker images, the execution of Docker containers, and the creation of Docker files,

	thereby speeding up the Dockerisation process. Depending on the use case, this framework may greatly enhance the productivity of Dockerizing Python-based apps while lowering the possibility of errors and possibly eliminating the need for manual intervention entirely. The automation is performed through two steps: generation of the Docker file, construction of the Docker image, and container execution. Following the generation of the Docker file, the framework automatically builds the Docker image. Reduced mistakes, reproducibility, consistency, and efficiency are among the advantages of the automated Docker architecture. The suggested framework provides a useful tool for developers in DevOps teams, boosting the productivity and consistency of the Dockerisation process. The authors' work underlines the value of automation in accelerating the Dockerisation process and proving Docker's potential.

Table: 2.2 Automated Dockerisation of Python Application

C.

Title	Building Modern Clouds: Using Docker, Kubernetes & Google Cloud Platform [3]
Author	Jay Shah, Dushyant Dubaria
Year	2019
Summary	The study paper looks into the intricacies of modern cloud infrastructure and its implementation with Docker, Kubernetes, and the Google Cloud Platform. The essay gives an in-depth discussion of these technologies and their roles in the development of scalable, robust, and cost-effective cloud systems.

	The paper's author initially introduces us to the concept of cloud computing, highlighting its advantages over traditional on-premises architecture. He then discusses Docker, a containerization platform that allows applications and their dependencies to be bundled into self-contained, lightweight containers. Docker containers are useful for cloud deployments because they enable isolation, mobility, and consistency. Later, the author explores Kubernetes, a container orchestration system that simplifies containerized application management. Kubernetes facilitates container deployment, scaling, and management across a cluster of servers, providing efficient resource consumption and high availability. The paper then focuses on the Google Cloud Platform (GCP), a full cloud computing suite that comprises compute, storage, networking, and databases. GCP, with its scalability, dependability, and security features, provides a robust framework for constructing modern clouds. The article emphasizes the scalability, resilience, cost-effectiveness, and agility of using Docker, Kubernetes, and GCP. Although we do not anticipate adopting GCP, this study paper offers our team with a clear understanding of how Docker and Kubernetes interact with one another and may benefit us in our future initiatives. The article's author explores the issues of containerizing Python programs and presents an automatic Dockerisation approach built particularly for Python-based projects. The previously mentioned Python framework automates the development of Docker images, the execution of Docker containers, and the creation of Docker files, hence expediting the Dockerisation process. Depending on the use case, this framework will considerably boost the productivity of Dockerizing Python-based apps while decreasing the risk of errors and, in some situations, totally removing the need for manual intervention.

	The automation is performed in two steps: the development of the Docker file, the construction of the Docker image, and the execution of the container. Following the generation of the Docker file, the framework builds the Docker image automatically. The automated Docker architecture has various advantages, including decreased errors, reproducibility, consistency, and efficiency. The recommended framework is a great tool for developers working in DevOps teams, as it enhances the productivity and consistency of the Dockerisation process. The authors' work underscores the relevance of automation in expediting Dockerisation and proving Docker's potential.

Table: 2.3 Building Cloud using Docker, Kubernetes and GCP

D.

Title	Software Containerisation with Docker [4]
Author	RC Diaz Alonso
Year	2017
Summary	Since the release of Docker, containerization has emerged as a disruptive technology, allowing software applications and their dependencies to be bundled into lightweight, isolated containers. The paper by Diaz Alonso digs into the complexity of software containerization with Docker, providing a complete review of its ideas, benefits, and implementation. The notion of containers and its advantages over traditional virtualization techniques are described at the beginning of the paper. It emphasizes the lightweight nature of containers, their ability to share the host operating system kernel, and how their isolation features aid in the avoidance of application conflicts.

	This document delves into deeper depth regarding the development, operation, and management of Docker containers. It describes how to produce Docker images from Docker files and then use those images to launch containers. It also introduces the concept of Docker orchestration, which allows you to manage many containers as a single service. By demystifying the concepts of containers, Docker images, and Docker files, the article walks readers through the process of constructing, running, and maintaining Docker containers. The essay also illustrates the advantages of Docker for software development and deployment, making it a great resource for anyone researching containerization technologies.

Table: 2.4 Software Containerisation in Docker

E.

Title	Microsoft Azure: Planning, Deploying, and Managing Your Data Centre in the Cloud [5]
Author	Julian Soh, Marshall Copeland, Anthony Puca, Mike Manning, and David Gollob
Year	2015
Summary	This book goes into the complex world of Microsoft Azure and leads readers through the process of creating, implementing, and maintaining cloud-based data centres. The book, aimed for both corporate leaders and IT specialists, gives a full assessment of the benefits, challenges, and variables to consider when picking Azure for data centre requirements. The planning phase comprises a complete analysis of the company's present IT burden, applications, infrastructure, and business goals. The authors underline

	the significance of aligning cloud adoption with strategic goals, enabling a smooth transition, and leveraging on Azure's benefits. The book then gets into the intricacies of installing Azure, providing step-by-step instructions for creating and configuring virtual machines, storage accounts, networks, and security groups, among other Azure services. The writers handle a wide range of IT infrastructures by discussing cloud-based and on-premises deployment scenarios. Following the installation of the Azure infrastructure, the book concentrates on running the cloud data centre efficiently. This includes monitoring how resources are used, maximizing performance, and ensuring security and compliance regulations are followed. The authors provide extensive descriptions of Azure's administrative tools and services so that readers may keep control over their cloud-based data centre. Because of its detailed methodology, insightful ideas, and real-world examples, the book is an invaluable resource for anyone embarking on an Azure cloud adventure.

Table: 2.5 Literature Review on Azure

F.

Title	Docker: Lightweight Linux Containers for Consistent Development and Deployment [6]
Author	Dirk Merkel
Year	2014
Summary	Environments for development and deployment have always required to be dependable and efficient. Traditional virtualization technologies, even when they function effectively, could have a substantial overhead that restricts

	mobility and agility. The author of this research paper developed Docker, a unique containerization strategy that leverages lightweight Linux containers, to establish stable and portable development environments. Docker containers are self-contained software packages that encapsulate a program's code, dependencies, and runtime environment. This functionality allows the application to execute smoothly across a wide range of hardware platforms and operating systems. Docker containers share the host kernel, which greatly decreases overhead and resource needs, allowing for faster startup times and greater resource utilization than traditional virtual machines. The four most frequently mentioned Docker benefits are efficiency, isolation, portability, and consistency. Docker containers reduce mistakes and irregularities by keeping constant conditions throughout the development, testing, and production phases. Docker containerized apps may be quickly deployed on a wide range of platforms, including cloud servers and laptop computers, with no compatibility difficulties. Docker containers separate various applications from one another, preventing conflicts and maintaining system stability. Docker containers separate various applications from one another, preventing conflicts and maintaining system stability. The authors' Docker work provides a viable answer to the issues associated with dependable and effective software development and deployment, and it marks a substantial advancement in containerization technology. Docker is a popular solution for modern software development teams due to its lightweight architecture, portability, and isolation qualities, which enable them to create and launch apps more quickly and reliably.

Table: 2.6 Review on Docker

G.

Title	Browser Based IDE to Code in Cloud [7]
Author	Lakshmi M Gadhikar, Lavanya Mohan, Megha Chaudhari, Pratik Sawant, Yogesh Bhusura
Year	2013
Summary	As cloud computing has risen in popularity, the necessity for cloud-based development tools has skyrocketed. Browser-based Integrated Development Environments (IDEs) have developed as a viable alternative to standard desktop IDEs for cloud coding. The ability to work with the code, the project, and the team from any location and on any internet-connected device is the most crucial aspect of an IDE. This permits distant cooperation and promotes developer accessibility. Thanks to browser-based integrated development environments (IDEs), developers may communicate in real time with multiple users on the same codebase. This real-time collaboration enhances team productivity and fosters information exchange. Browser-based IDEs, which are usually cloud-hosted and available as subscription services, eliminate the upfront expenditures associated with traditional desktop IDEs. Cloud IDEs are appealing to both consumers and enterprises due to their low cost. Developing teams and projects can grow, and cloud IDEs can stay up with them. Because of its scalability, cloud IDEs may enable large-scale development initiatives. Numerous studies have been undertaken to evaluate the advantages of browser-based integrated development environments (IDEs) over traditional

	IDEs, and all have determined that modern IDEs outperform classic desktop IDEs. There are three sorts of IDEs: general purpose, which supports a wide range of features and programming languages, professional, and specialized. Language-specific integrated development environments (IDEs) are ones that are developed particularly for and only support a certain language type. Web apps are designed or created with the use of web-based IDEs. Despite the issues related with browser-based integrated development environments (IDEs), such as accessibility and collaboration, the IDE's speed should not be compromised. This means that the IDE can work flawlessly on cloud containers rather than PCs, even if the hardware specs are inadequate. Given that the IDE may involve sensitive data, they must ensure sufficient security. The IDE must support multiple programming languages. The merits and downsides of browser-based IDEs are eventually explored in the paper's conclusion. Browser-based IDEs have the potential to drastically transform the software development industry if these challenges are handled.

Table: 2.7 Advantages of Browser Based IDE

H.

Title	Real-time collaborative coding in a web IDE [8]
Author	M. Goldman, G. Little, R.C. Miller
Year	2011
Summary	The capacity of developers to collaborate efficiently is crucial to the success of a project in software development. Traditional desktop-based IDEs, despite being strong tools, frequently fail to enable smooth collaboration, especially when engineers are geographically scattered. Collabode, a web-based

	integrated development environment (IDE) created expressly to facilitate real-time collaborative coding, was proposed as a solution to this problem by Goldman, Little, and Miller (2011). Collabode's web-based architecture has various advantages over traditional IDEs. For starters, it eliminates the requirement for software installation, allowing developers to utilize the IDE from any internet-connected device. Because of this improved accessibility, developers can collaborate remotely no matter where they are physically located. Second, thanks to Collabode's real-time collaboration features, developers may edit in tandem and interact simply. By allowing developers to remark on one other's work in real time, real-time code change watching encourages a sense of shared codebase ownership. One of Collabode's primary features is its error-mediated integration technique. By identifying and reporting any defects when developers make changes to the code, Collabode ensures that only error-free updates are merged into the shared repository. This error-aware method preserves the integrity of the codebase and avoids conflicts. Collabode is an important contribution to the field of web-based integrated development environments (IDEs), paving the way for more efficient and collaborative software development processes. Collabode is an essential tool for modern development teams since it stimulates knowledge sharing, improves code quality, and boosts developer productivity by leveraging web technologies and real-time collaboration features.

Table: 2.8 Study on Real Time Collaborative Browser IDE

I.

Title	Automatic Software Deployment in the Azure Cloud [9]

Author	Jacek Cala, Paul Watson
Year	2010
Summary	The study paper presents a mechanism for automatic software deployment on the Azure cloud. The framework defines the infrastructure and software configurations for the application using Azure Resource Manager (ARM) templates. The framework then handles the application's deployment to the Azure cloud automatically via Azure Pipelines. When compared to traditional manual deployment techniques, the framework offers various advantages. It is more efficient since apps may be deployed in minutes rather than hours or days. It is more dependable since it is less prone to human error. It is also more scalable because it can be used to deliver applications to a variety of situations. The framework has been used to successfully deploy several apps to the Azure cloud. According to the designers, the framework dramatically decreased the time and effort required to launch apps. They also found that the framework increased the dependability and quality of the deployments. The methodology provided in the study article proposes a possible new way for handling software deployment on the Azure Cloud. The framework has been used to successfully launch multiple apps on the Azure Cloud and is effective, dependable, and scalable.

Table: 2.9 Automatic Software Deployment of Azure

J.

Title	Adinda: a knowledgeable, browser-based IDE [10]
Authors	Arie Van Deursen, Ali Mesbah, Bas Cornelissen, Andy Zaidman, Martin Pinzger, Anja Guzzi
Year	2010

Summary	Integrated development environments, or IDEs, are increasingly core software development tools. To aid programmer development, they feature a variety of functions included as editing, formatting, debugging, deployment, extensions, and more. Classic IDEs, on the other hand, offer severely limited accessibility and collaboration functionality. The authors of this study established the Adinda idea to address this issue for developers; it has a web-based architecture and gives a number of advantages over standard desktop-based IDEs. To begin with, because it is web-based, there is no need to install it on desktop PCs. Users have access to their projects and works from anywhere, allowing programmers to collaborate, work remotely, and share their work with others. Second, Adinda's browser-based methodology promotes real-time communication among developers. Adinda makes an attempt to use web technologies such as WebSocket and Ajax, which enable parallel debugging, code editing, and communication among numerous developers working on the same piece of code. This in-the-moment collaboration enhances team productivity and knowledge. Adinda's creative approach to IDE design garnered her considerable praise from the software development community. For example, a study conducted by the Adinda founders and research paper writers demonstrated that Adinda dramatically reduced development time when compared to a regular desktop IDE. An additional analysis suggested that Adinda goes to significant pains to find potential problems that could benefit programmers in their work. All of Adinda's talents and research suggest that with appropriate web technology and knowledge-aware approach application, a very strong IDE may be built. Adinda supports software development teams in more effectively

	interacting, being more productive, and delivering better code. Adinda's capabilities are projected to extend more as web technologies evolve, enabling for more efficient and collaborative software development processes.

Table: 2.10 Literature on a Browser IDE Adinda

K.

Title	An interactive environment for real-time software development [11]
Authors	Persson, P. and Hedin, G
Year	2002
Summary	As a result, this study delves at how organizations throughout the world manage performance assessments, asking us to get our hands dirty. The literature review acts as a guide for us, indicating why this study is the star of the academic performance assessment show. These systems for judging success, you see, are the actual MVPs of the academic world. The sculpture is designed to resemble a detective. Wolf's Performance Appraisal Diagnostic Model from 2003 is currently their go-to tool. According to scholarly study, higher education institutions that seek to improve their academic ranking must grasp the complexity of measuring academic performance. Furthermore, how do they obtain the inside information? They built an international alliance with 19 universities from five continents, allowing us a front-row seat to the world's opinion of performance ranking. The major consequence is that the comparison study is essentially an afterthought in the argument over academic performance rankings. It investigates the factors that influence institutions' evaluation procedures using Wolf's diagnostic model as a framework. The findings are not just for show; they are crucial for refining current models and generating unique performance

	grading schemes tailored to individual educational institutions around the world. The main objective of a real-time system is to return the output by the deadline and at the correct moment. A real-time setup that misses deadlines must be assessed. When real-time systems attempt to track the amount of time necessary to accomplish operations in real-time, users find it engaging. The objective of the researchers is to match these forms of user- and real-time environment-presented expectations with the software used in this study paper.

Table: 2.11 Literature Review on Browser IDEs

2.2 KEY GAPS IN THE LITERATURE REVIEW

1 Browser-based IDEs and their use in the software industry.

Browser-based integrated development environments (IDEs) have altered the software development sector by giving a smooth and accessible programming environment that crosses geographical and device boundaries. With these cloud-hosted IDEs, developers can now code, debug, and launch programs straight from their web browsers, removing the need for numerous setups and local device installations.

Browser-based integrated development environments (IDEs) give developers a highly resource-rich and scalable workplace by leveraging cloud infrastructure. Because local hardware limits are no longer an issue, developers may work on sophisticated apps without worrying about system constraints. Furthermore, cloud-based IDEs boost teamwork and real-time code editing between geographically split teams by giving increased collaboration options.

They help to overcome the constraints of standard IDEs. Developers may access their work environment from any device with an internet connection, whether it's a desktop, laptop, tablet, or smartphone. Developers can work whenever and wherever they desire because of this independence, without sacrificing their uniqueness or productivity.

Browser-based integrated development environments (IDEs) improve the development workflow by adding necessary tools and functionalities directly into the IDE environment. Developers gain real-time access to code completion, syntax highlighting, debugging tools, and code integration with well-known version control systems such as Git. This excellent technique boosts code quality while saving development time.

Browser-based integrated development environments (IDEs) have risen in popularity in the software industry, particularly among front-end and web engineers. Because of their ease of use, accessibility, and cloud-based architecture, they are perfect for remote teams and collaborative projects. As cloud technologies expand, the use of browser-based integrated development environments (IDEs) is likely to grow, affecting the process of developing and distributing software.

The introduction of browser-based IDEs, which have spawned new paradigms and promoted a more collaborative and agile atmosphere, has profoundly affected software development techniques. The ability of developers to iterate quickly, receive real-time feedback, and communicate with team members in a seamless manner accelerates and improves software development cycles.

Browser-based integrated development environments (IDEs) represent a significant shift in the software development environment by presenting developers with a flexible and user-friendly platform that overcomes traditional barriers. Because of their cloud-based design, device agnosticism, and rapid development workflow, they are a crucial tool for current software development teams. Browser-based IDEs will shape future software development, and their effect on the software industry will only expand as they evolve.

2 Using Docker for containerization.

Docker provides a lightweight and portable approach of packing applications, which has fundamentally revolutionized the software development and deployment landscape. Docker containers encapsulate an application's code, runtime, system tools, system libraries, and settings to create a self-contained environment that can work on any system that has Docker installed. This enables continual application execution across all contexts and avoids the need for elaborate installation procedures.

One key feature of Docker is its portability, which allows developers to seamlessly transfer their applications from development to production settings without worrying about compatibility issues. Furthermore, Docker containers are segregated from one another, eliminating conflicts and guaranteeing that each application runs independently of other containers that are currently in use.

Docker containers are also efficient and lightweight since they employ shared host resources rather than constructing separate virtual machines. In addition to decreasing the stress on the host system, this resource-sharing method reduces container startup time and memory footprint.

Containers are frequently produced and executed regardless of infrastructure or environment thanks to Docker's repeatability features. Reproducibility is crucial for accelerating testing and deployment procedures as well as keeping consistent application behaviour.

3 Setting up environments for each different language in Docker.

One of Docker's best features is that it is a relatively portable technology. This means that migrating your Docker containers from one computer to another will be a breeze. This might be pretty handy if you need to deploy your application to a production environment or work on your code from several machines.

Docker containers demand less memory and may start up rapidly. This can be a big help if you're working on a less powerful development machine.

With Docker's support, we can create a virtual machine from an existing image and set it up for user access. These properties enable us to containerize programs or, more significantly, our desired setup environments, which we will make easily accessible to the user. A pre-built Development Environment image will speed up the server setup procedure and the application in general.

4 Azure for website and container deployment.

Microsoft Azure, a popular cloud platform with a wide range of services, is well-suited for website and container deployment. Azure gives developers and IT professionals unrivalled scalability, security, and flexibility as they create, launch, and manage applications.

Azure App Service [12] is a vital component for deploying and managing web apps, giving a number of deployment solutions to satisfy a wide range of application demands. With its uncomplicated setup and configuration, Azure App Service streamlines the process of developing a web application, whether you're starting from scratch or migrating an existing application.

Azure App Service supports a wide number of programming languages and frameworks, including ASP.NET, Node.js, PHP, and Java. Developers can choose the technologies that best match the needs of their applications due of its adaptability. Furthermore, Azure App Service supports a range of deployment options, including App Service Plans, WebJobs, and Functions, offering users flexibility in resource allocation and scaling.

Azure Container Instances (ACI) [13] emerges as a solid solution for containerized applications, enabling straightforward container deployment and maintenance without the requirement for infrastructure administration. ACI frees developers to focus on creating and launching applications by reducing the complexities of standard container orchestration technologies.

Developers may easily use Azure Container Registry, a private container registry within Azure, or Docker Hub to deploy containers with ACI. The capacity of ACI to auto-scale assures optimum resource allocation, optimizing both cost-effectiveness and performance.

Azure App Service and ACI work in collaboration with Azure Pipelines [14], a continuous integration and delivery (CI/CD) platform that enables automated deployments and simplified release management. Azure Pipelines accelerates the development process and provides predictable and consistent application releases by automating the build, test, and deployment processes.

Azure Deployment Manager [15] features a centralized orchestration framework. Developers may use Azure Deployment Manager to define and manage deployment sequences, ensuring that apps are delivered with the correct configuration and in the correct order.

Azure stresses security as well, giving a secure foundation for our installations. Azure delivers network security, data encryption, and identity and access management.

Despite all of these advantages, the pay as you go strategy for deploying this type of website is the most practical and successful. Images are saved in the cloud and used to spin up virtual machines on demand; if a virtual machine is idle for a lengthy period of time, a fresh image of the computer is created, tagged with the user, and stored again in storage. This procedure is repeated for each succeeding instance, suggesting that the pay as you go model is the most reasonable and rational alternative for deploying our program.

5 Automatic Scaling of Azure Virtual Machines [16].

Scalability is one of the key advantages of using virtual machines (VMs). Virtual machines (VMs) can be swiftly scaled to meet changing project demands. As a result, they are a great alternative for projects with different requirements.

There are two sorts of scaling options: vertical scaling and horizontal scaling. The capacity to extend a virtual machine's (VM) RAM, CPU, or storage is referred to as "vertical scalability." Increase the hardware on the physical server where the virtual machine is running to do this. Vertical scalability is a good solution for projects with predictable workloads. Horizontal scalability refers to the ability to add more virtual machines (VMs) to a pool of VMs. This can be performed by installing new virtual machines (VMs) on extra physical servers. Horizontal scaling is a viable option in circumstances when workload fluctuations are a problem.

The Pay as You Go [17] price section also tackles the scalability of a virtual machine. The automated scaling capability relieves the cloud administrator of the load of scaling by automatically scaling the servers up and down if particular circumstances are satisfied.

Azure offers virtual machine scalability and provides a user-friendly dashboard that displays the historical performance of each machine as well as the projected consumption pattern.

6 Microsoft Azure vs Amazon Web Services vs Google Cloud Platform.

Azure, AWS, and GCP are the three cloud computing heavyweights. Each offers a diversified range of services to satisfy a wide range of needs and tastes. But, in the midst of all of this competitiveness, Azure emerges as the clear champion, luring customers with an unrivalled combination of innovation, scalability, and affordability.

Azure's cutting-edge technologies, such as Azure Cognitive Services, which employ artificial intelligence to deliver apps with human-like skills, show the company's forward-thinking approach. Its focus on scalability guarantees that it can respond to changing needs, whether managing traffic surges or integrating new applications.

Azure also shines in terms of cost-effectiveness, offering a choice of pricing options that are tailored to match the demands of diverse enterprises and guarantee that their cloud expenditure is optimized. This is the entire rationale for our usage of Azure in this project, as well as our dedication to it; we seek to fully use Azure to the degree that it can offer us.

7 Azure Virtual Machines, Docker Containers, Browser IDEs [18].

From all of the Research Papers, Extensive Browsing, Using Some Cloud IDEs, and everything else, we can finally say that a cloud service provider like Azure to contain the project's servers and all of its virtual machines, a script to create development environments to create Docker Images, and a fully functional website are the main building blocks for our project, and after successfully fulfilling these three prerequisites, we can say that our project will start looking like goo. VMs provide a scalable and secure platform, Docker integrates all dependencies into one, enabling seamless portability and deployment, and Browser IDEs provide a cloud-based development environment that enables developers to code, debug, deploy, and interact.

CHAPTER 3: SYSTEM DEVELOPMENT

The development of the cloud based integrated development environment system was a hands-on experience for us. Creating a generalized system for this means that we need to create something that can be modified later and tailored to specific needs.

3.1 REQUIREMENTS AND ANALYSIS

3.1.1 CLOUD COMPUTING

Introduction

Cloud computing is a massive development in the quickly developing field of information technology that has fundamentally revolutionized the underlying dynamics of data processing, management, and accessibility. Businesses and industries have entered a paradigm with unsurpassed efficiency, scalability, and flexibility with the emergence of cloud computing.

This section goes into a comprehensive review of the wide issue of cloud computing. It digs deeply into the fundamental requirements and extensive analyses that underpin this game-changing technology. This section tries to provide a clear road map for understanding, implementing, and maximizing the potential of cloud-based solutions in a variety of sectors and companies by undertaking a complete assessment of the necessary conditions and analytical aspects.

The disruptive significance of cloud computing resides in its ability to operate beyond traditional boundaries, creating a dynamic ecosystem of computer resources and services. It has radically altered the way any enterprise view IT infrastructure by giving them on-demand access to a common pool of reconfigurable resources such as networks, servers, storage, and applications. This accessibility, together with minimal maintenance costs and regular network availability, captures the heart of cloud computing's disruptive influence.

The section aims to break down the fundamental criteria needed to navigate the intricate web of cloud-based services in this large ecosystem. Furthermore, it tries to highlight the analytical elements essential for a thorough understanding and strategic deployment of cloud computing across a range of industry sectors.

The purpose of this research is to uncover the essence of cloud computing's disruptive potential and provide stakeholders, decision-makers, and tech enthusiasts with a thorough understanding of its basic components, prerequisites, and analytical frameworks. This section strives to open the door to the infinite possibilities of cloud computing and to inspire efficiency, scalability, and innovation in the digital age by doing so.

Essential Characteristics of cloud:

1. **On-Demand self-service**

   The concept of "on-demand self-service" symbolizes a crucial feature that enables users with unrivalled freedom and flexibility when it comes to using computer resources. This crucial feature announces a paradigm shift by enabling users to autonomously acquire and control computer power without the requirement for direct human engagement from service providers.

   - Unauthorized Use of Computer Resources

     With on-demand self-service, users can get and use computer resources according to their own requirements. It reduces the traditional limits imposed by human intermediaries or cumbersome procurement procedures, allowing for the self-service provisioning of resources such as server time and network storage.
   - Managing Resources Automatically

     Using this functionality, users can automatically access and assign processing capacity in response to changing needs. This independence enables for the instant acquisition of resources without the need for manual approvals, allowing for a swift responsiveness to changing needs.
   - Eliminating Customer-Service Provider Interaction

     The capacity of on-demand self-service to eliminate the requirement for constant human touch with service providers while allocating resources is crucial. This autonomy liberates users from time and geographical constraints, allowing them to access and control computer resources quickly and effectively.
   - Cloud Users' Repercussions

For cloud users, on-demand self-service implies a fundamental shift in operational agility. It helps organizations to instantly distribute and employ resources, granting them previously unheard-of power and flexibility. This feature helps users to manage their computing environment proactively by facilitating scalability, cost-efficiency, and the smooth adaptation of resources to changing workloads.

Essentially, on-demand self-service becomes a pillar of the cloud computing environment, transforming the interaction between user and provider by empowering users to acquire, manage, and optimize computing resources promptly and autonomously. Its impact is seen across a wide spectrum of enterprises, affecting how cloud-based services are accessed and used in today's fast-paced, ever-changing digital environment.

2. **Broad network access.**

In the context of cloud computing, the essential feature of wide network access provides an unequalled accessibility paradigm that enables users to effortlessly leverage cloud-based capabilities on a range of devices and platforms. This feature underscores the inclusive and universal nature of cloud services, making it easier to use them with a range of client platforms and standard procedures.

The availability of abilities in general Through the Network Broad network connectivity ensures that the cloud architecture's capabilities and services are continually accessible over the network. Cloud services can now be available from a range of computer devices, including laptops, workstations, tablets, and mobile phones, thanks to this access. It also goes beyond the limits of specific devices or platforms.

- Entry Points Identified

  The employment of standardized access mechanisms is vital to broad network access. Users can communicate with cloud-based services utilizing these standardized protocols and interfaces, which allow compatibility and interoperability across heterogeneous thin and thick client systems.
- Encouragement of the Use of a Variety of Devices

Cloud services provide seamless access regardless of a device's form factor or computing capability by encouraging usage across many platforms and devices, such as thin or thick clients. This adaptability promotes user comfort and flexibility by allowing cloud services to be used on devices that match specific needs or organizational goals.

- User Experience Consequences

  Cloud computing provides its consumers with a varied spectrum of network access, resulting in a more comprehensive and adaptable experience. It overcomes the hurdles caused by device-specific limitations, allowing users to effortlessly access and use cloud services across multiple platforms. This enhances accessibility and productivity.

3. Resource Pooling

Resource pooling is an important component of cloud computing architecture because it provides a dynamic model in which the provider's computer resources combine and are appropriately shared among several clients. This feature leverages a multi-tenant paradigm to dynamically assign various physical and virtual resources in response to changing client demand.

- Multi-Tenant Resource Allocation

  In the context of resource pooling, cloud providers employ a multi-tenant model to share resources among numerous users or tenants. This pooling technique maximizes resource consumption by sharing resources like storage, computational power, memory, and network bandwidth across several users in an ideal manner.

- Assignment and reallocation in real time

  The key to resource pooling is its dynamic character; rather than being statically allotted, resources are dynamically distributed and reassigned in response to changing customer demands. This dynamic allocation assures optimal resource consumption by allowing for varying workloads and eliminating under- or overprovisioning.

- Location Abstraction and Independence

  Customers acquire an impression of location freedom as a result of resource pooling. Users can express their preferences at a higher level of abstraction, such as choosing the area or data centre where their resources are given, even if they frequently have no explicit control over the precise physical location of the resources that are distributed.

- Customer Repercussions

  Users of cloud services benefit from greater scalability and flexibility as a result of resource pooling. Because of it, users can access and use resources without having to worry about the complexity of the physical infrastructure. This abstraction enhances cost-effectiveness and efficiency by allowing users to scale resources up or down based on their needs.

As a result, resource pooling is crucial to cloud computing, enabling an environment in which resources are dynamically regulated and effectively utilized to suit the needs of a diverse collection of users. It shows how computing paradigms are changing toward a shared, adaptive, and flexible infrastructure that provides scalable resources in a virtualized context.

4. Rapid Elasticity

Rapid elasticity is a unique property of cloud computing that demonstrates the exceptional adaptability and scalability that cloud services give to consumers. This key feature helps clients to rapidly supply and de-provision capabilities by allowing them to dynamically scale computer resources in response to changing demands.

- Demand-driven flexible scaling

  Quick flexibility provides for the smooth and fast changing of processing power to match current demand. This versatility is illustrated by the ability to rapidly scale resources inward during periods of low consumption or outward during periods of high demand, guaranteeing optimal resource allocation at all times.

- Automation of Provisioning and Release

  Automation is commonly employed in quick flexibility to provision and release resources while eliminating the demand for human engagement. This automation speeds the process, allowing for speedy alterations to resource distribution without requiring substantial administrative labour or manual intervention.

- The concept of infinite potential

  From the user's perspective, the possibilities that might be supplied in a cloud environment frequently appear to be nearly unlimited. By allowing users to appropriate computing resources in any quantity at any moment, this feature enables an environment in which scaling limits are essentially non-existent.

3.1.2 SERVICE MODELS OF CLOUD: SOFTWARE AS A SERVICE

Software as a business (SaaS) is a prominent business model in cloud computing that offers clients with access to provider-hosted apps that are set up over a cloud infrastructure. Users interact with programs directly through the internet in this paradigm, eliminating the need for local installs. They accomplish this through the use of a web browser or a program interface.

- Client Device Interoperability
  By letting users access apps from a number of client devices, SaaS promotes flexibility and accessibility. Users can use apps using a program interface or a thin client interface, such as a web browser, allowing for seamless device-to-device interaction without the need for specialized local installations.
- Consumers' Disengagement from Infrastructure Management
  A major feature of SaaS is the consumer giving control over infrastructure management. Users do not manage or control the underlying cloud infrastructure, which comprises servers, network components, operating systems, and storage. Furthermore, with the exception of a few user-specific setup choices, users often have no control over the real capabilities of any application.
- The provider is responsible for the infrastructure's upkeep.
  The supplier controls all infrastructure upkeep and administration under the SaaS model, assuring flawless application performance and delivery. This arrangement relieves users of the effort of managing the infrastructure, allowing them to focus exclusively on enjoying the software's features.
- Customer Impacts
  When clients use SaaS, the paradigm simplifies application access and provides a hassle-free experience without the requirement for infrastructure upkeep. Users often have limited control over application setups outside user-specific options, therefore this ease comes at the tradeoff of less customization.

Software as a Service (SaaS) is the apex of accessibility and simplicity in cloud-based software distribution. By making programs easily accessible over the cloud, SaaS relieves users of infrastructure administration. This boosts productivity and accessibility while maintaining application availability across a number of client devices.

3.1.3 INTEGRATED DEVELOPMENT ENVIRONMENT

An Integrated Development Environment (IDE) is a comprehensive software tool meant to improve and accelerate the software development process. It functions as a graphical user interface (GUI) workbench, consolidating several features and tools necessary for developing software applications into a single location.

- Workbench for Software Development
  An IDE, which serves as a centralized workspace, integrates all of the software development tools. It provides an integrated environment that includes all the necessary tools for developing applications successfully, including version control systems, data structure browsers, code editors, debuggers, and other crucial utilities.
- Streamlining Developers' Workflow
  An IDE's primary goal is to facilitate developers' workflow by providing a standardized interface that removes the need to switch between various tools or applications. The smooth integration of features such as code editing, debugging, version control, and project management allows developers to work quickly and effectively in a single environment.
- The learning curve and increasing production
  By combining tools, an IDE offers developers a uniform interface and dramatically boosts productivity. It speeds up the completion of numerous development processes, simplifies action execution, and provides a single user interface (UI) for connected components. This homogeneity minimizes the learning curve for developers who utilize several languages or work on multiple projects.
- A Wide Range of Tools and Languages are Supported
  One of the most remarkable features of an IDE is its ability to support a wide range of programming languages and tools. It fulfils the demands of developers

working with a range of programming languages by providing language-specific features like syntax highlighting, auto-completion, and debugging tools.

- Evolution of Development Practice
  Previously, developers had to traverse through numerous tools for debugging, linking, editing, and compilation. Today's IDEs merge these features into a single unified platform, greatly simplifying the development process and promoting an environment in which developers can efficiently manage their projects from conception to implementation.

An Integrated Development Environment (IDE) is a vital technology that enhances software development processes by unifying relevant tools, streamlining procedures, and presenting developers with a unified and effective workspace for constructing complex and dependable software applications.

3.1.4 SHELL: INTERFACE TO THE UNIX SYSTEM

The shell is vital in the UNIX system because it acts as a bridge between users and the underlying operating system. It operates by taking user input, running instructions or programs in response to that input, and then showing the results when the execution is complete. The shell, in essence, provides a user-interactive environment for command execution, application execution, system management, and shell script management.

- Important Characteristics
  Fundamentally, the shell supports the execution of UNIX programs and commands. It decodes user commands and turns them into instructions that the operating system can grasp and execute. Once a program has finished running, the output is displayed to the user via the shell.

- Shells of Various Shapes
  Shells, like operating systems, come in a variety of flavours or varieties, each with its own set of recognized commands and capabilities. Bash (Bourne Again Shell), C Shell (csh), Korn Shell (ksh), and other popular shells provide configurable features and command syntaxes based on user preferences or required system requirements.

- Quick command and execution

  The shell shows the command prompt (symbolized by characters such as '$') to tell the user that it is ready for input. When requested, users can input commands by typing them in. When the Enter key is pushed, the shell scans the input and searches for the first word—a string of characters separated by tabs or spaces—to determine which command should be run.

- Function in Command Interpretation

  The shell's principal job is to understand and execute user instructions within the operating system. It functions as a command translator, understanding user input, locating the relevant application or command, and commencing execution based on user inputs.

- Relevance to System Interaction

  Because it serves as a conduit for user interaction with the operating system, the shell is a crucial part of UNIX-based systems. Its features include shell script automation, program execution, command interpretation, and an easy-to-use interface that lets users run programs, effectively manage system operations, and automate tasks.

3.1.5 PYTHON: A HIGH-LEVEL PROGRAMMING LANGUAGE [19]

Python is a well-liked high-level programming language that is easy to learn, flexible, and uncomplicated. Since its invention in the late 1980s by Guido van Rossum, Python has gained popularity across a variety of industries thanks to its user-friendly syntax, extensive standard library, and active community.

- Readability and Usability

  Python's emphasis on readability and simplicity is one of its distinguishing traits. Because of the language's compact and straightforward syntax, programmers can express thoughts and concepts in less lines of code than in other languages. Its readability not only makes the code easier to use and maintain, but it also aids in code comprehension.

- Flexibility and a Large Library

  Python's vast library and framework ecosystem provides developers with a wealth of resources and tools for a wide range of applications. Python's vast standard libraries, which encompass web development, data analysis, machine learning, and scientific computing, contribute to its versatility in a wide range of domains.

- Typing Comprehended and Adaptive
  Python is an interpreted language, which means that it executes code without first compiling it. This feature allows for brief code section trials and testing, which aids in rapid prototyping and development. Python also includes dynamic typing, which promotes flexibility while requiring type consistency due to variables' potential to change types dynamically while being executed.

- Philosophies of Open Source and Community
  The Python community makes significant contributions to the language's evolution. The active community of Python developers and enthusiasts fosters cooperation, knowledge sharing, and ongoing advancement. Since it is open-source, contributions are welcome and lead to frequent updates, enhancements, and a robust community of third-party modules and packages.

- Applications in a Variety of Fields
  Python is widely used in many different fields, including web development (using frameworks like Django and Flask), scientific computing, automation, scripting, data analysis and visualization (using libraries like Pandas, NumPy, and Matplotlib), artificial intelligence and machine learning (using TensorFlow and PyTorch), and many more. Both inexperienced and seasoned developers find its versatility and ease of use appealing.

Finally, Python's appeal in the programming community can be due to its simplicity, versatility, vast library, and welcoming community. Python is an excellent choice for beginners, experts, and businesses searching for a powerful yet approachable programming language for a variety of jobs and solutions due to its adaptability and accessibility.

3.1.6 DJANGO: A HIGH-LEVEL PROGRAMMING LANGUAGE [20]

Django is a strong, high-level web framework built on Python that is known for its scalability, efficiency, and adherence to the maxims "explicit is better than implicit" and "don't repeat yourself" (DRY). Django, a web development framework, makes it simple to design complex, database-driven websites. It accomplishes this by providing a wide range of tools and functionalities.

- Quick Development Structure
  Django's principal objective is to make it easier to construct web applications. It simplifies core web development activities and lets you to construct feature-rich websites with less coding work thanks to its many built-in capabilities and conventions.

- Model-View-Template (MVT) Architecture
  Django code components are arranged using the Model-View-Template (MVT) architectural design, which is comparable to the Model-View-Controller (MVC) architecture. This separation of responsibilities increases maintainability and scalability by clearly distinguishing and maintaining data models (Model), user interface logic (View), and presentation templates (Template).

- Battery-Powered Concept and Integrated Parts
  One of Django's differentiating features is its "batteries-included" approach, which provides a large range of built-in components and functionalities. These capabilities, which include an ORM (Object-Relational Mapping) for database connection, a sophisticated admin panel for site administration, URL routing, form handling, authentication, and strong security measures, substantially simplify the design process.

- Scalability and adaptability
  Django's ability to manage projects of different size and complexity proves its scalability. Django's essential stability is retained thanks to its modular architecture, while developers can integrate new features via third-party packages or extensions, boosting flexibility.

- Community help and documentation
  Django has a strong community that supports cooperation, information exchange, and constant development. Its extensive documentation and profusion of tutorials make learning and troubleshooting information easily accessible to developers.

- Case Studies and Applications

Django applications can be found in a wide range of web development projects, including scientific computing platforms, social media sites, e-commerce websites, content management systems (CMS), and others. Because of its adaptability and scalability, it is the finest solution for both small-scale projects and enterprise-level systems.

3.1.7 JAVASCRIPT [21]

JavaScript is a dynamic and adaptive computer language that is mostly used for web development. Brendan Eich introduced JavaScript in the mid-1990s, and it has matured into a vital tool for developing dynamic, interactive websites with features that improve user interaction and experience.

- Client-Side Scripting Language
  JavaScript is extensively utilized as a client-side scripting language in web browsers. It lets web developers to communicate with webpage elements, dynamically update information, respond to user activities, and create interactive features without requiring server-side processing.

- The Versatility and Nature of Different Paradigms
  JavaScript is a multi-paradigm programming language that may be used to write both object-oriented and procedural programs. Developers may add a number of capabilities, including as asynchronous programming, DOM manipulation, event management, and more, due to its adaptability.

- Environmental Concerns and Widespread Use
  JavaScript is a widely used internet technology that is crucial for web development. Its ecosystem offers a bevy of tools and frameworks (like React, Angular, and Vue.js) that facilitate development and make it easier to construct complex, feature-rich online apps.

- Asynchronous Programming using Promises and Async/Await
  Because JavaScript is asynchronous, it can execute without generating a block, making it handy for tasks such as receiving data from servers or completing lengthy procedures without crashing the user interface. Async/Await and

Promises, two contemporary JavaScript technologies, simplified asynchronous code administration and boosted readability.

- Server-Side Development with Node.js
  With the debut of Node.js, the sphere of application for JavaScript was enlarged from the browser to server-side development. JavaScript may now be used to develop scalable and effective server applications, providing for high-performance server-side programming, thanks to Node.js' event-driven design.
- Uniformity and Constant Development
  JavaScript is continually being updated and developed in order to improve language features, security, and speed. Standardization groups such as ECMAScript aid in the preservation of consistency and interoperability among diverse JavaScript implementations.
- Several Use Cases
  JavaScript is widely used and adaptable, with applications in a variety of domains including web development, mobile app development (using frameworks such as React Native and Ionic), game development (using frameworks such as Phaser and Three.js), server-side applications, IoT (Internet of Things), and more.

JavaScript is a vital component of current web development, allowing programmers to create dynamic, responsive, and engaging web applications for a number of platforms. This underscores JavaScript's significance in the digital world.

3.1.8 REACT: THE LIBRARY FOR WEB AND NATIVE USER INTERFACES [22]

Facebook invented React, a sophisticated and popular JavaScript library for designing user interfaces (UIs). React, a prominent framework recognized for its efficiency, adaptability, and component-based design, has altered how web developers construct dynamic, interactive products.

- Component-Based Architecture
  React's architecture is founded on components. React allows developers to create complex user interfaces by assembling reusable and encapsulated user interface

components. This modular paradigm increases code reuse, speeds up development, and simplifies maintenance.

- Advanced DOM for Efficient Rendering
  React makes use of Document Object Model (DOM) virtualization to boost rendering efficiency. Because it builds an in-memory model of the actual DOM, React can efficiently update and render just the components that have changed. This method increases application speed by eliminating needless re-renders.

- Declarative and proactive
  Because React is declarative, developers may describe the user interface (UI) based on its present state rather of needing to specify how to update the DOM. The reactive nature of React also ensures that user interface components automatically adapt to changes in data or state, making UI and data synchronization easier. JSX, or JavaScript Syntax Extension, is utilized by React. JSX is an extension that allows programmers to write JavaScript code that seems like HTML. This method streamlines the design of component structures, making it easier to visualize and maintain user interface elements inside the codebase.

- Data Flow in Only One Direction
  React implements a unidirectional data flow, in which information goes entirely in one way from parent to child components. Debugging is simpler since the consistent and transparent data flow makes it easy to determine the source of faults within the application.

- Rich Ecosystem and Community Support
  React has a healthy ecosystem thanks of a number of modules and utilities that extend its features (such as Redux for state management and React Router for routing). The abundance of tools, tutorials, and pre-built components offered by the powerful community support makes creation straightforward.

- Interoperability Across Multiple Platforms
  React's versatility extends beyond online apps. Native applications for the iOS and Android platforms can be produced using frameworks such as React Native, which uses React's component-based architecture. As a result, internet and mobile apps can share a substantial percentage of the codebase.

- Usage and Industry Acceptance
  Because of the rising acceptance of React across a range of industries, developers can now construct interactive user interfaces for social networking platforms, e-commerce websites, streaming services, and more. Because of its performance, flexibility, and ease of use, it is the favoured solution for many web development projects.

By promoting component-based architecture, superb rendering, and a declarative technique, React has significantly revolutionized the online development scene. It has also provided developers with a strong and efficient toolkit for developing scalable and interactive user interfaces.

3.1.9 DOCKER [23]

Docker is a sophisticated platform that harnesses containerization technology to simplify program development, delivery, and operation. It provides a standardized technique of packaging applications and their dependencies into lightweight, portable containers to ensure interoperability across varied settings.

- Containerization Technology
  Docker employs containerization technology to encapsulate applications, as well as their libraries and dependencies, inside containers. Because of the isolation, portability, and low weight of these containers, programs can function reliably in a range of settings, from development to production.

- Docker Images and Containers
  Docker is based on two core concepts: images and containers. Docker images are architectural blueprints that specify an application's requirements and environment. These images are used to generate containers, which are instances that run as executable, isolated packages and ensure consistent application performance regardless of the underlying infrastructure.

- Mobility and Consistency
  Docker's most significant characteristics are its stability and portability. Because Dockerized programs may operate on any system that supports Docker, developers may design apps in one environment and deploy them effortlessly

across numerous platforms, including cloud environments, without encountering compatibility difficulties.

- Development Process Simplified
  Docker simplifies the software development process by making it easy to establish self-contained and repeatable environments. Developers can use Docker Compose to build multi-container configurations, compose programs, and manage the multiple services required for development, testing, and debugging.
- Scalability and resource efficiency
  Docker's container-based architecture enables for scalability and optimal resource use. Because containers share resources with the host system, they can run independently in numerous isolated instances on the same machine. Because of its scalability, Docker is an ideal solution for constructing microservices-based systems.
- Environment and Docker Hub
  Docker Hub, as a centralized repository for Docker images, offers a massive library of pre-built images donated by users. This ecosystem accelerates development by allowing developers to use pre-existing photos, edit them as needed, and then share their improved images with others.
- Acceptance in the Industry and Applications
  Docker is widely deployed in a range of industries, ranging from cloud computing and IoT to software development and DevOps. It is applied in the automation of container orchestration (using tools such as Kubernetes), continuous integration and delivery (CI/CD), and repetitive development environments.
- Prospective Trends and Continuous Improvement
  Docker is still changing as technology does. It maintains up with the newest containerization technology breakthroughs, fixing security vulnerabilities, increasing performance, and partnering with the community to expand its feature set.

Docker has fundamentally transformed the way programs are produced, packaged, and deployed by providing a consistent and effective technique of controlling application environments through containerization. This strategy has boosted the agility, scalability, and portability of current software development operations.

3.1.10 AZURE CONTAINER SERVICES

Azure is a cloud computing platform which provides cloud related services. Azure is developed by Microsoft. Azure offers a wide range of cloud services, including computing, analytics, storage, networking, machine learning and artificial intelligence tools. Azure allows user to deploy, host and manage applications and services through Microsoft-managed data centres. User can use Azure to launch Virtual Machines from a wide range of operating systems, store data in databases, internet of things (IoT) solutions.

Azure provides some services related to containers and use of these containers. Generally called Azure Container Services (ACS). These include storing the container in some specified cloud storage called Azure Container Registry (ACR), using these stored containers to deploy or host applications called Azure Container Apps (ACA), to do so the Azure automatically makes a cloud container environment and then hosts the application and gives us a link to follow to our application.

Azure Container Registry enables developers and organisations to securely store and manage their docker container images in the cloud. Some features of ACR include Private Registry, Scalability, Integration with Azure Services, Security, Geo-Replication and Webhooks.

Azure Container Apps (ACA) is a fully managed serverless container service. It enables developers to run containerized applications without managing infrastructure that lies behind an application. ACA removes all the complexities of container orchestration that comes with it and also removes the burden of infrastructure management. Key features of ACA can include serverless infrastructure, supports docker containerization, ACA supports automatic scaling, compatibility with other azure services and pay-as-you-go pricing model making it easy to deploy anytime. It also includes Log Monitoring and Analytics.

3.2 SYSTEM DESIGN

3.2.1 BASIC DESIGN IDEA

The system architecture of the cloud-based IDE is designed to ease code development, compilation, and execution in a distributed environment. In this client-server architecture, a web browser serves as the client, while a backend consisting of web servers and compilers serves as the backend. This is a more detailed explanation of how the system works:

User Interaction:

A user is someone who accesses the IDE using a web browser. The user initiates actions such as authoring, running, and debugging code.

Client-side, web browsers serve as the client interface. It is in charge of managing input/output processes and displaying the IDE's graphical user interface (GUI). When users conduct operations, such as executing a software, they send requests to the web server.

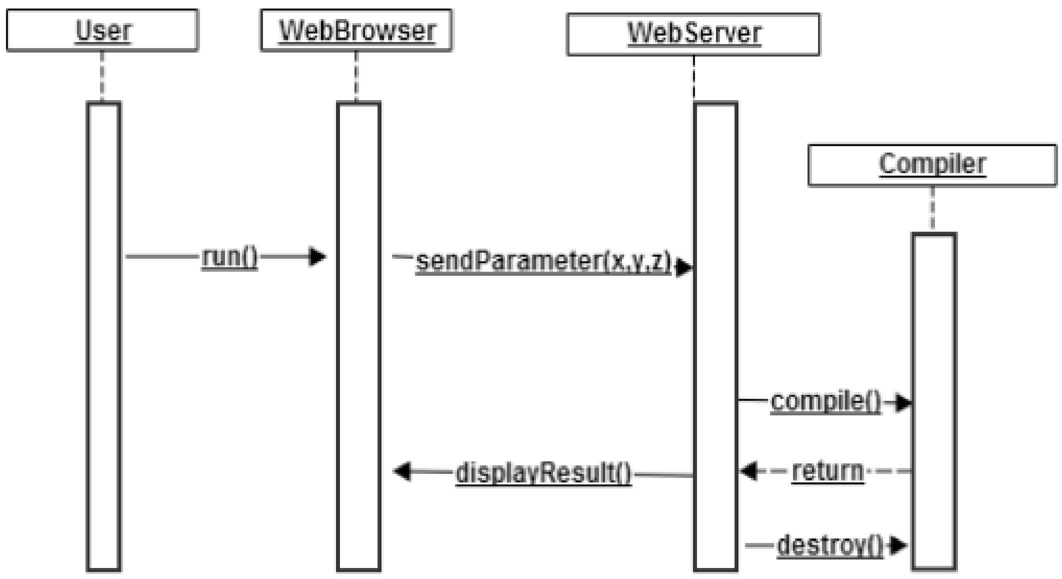


Figure 3.1: Working of Browser based IDE

Server-Side:

The web server serves as the IDE's middleware as well as its primary processing unit. It assesses requests received from web browsers to decide what needs to be done. In response to code execution requests, it bundles the code parameters and communicates them to the compiler service.

A compiler is a specialized service that compiles and runs code. It compiles the code in an isolated and secure environment, then runs it using the code parameters it acquired from the web server. Following that, the web server receives the compilation and execution results, which may contain error warnings or output data.

Response Flow:

The web server processes the results depending on the information provided by the compiler. This could include repairing any problems that happened during compilation or execution, as well as formatting the output.

After receiving the processed results from the web server, the web browser delivers the output or error notifications to the user.

Resource management and cleanup:

Following execution, the compiler service cleans away any leftover junk to guarantee optimal system resource management. In preparation for the next action, this entails deleting temporary files, reallocating memory, and resetting the environment.

3.2.2 LOCAL OS DEVELOPMENT VS REMOTE OS DEVELOPMENT

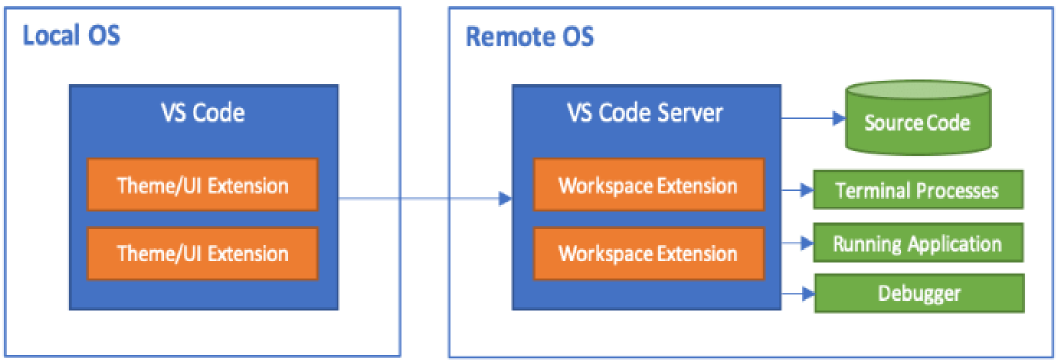


Figure 3.2: Difference between Local OS and Remote OS

Local Development:

- VS Code on Local OS: The developer's computer is where the development environment is immediately configured. The local operating system houses the codebase, extensions, and all of the tools.
- Theme/UI Extension: These are functional and graphical add-ons for the local Visual Studio Code that improve the user interface and overall experience. They have no effect on the remote server; they solely have an effect on the local development environment.
- Direct Interaction: Using their computers, developers write, test, and execute the code directly. Compilers, debuggers, source code, and the whole development stack are all hosted locally.

Resource Dependency:

- The local machine's resources CPU, memory, and storage directly affect the development environment's performance and capabilities.
- Remote Development: VS Code server on remote OS: Frequently hosted in the cloud, VS Code server operates on a remote computer or server. The codebase and development environment are hosted on this distant server.
- Workspace Extension: The remote server is configured with workspace-specific extensions. Compilers, linters, and other instruments necessary for the development process might be among them.
- Source Code: Without needing to clone the code onto their local computers, developers can view and edit it thanks to the source code being hosted on the remote server.
- Terminal Processes: To utilize the resources of the remote server, terminal processes such scripts, test runners, and build commands are run there.
- Running the Application: It is advantageous to test in an environment that is similar to the production setup because the application operates in the remote environment.
- Debugger: To troubleshoot remotely, the IDE on the local workstation establishes a connection with the remote environment.

Principal Disparities:

- Consistency of Environment: Since all team members connect to the same remote setup, remote development guarantees uniformity in the development environment.
- Resource Utilization: Since heavy-duty tasks are transferred to the distant server, remote development places less strain on the local machine's resources.
- Accessibility: Remote development is perfect for distributed teams and remote work situations since it can be accessible from any location.
- Security: Keeping the source code on a distant server can increase security, particularly if the server is housed in a secure network or cloud environment.
- Complexity of Setup: Setting up local development environments for particular projects can be quicker and easier, but ensuring uniformity throughout teams can be difficult. Although it involves initial server configuration, remote development has the potential to be more manageable and scalable over time.

In conclusion, remote development takes place on an external server and offers benefits in terms of consistency, teamwork, and resource efficiency, whereas local development is contained within a developer's computer.

3.2.3 HIGH LEVEL SYSTEM DESIGN

1. Client Interface:

   Web-based Frontend: This interface serves as the user's entry point, providing a visually appealing and intuitive environment for code editing and interaction. It must support various devices and browsers, offering a responsive design for optimal user experience.

   Authentication: Implementing robust authentication mechanisms ensures secure user access to the IDE, employing protocols like OAuth or SAML for secure identity verification, access control, and user session management.

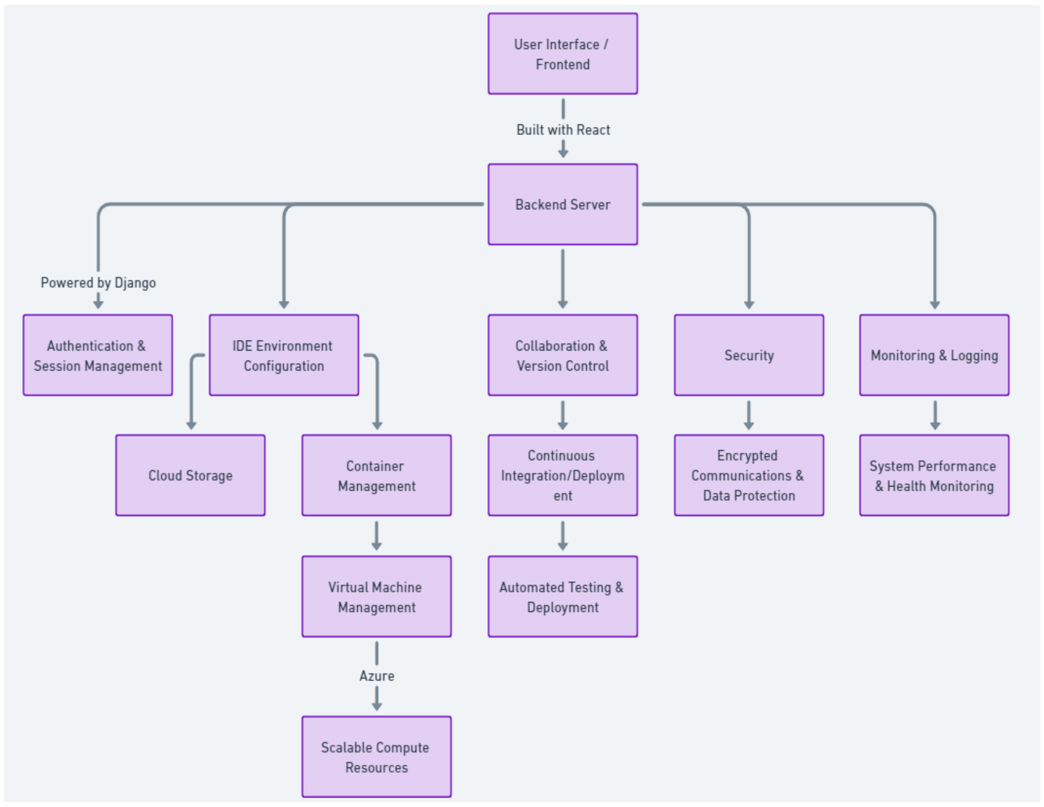

User Interface / Frontend
Built with React
Backend Server
Powered by Django
Authentication & Session Management
IDE Environment Configuration
Collaboration & Version Control
Security
Monitoring & Logging
Cloud Storage
Container Management
Continuous Integration/Deployment
Encrypted Communications & Data Protection
System Performance & Health Monitoring
Virtual Machine Management
Automated Testing & Deployment
Azure
Scalable Compute Resources

Figure 3.3: High Level System Design

2. Backend Services:

   Code Editor and Compiler/Interpreter Services: These server-side components handle code editing functionalities, compilation, and execution of code across various programming languages.

   File Storage: Utilizing cloud-based storage solutions enables efficient and scalable storage for code files and project data, offering reliability, accessibility, and data redundancy.

   Database: Storing user information, project configurations, and metadata requires a reliable and scalable database system to ensure data integrity and efficient retrieval.

3. Collaboration Tools:

   Version Control Integration: Integrating with version control systems like Git allows users to manage code versions effectively, ensuring collaboration and code history preservation.

Real-time Collaboration: Facilitating simultaneous work on the same project by multiple users through real-time editing, commenting, and sharing of code changes.

4. Cloud Infrastructure:

   Scalable Compute Resources: Leveraging cloud-based scalable compute resources ensures the IDE can handle varying workloads and user demands effectively.

   Containerization and Orchestration: Employing Docker and Kubernetes streamlines deployment, enhances resource utilization, and enables efficient management of services.

   Load Balancing: Distributing incoming traffic across servers ensures optimal performance, availability, and reliability.

5. Security:

   Data Encryption: Implementing robust encryption protocols for data in transit and at rest to safeguard sensitive information.

   Access Control: Implementing role-based access control (RBAC) ensures users have appropriate permissions, limiting access to critical resources.

   Network Security: Deploying firewalls, VPNs, and intrusion detection systems to protect against external threats and unauthorized access.

6. Monitoring and Logging:

   Performance Monitoring: Utilizing tools like Prometheus or AWS CloudWatch to monitor system health, performance metrics, and resource utilization.

   Logging: Collecting and analysing logs to identify issues, troubleshoot errors, and ensure compliance with auditing requirements.

7. Integration with Other Cloud Services:

   APIs for Extensibility: Providing APIs allows integration with various cloud-based services, databases, AI/ML platforms, enhancing the IDE's capabilities.

Serverless Tasks: Utilizing serverless functions for specific, on-demand tasks that can scale automatically based on usage.

8. DevOps:

   Continuous Integration/Continuous Deployment (CI/CD): Automating deployment pipelines using CI/CD tools ensures swift and reliable delivery of code changes to production environments, facilitating faster iterations and updates.

9. User Support and Documentation:

   Help Centre: Offering comprehensive user guides, FAQs, and support channels to assist users in utilizing the IDE efficiently.

   Interactive Tutorials: Providing interactive tutorials helps onboard new users and ensures they understand and utilize the platform's features effectively.

10. Disaster Recovery and Data Backup:

    Regular Backups: Implementing regular backups ensures data safety and minimizes the risk of data loss.

    Redundancy: Utilizing redundant systems and disaster recovery strategies to maintain high availability and minimize service disruptions in case of failures or disasters.

3.3 DATA WAREHOUSING

The vast amount of data accessible can be used creatively in a cloud-based Integrated Development Environment (IDE) with data warehousing to improve user experience:

- **Customized work Environments:** By evaluating users' work patterns and preferences, the IDE may automatically alter settings, offer tools, and provide templates that correspond to specific coding approaches, boosting productivity and comfort.
- **Suggestions for Events and Workshops:** Using information about user interests and previous event participation, appropriate workshops, hackathons, or seminars can be presented, supporting lifetime learning and skill building.

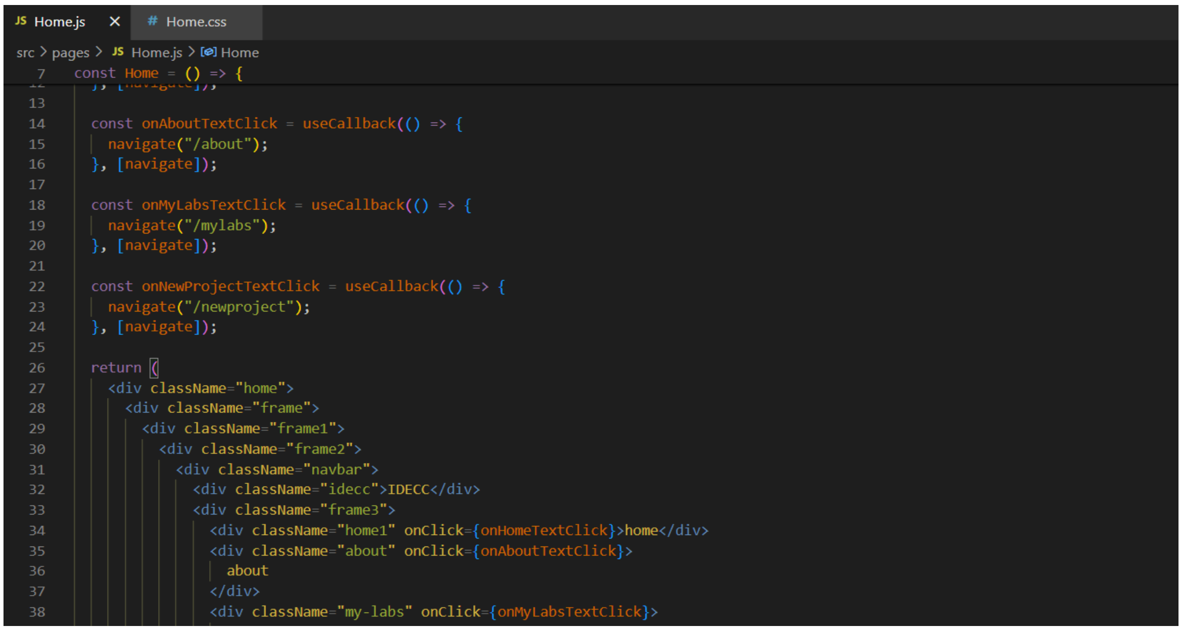
- **Collaboration and networking opportunities:** By connecting users with similar abilities or interests, the IDE can identify possible project partners, increasing innovation and a community of practice.
- **Resource Optimization:** By evaluating consumption patterns, cloud resources can be allocated more effectively, ensuring that users have access to the processing power they require while not overburdening the system.
- **Personalized Learning Pathways:** The IDE can recommend learning modules, documentation, and tutorials based on the learner's learning objectives and current skill level.
- **Real-time Code Analysis and Feedback:** By leveraging data from numerous coding projects, the IDE may deliver real-time recommendations, error detection, and code optimization guidance to increase code quality and efficiency.
- **Project Trend Analysis:** Using information on the most popular frameworks, languages, and tools within their user base, people can select technologies that are in demand or becoming more popular.
- **Improved Security:** By monitoring and analysing code patterns and network connections, the system may detect and alert users of potential security concerns in their projects.
- **Predictive troubleshooting:** Using previous data and trends, the IDE can anticipate and handle frequent problems or bottlenecks that developers may encounter.
- **Integration with Academic and Research Activities:** Educational institutions can improve academic outcomes and accelerate procedures by merging research projects, coursework, and academic activities into an integrated development environment (IDE).

A data warehousing-enabled cloud-based IDE can greatly improve user experience by enabling flexibility, higher productivity, and a comprehensive awareness of user requirements and habits.

3.4 IMPLEMENTATION

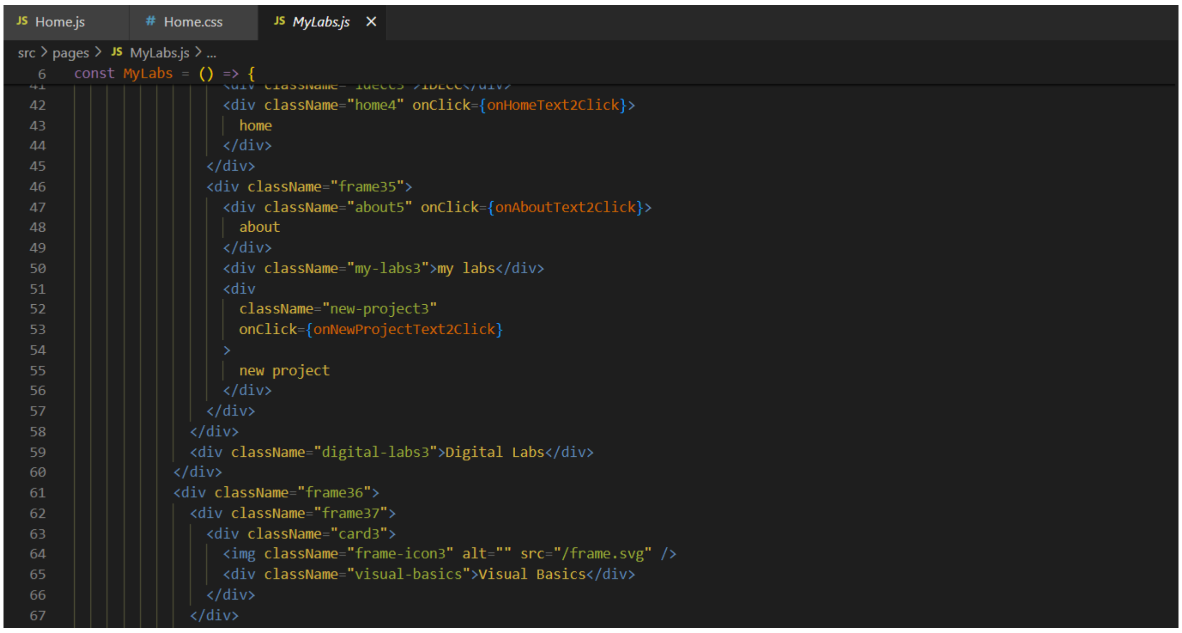
3.4.1 FRONTEND IMPLEMENTATION

Figure 3.4 & *Figure 3.5* show the implementation code of the Frontend Website, made using React Framework. HTML, CSS & JS are used to implement this.



```
const Home = () => {
  const onAboutTextClick = useCallback(() => {
    navigate("/about");
  }, [navigate]);

  const onMyLabsTextClick = useCallback(() => {
    navigate("/mylabs");
  }, [navigate]);

  const onNewProjectTextClick = useCallback(() => {
    navigate("/newproject");
  }, [navigate]);

  return (
    <div className="home">
      <div className="frame">
        <div className="frame1">
          <div className="frame2">
            <div className="navbar">
              <div className="idecc">IDECC</div>
              <div className="frame3">
                <div className="home1" onClick={onHomeTextClick}>home</div>
                <div className="about" onClick={onAboutTextClick}>
                  about
                </div>
                <div className="my-labs" onClick={onMyLabsTextClick}>
```

Figure 3.4: Home Page React Code

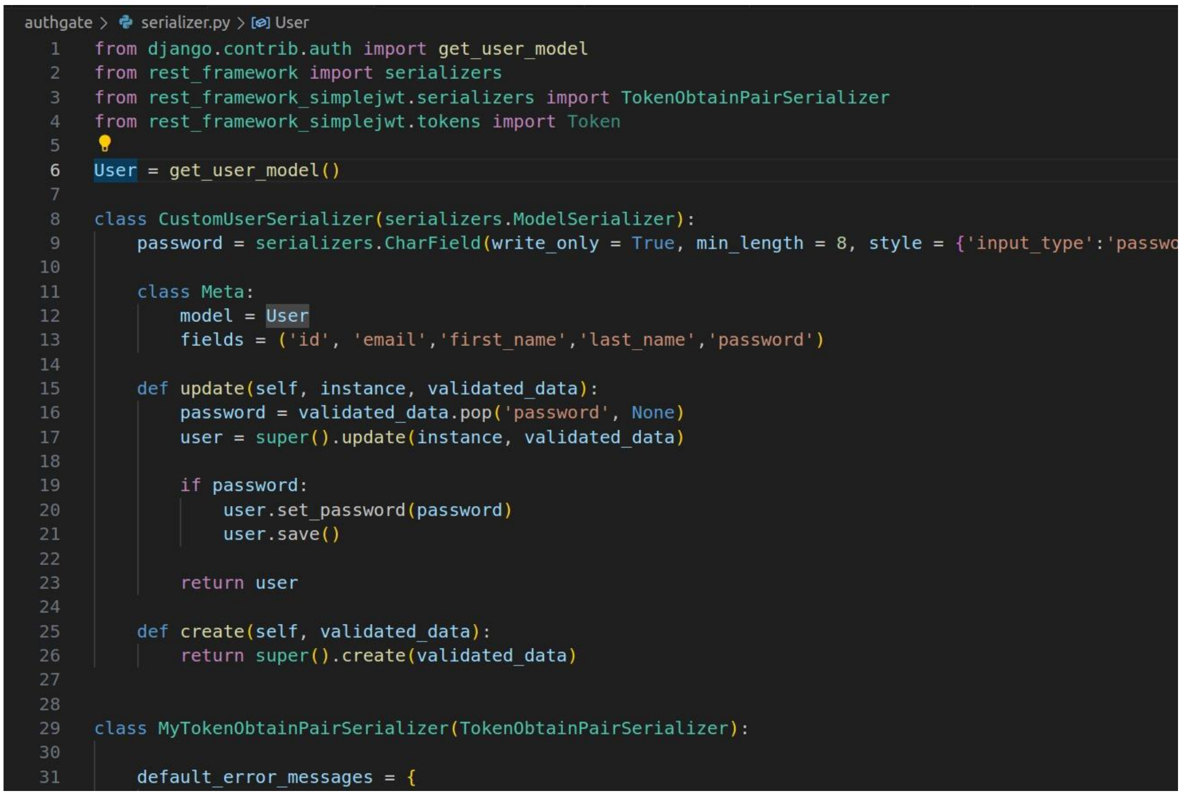


```
const MyLabs = () => {
                  <div className="home4" onClick={onHomeText2Click}>
                    home
                  </div>
                </div>
                <div className="frame35">
                  <div className="about5" onClick={onAboutText2Click}>
                    about
                  </div>
                  <div className="my-labs3">my labs</div>
                  <div
                    className="new-project3"
                    onClick={onNewProjectText2Click}
                  >
                    new project
                  </div>
                </div>
              </div>
              <div className="digital-labs3">Digital Labs</div>
            </div>
            <div className="frame36">
              <div className="frame37">
                <div className="card3">
                  <img className="frame-icon3" alt="" src="/frame.svg" />
                  <div className="visual-basics">Visual Basics</div>
                </div>
              </div>
```

Figure 3.5: My Labs Page React Code

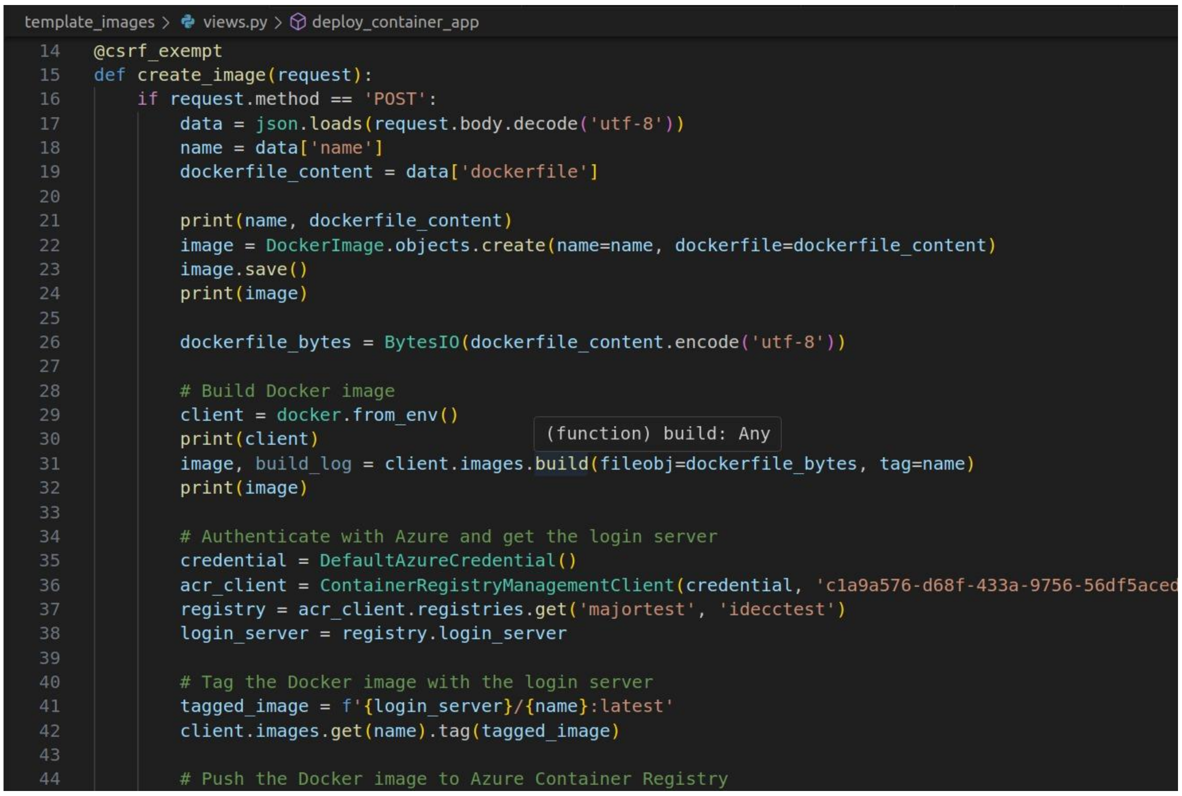
3.4.2 BACKEND IMPLEMENTATION

Backend of the project is made in python-based Django framework. And Rest Architecture has been used for standardisation and scalability. (refer *Figure 3.6* & *3.7*)



```python
from django.contrib.auth import get_user_model
from rest_framework import serializers
from rest_framework_simplejwt.serializers import TokenObtainPairSerializer
from rest_framework_simplejwt.tokens import Token

User = get_user_model()

class CustomUserSerializer(serializers.ModelSerializer):
    password = serializers.CharField(write_only = True, min_length = 8, style = {'input_type':'passwo

    class Meta:
        model = User
        fields = ('id', 'email','first_name','last_name','password')

    def update(self, instance, validated_data):
        password = validated_data.pop('password', None)
        user = super().update(instance, validated_data)

        if password:
            user.set_password(password)
            user.save()

        return user

    def create(self, validated_data):
        return super().create(validated_data)


class MyTokenObtainPairSerializer(TokenObtainPairSerializer):

    default_error_messages = {
```

Figure 3.6: Authentication & Authorization code in Django



```python
@csrf_exempt
def create_image(request):
    if request.method == 'POST':
        data = json.loads(request.body.decode('utf-8'))
        name = data['name']
        dockerfile_content = data['dockerfile']

        print(name, dockerfile_content)
        image = DockerImage.objects.create(name=name, dockerfile=dockerfile_content)
        image.save()
        print(image)

        dockerfile_bytes = BytesIO(dockerfile_content.encode('utf-8'))

        # Build Docker image
        client = docker.from_env()
        print(client)
        image, build_log = client.images.build(fileobj=dockerfile_bytes, tag=name)
        print(image)

        # Authenticate with Azure and get the login server
        credential = DefaultAzureCredential()
        acr_client = ContainerRegistryManagementClient(credential, 'c1a9a576-d68f-433a-9756-56df5aced
        registry = acr_client.registries.get('majortest', 'idecctest')
        login_server = registry.login_server

        # Tag the Docker image with the login server
        tagged_image = f'{login_server}/{name}:latest'
        client.images.get(name).tag(tagged_image)

        # Push the Docker image to Azure Container Registry
```

Figure 3.7: Creating Virtual Image in Azure using Django

3.5 KEY CHALLENGES

The major issues we faced during the development process include the following:

- Starting on the project was a big hurdle because we found it very hard to point ourselves in the right direction.

- Creating the Docker file was a little bit challenging to implement at first.

- Configuring Django to Azure was quite hard and challenging as Azure subscription was throwing error.

CHAPTER 4: TESTING

4.1 TESTING STRATEGY

The IDE on Cloud will be tested using a combination of manual and automated testing. Manual testing will be performed by small group of students to verify the functionality of the IDE on Cloud. Automated testing will be performed using a variety of tools, such as Selenium and pytest, django Tests.

Manual Testing:

Manual testing will be performed on the following areas:

- User interface
- Functionality
- Performance
- Security

Automated Testing:

Automated testing will be performed on the following areas:

- User interface
- Functionality
- Performance

Automated testing tools will be used to verify the functionality of the IDE on Cloud and to identify any potential bugs.

The IDE on Cloud will be automatically tested using a variety of tools, such as:

- Selenium
- Unittest

Django's unit tests use the Python standard library module unittest. To define tests, this module uses a class-based methodology. This means that each test has its own class, which is inherited from the unittest.the TestCase class.

Before you can construct a unit test, you must first create a new class that derives from unittest. Study of a single case. The class's next step is to define one or more "test"-starting methods. Each test method should evaluate a separate aspect of your code.

For example, if you're developing a unit test for a function that adds two numbers, you could call it "test_add_numbers." This technique would require two parameters: the first being the number to be added, and the second being the second number. The procedure would compare the outcome to the projected value after adding the two integers. If the outcome matched the expected value, the test would pass. Otherwise, the test would fail.

After you've defined your test methods, you can execute them by invoking the unittest.main() function. This function will execute every test in the current file and will also print a report with the results. The report will indicate the number of tests done, the number of tests that passed, and the number of tests that failed.

Django also includes a variety of tools to help you write unit tests. The coverage module can be used to determine the test coverage of your code, and the unittest.mock module can be used to mock out dependencies in your tests.

Also, Azure provides Log Analytics and Monitoring data to the user or developer so that one can manage their resources effectively saving cost.

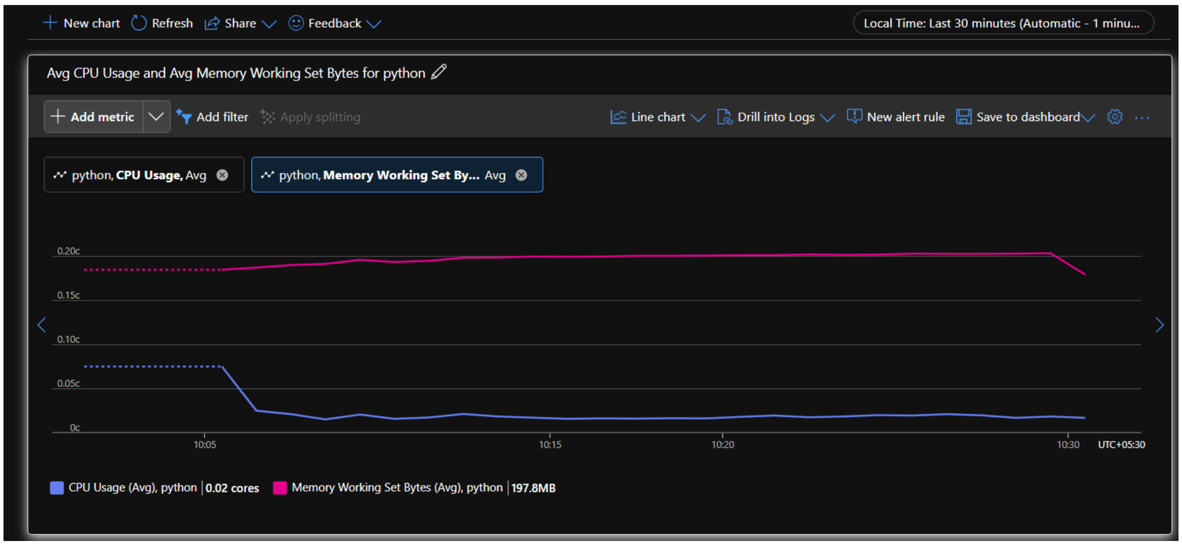


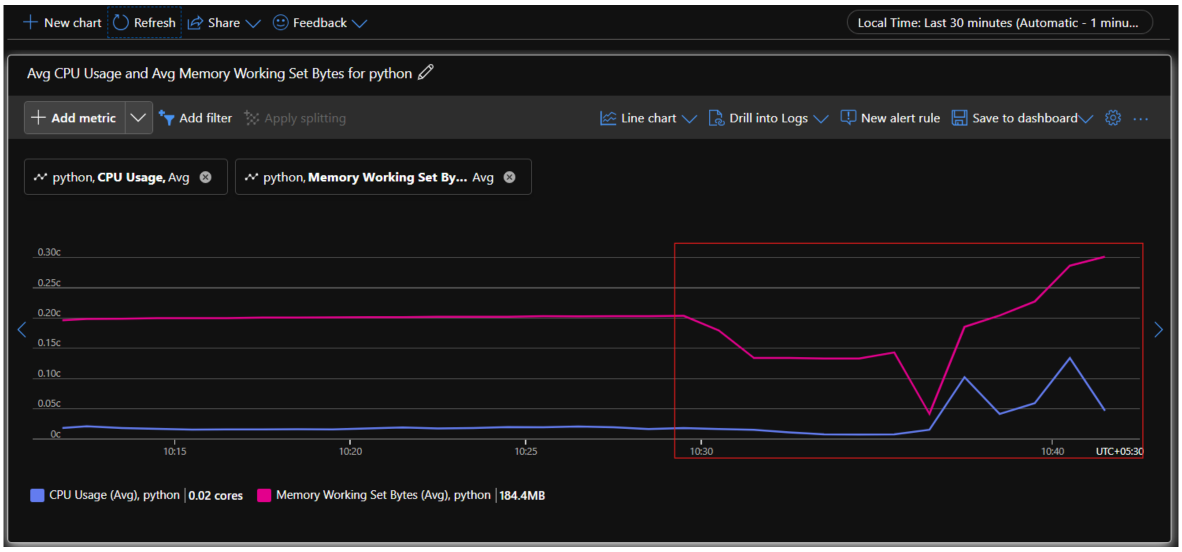
Figure 4.1: Python container metrics over a simple python code

To test how much load is being exerted on our docker container images servers we ran a python code of a countdown timer for 30 minutes and using the metrics analysis of Azure, we got the graph in *Figure 4.1* average CPU core usage and Working set of memory in bytes.

```
import time

def countdown(time_sec):
    while time_sec:
        mins, secs = divmod(time_sec, 60)
        timeformat = '{:02d}:{:02d}'.format(mins, secs)
        print(timeformat, end='\r')
        time.sleep(1)
        time_sec -= 1

    print("stop")

countdown(1800)
```

As we can see the maximum CPU usage was gone up to 0.08 cores when the server launched initially, after that the average was roughly around 0.03 cores and the memory being used is around 200 MB, note that 1 core of CPU and 1 GB memory was provided showing that a lot of memory and CPU is underutilized meaning we can save resources by assigning even less resources to simple tasks.

Next, we wanted to run the same python container under same configuration a Machine Learning code and test the strain on the system once again.



New chart
Refresh
Share
Feedback
Local Time: Last 30 minutes (Automatic - 1 minu...
Avg CPU Usage and Avg Memory Working Set Bytes for python
Add metric
Add filter
Apply splitting
Line chart
Drill into Logs
New alert rule
Save to dashboard
python, CPU Usage, Avg
python, Memory Working Set By... Avg
0.30c
0.25c
0.20c
0.15c
0.10c
0.05c
0c
10:15
10:20
10:25
10:30
10:40
UTC+05:30
CPU Usage (Avg), python 0.02 cores
Memory Working Set Bytes (Avg), python 184.4MB

Figure 4.2: Python container metrics over a small ML code

As we can see in *Figure 4.2* the average CPU utilization increased and also the average memory has jumped from 200 MB to 300 MB, we used numpy library in this and made it multiply two random 1000 row matrices over themselves.

```python
import numpy as np

def main():
    matrix_size = 1000

    matrix_a = np.random.rand(matrix_size, matrix_size)
    matrix_b = np.random.rand(matrix_size, matrix_size)

    result = np.dot(matrix_a, matrix_b)

if __name__ == "__main__":
    main()
```

From these two experiments or trials run we can say that the CPU utilization depends directly on the type of code we are running, but even so the maximum utilization of CPU doesn’t reach 1 core or even 0.5 core hence we can save our billing and resources by providing less resources to the machine, and depending on heavy usage the user may select different resource or the system will scale itself.

CHAPTER 5: RESULTS AND EVALUATION

5.1 RESULTS

5.1.1 ANALYSIS

Cloud based IDE removes all the headaches of the developer to setup their own personal environment in their local system as IDECC will make it for them within few minutes, while doing the same in their own system will take up a lot of time depending on the complexities and the dependencies of different languages or frameworks.

Cloud Based IDE saves a lot of resources and cost for the user, as all of the computation power is being used in the Cloud server. This feature of cloud computing removes the need of personal computer with high configuration.

Cloud Based IDE also enables the idea of working collaboratively for a project team or developer team. Many projects require collaborative effort of the team to build the same application, working collaboratively in Cloud IDE will allow the same team members to work on the same code sitting in different places, connected through the internet.

The main aim of our project is to let the developers learn, make projects, collaborate with each other without having the need to install any IDE or packages on the computer being used by the user, provided that the developer is connected to the internet.

5.1.2 DEFINING THE NEW APPROACH

The limitations imposed by the normal Integrated Development Environments, which are installed in computer as already discussed are very constricting and the user may not feel liberated while using them, if all the same features that are provided by a classic IDE are provided in the cloud without having to install the dependencies related to language and not having to keep care of the environment variables will increase the productivity of the users and improve the development lifestyle of the community as a whole. The introduction of Cloud IDE aims to increase the collaboration that is seen between the coders or developers. Increasing the collaboration level between team members will surely increase the production level and the quality of the application being developed by the team.

Docker is helping us in making containers of the Virtual Machines and then to spin them up on demand by the user. As more and more languages and frameworks will be developed our aim will be to increase the languages on the IDE as well. Container images are stored in Azure Container Registry.

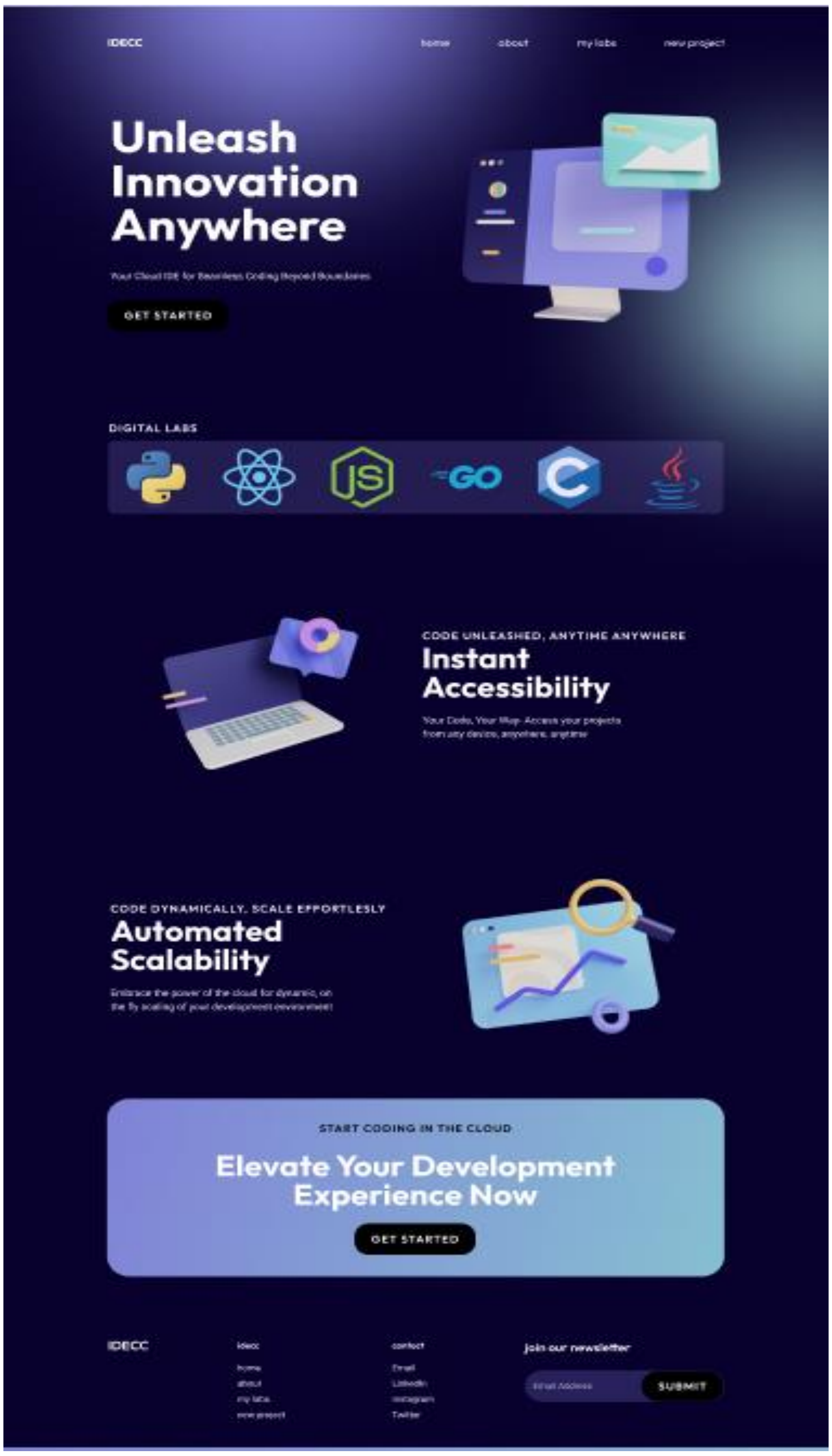
Pre-Configured Environments tailored for different programming languages and frameworks, eliminate the need for manual setup, Docker Container of the desired Programming Language will contact the Virtual Machine and a new Virtual Machine will start up and an IDE will be launched and then the user is ready to code. With the User-Friendly Interface of the Website and the Integrated Development Environment the user will find it very easy to navigate and use the services to its benefit.

With the introduction of Cloud Based IDE the collaboration between team members increases as there are no more compatibility issues with the system and the environment as well which creates a good coding environment within the team. With the advantages provided by cloud computing which include services like automatic scaling and pay as you go services will be very useful and functional for the project.

Ultimately all these will lead to increase in productivity and time saving of the team and developer as an individual.

5.1.3 HOME PAGE

This is the Home page of the website; the developer will land here after signing in to the website. This page gives the general overview of the application.


IDECC

home about my labs new project

Unleash Innovation Anywhere

Your Cloud IDE for Seamless Coding Beyond Boundaries

GET STARTED

DIGITAL LABS

CODE UNLEASHED, ANYTIME ANYWHERE

Instant Accessibility

Your Code, Your Way. Access your projects from any device, anywhere, anytime

CODE DYNAMICALLY, SCALE EFFORTLESSLY

Automated Scalability

Embrace the power of the cloud for dynamic, on the fly scaling of your development environment

START CODING IN THE CLOUD

Elevate Your Development Experience Now

GET STARTED

IDECC

idecc: home, about, my labs, new project

contact: Email, LinkedIn, Instagram, Twitter

join our newsletter

Email Address | SUBMIT

Figure: 5.1 Home Page of the Website

5.1.4 NEW PROJECT

New Project section is where the developer can start working on their new project, which can be selected by the shown cards of available languages or frameworks.

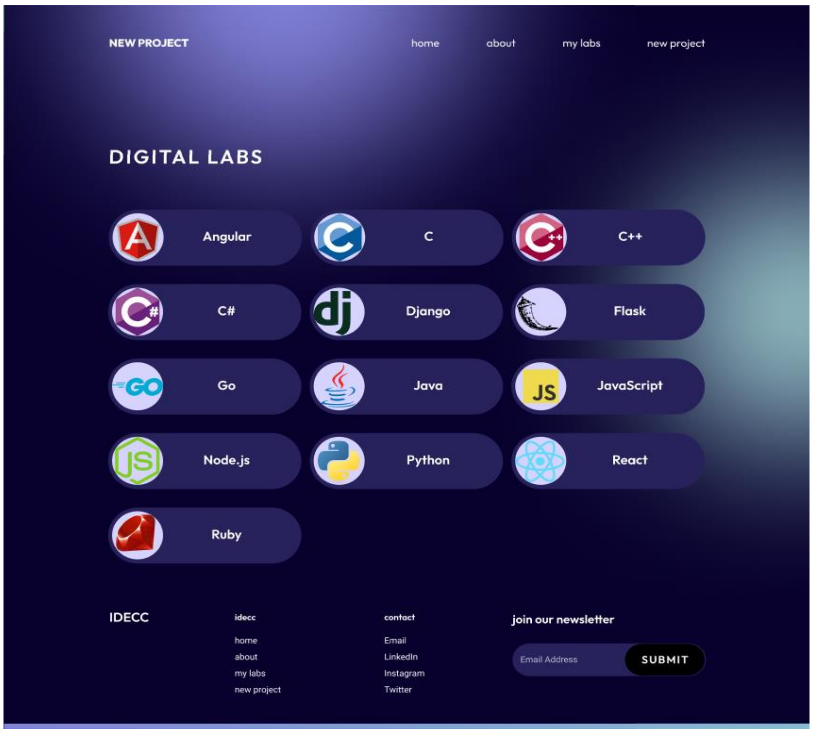


Figure 5.2: New Project section of the website

5.1.5 MY LABS

Projects that are developed by user are present in the My Labs section of the website along with their assigned name.

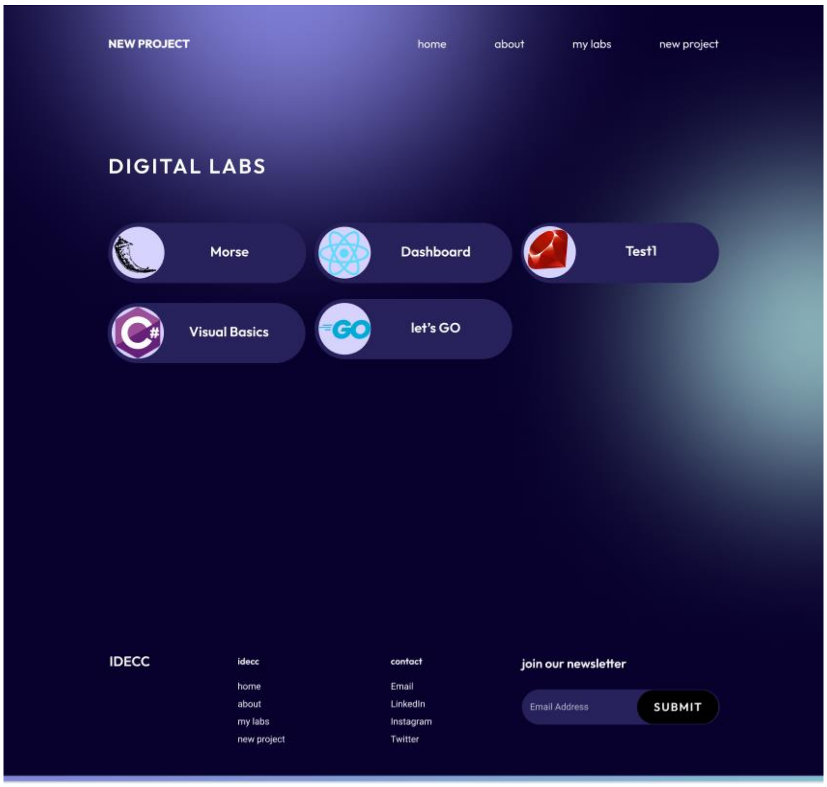
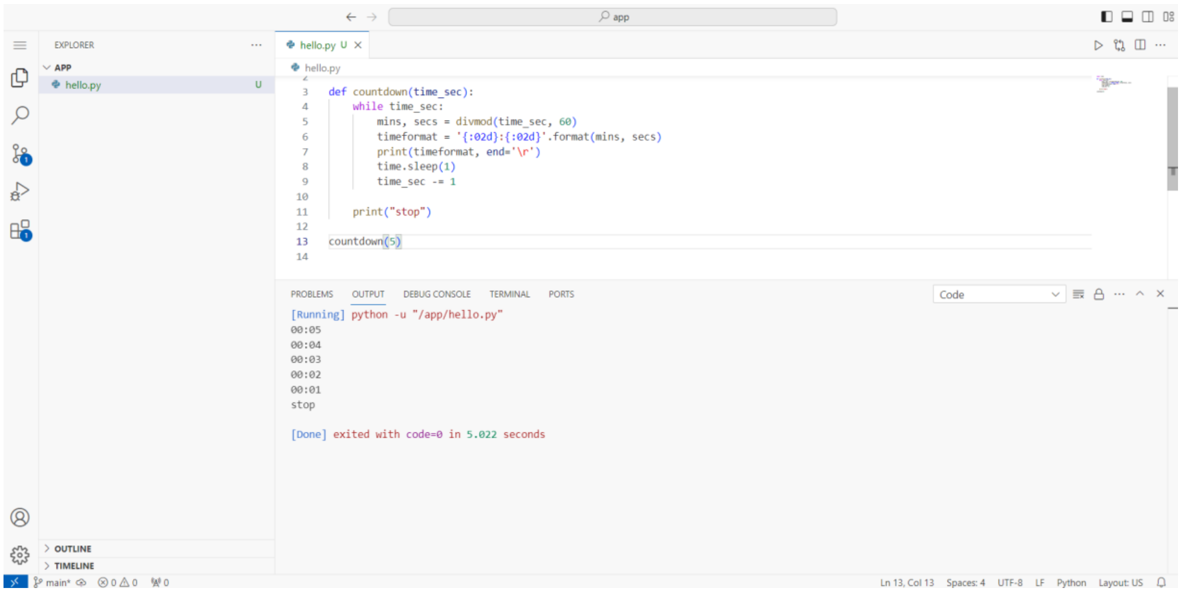


Figure 5.3: My Labs section of the Website

5.1.6 AZURE CONTAINER SERVICES

Azure Container Services are used to launch the IDE using Azure Container Registry to store the Docker image and Azure Container Apps to launch the application.



```python
def countdown(time_sec):
    while time_sec:
        mins, secs = divmod(time_sec, 60)
        timeformat = '{:02d}:{:02d}'.format(mins, secs)
        print(timeformat, end='\r')
        time.sleep(1)
        time_sec -= 1

    print("stop")

countdown(5)
```

```
[Running] python -u "/app/hello.py"
00:05
00:04
00:03
00:02
00:01
stop

[Done] exited with code=0 in 5.022 seconds
```

Figure 5.4: Hosted Application on URL

5.2 COMPARISON WITH EXISTING SOLUTIONS

The existing solutions with which we have compared our project are

1. codedamn [24]
2. PW skills [25]
3. Local IDE

These are the two already existing cloud-based IDE providers with which we can compare our project, both websites have considerable user flow as well so we know that we are not comparing our website with stagnant IDEs.

Codedamn rather than being a purely Browser based IDE is more of a full website providing every kind of thing a learning developer may need, codedamn provides user with courses on different programming languages then teach them in its own IDE, it has a section fully dedicated to problem solving which has a leaderboard of its own, a projects section where you can apply to do a project with the help of different languages, so codedamn rather than being a dedicated IDE is a lot more than that, but the basic concept behind codedamn's IDE lies in the frontend calling the server to run up the Containers which then call the Virtual Machines and the system is then spun up for the user.

PW Skills is the inspiration of our design and the initial thought of browser IDE. PW Skills Lab is an only IDE website, it uses vscode's IDE outline to showcase their containers, a list of programmable languages and frameworks is present and when selected it asks for a name for the project, after that it takes some time to spin up the development environment and then it reaches the IDE which is an exact copy of vscode's [26] PC IDE. You can code in this Lab and then when you close the IDE/Browser an image is created of the latest modification you have done on the website which then tore it in storage, creates a new card in my labs area of the PW Skills and then closes the virtual machine running for that particular project.

Local IDEs will be too much of a hassle to use once any developer will have the experience of browser-based IDE, setting up a development environment, installing dependencies and what not.

Comparing our project with codedamn's playground and PW Skills Labs still a lot of work has to be done, but our project seems to be headed in the right direction. Making images from containerised Dockers is the main step and setting up of Virtual Machines in the Cloud Infrastructure.

CHAPTER 6: CONCLUSIONS AND FUTURE SCOPE

6.1 CONCLUSION

6.1.1 PROGRESS OVERVIEW

The necessary components of the project that are needed to make this browser-based IDE are complete, the understanding of Docker containers has been achieved, images containing pre-configured development environments will be containerised in Docker and will be used every time to run up environments in the IDE in browser.

We have used Azure as our cloud provider as it has Azure Container Services which include Azure Container Registry to store the Docker container images and then can spin up the machine using Azure Container Apps.

6.1.2 MILESTONES ATTAINED

- Completion of the user interface design.
- Proper understanding of browser-based IDE
- Use case of Docker and containers
- Azure services to be used while deployment

6.1.3 CHALLENGES ENCOUNTERED

The challenges encountered during the learning of the project were mainly lack of resources, there are not a lot of things available online about browser-based IDEs like videos and/or articles, some research papers are available but they seem to have lack of information on things which are important in the making of cloud IDE or are just incomplete.

PW Skills and codedamn did give us some idea of the required things but still a lot of things were unanswered, about the in depth working of Docker and containers.

Codedamn gave us a good overview of the internal working of the servers and how the connections are made.

6.1.4 SUMMARY

We have integrated the vscode in web browser using the Azure Container Registry and Azure Container Apps. The backend server is connected to the frontend website using Django and React Framework. The login page and sign-up page are connected to the backend database server which will hold the user data and map the container images to correct user. The browser-based vscode is fully collaborative, and the code can also be pushed outside from the system for example github as it also has internet access. The developer will have full access to the Ubuntu system in the server, and they can utilize the server however they want.

6.2 FUTURE SCOPE

The continuation of this project will be to integrate it with other languages and frameworks, providing more options and variety for the user to choose from.

Docker images of all the new frameworks or languages will also be stored in the Azure Container Registry.

Future scope includes making the website more user friendly and responsive. Improving the login and sign-up pages is also one of our future considerations.

One more thing to consider in our future endeavours is that the creation of a new virtual machine takes time which can increase the time taken for the user to wait by a lot, so a way to decrease the waiting time wither by creating the virtual machine in advance or by some other measure is important to improve the user satisfaction.

Script to automatically scale the virtual machines up or down depending on the use case is to be made so as to avoid the extra billing cost on the admin end, and the script to shut down the idle virtual machines after taking their docker image and storing them with respect to each user in the cloud storage.

REFERENCES

[1] Desai, V. P., et al. "Microsoft azure: Cloud platform for application service deployment." Int. J. Sci. Res. in Multidisciplinary Studies Vol 7.10 (2021).

[2] Kedambadi Shreekar, Shryni. Automated Dockerization of Python based Web Apps. Diss. Dublin, National College of Ireland, 2020.

[3] Shah, Jay, and Dushyant Dubaria. "Building modern clouds: using docker, kubernetes & Google cloud platform." 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC). IEEE, 2019.

[4] Díaz Alonso, Rubén Cayetano. "Software containerization with Docker." (2017).

[5] Copeland, Marshall, et al. Microsoft Azure: planning, deploying, and managing your data center in the cloud. New York, NY: Apress, 2015.

[6] Merkel, Dirk. "Docker: lightweight linux containers for consistent development and deployment." Linux j 239.2 (2014): 2.

[7] Gadhikar, Lakshmi M., et al. "Browser based IDE to code in the cloud." New Paradigms in Internet Computing (2013): 59-69.

[8] Goldman, Max, Greg Little, and Robert C. Miller. "Real-time collaborative coding in a web IDE." Proceedings of the 24th annual ACM symposium on User interface software and technology. 2011.

[9] Cała, Jacek, and Paul Watson. "Automatic software deployment in the azure cloud." IFIP International Conference on Distributed Applications and Interoperable Systems. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010.

[10] van Deursen, Arie, et al. "Adinda: a knowledgeable, browser-based IDE." Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 2. 2010.

[11] Persson, Patrik, and Görel Hedin. "An interactive environment for real-time software development." Proceedings 33rd International Conference on Technology of Object-Oriented Languages and Systems TOOLS 33. IEEE, 2000.

[12]https://learn.microsoft.com/en-us/azure/architecture/web-apps/app-service-environment/architectures/ase-standard-deployment [date of access:25-10-2023]

[13] https://learn.microsoft.com/en-us/azure/architecture/hybrid/hybrid-containers [date of access:28-10-2023]

[14] https://azure.microsoft.com/en-in/products/devops/pipelines [date of access: 28-10-2023]

[15] https://learn.microsoft.com/en-us/archive/msdn-magazine/2019/august/azure-devops-introducing-azure-deployment-manager [date of access: 28-10-2023]

[16] https://docs.citrix.com/en-us/tech-zone/design/design-decisions/azure-instance-scalability.html [date of access: 25-10-2023]

[17] https://azure.microsoft.com/en-in/pricing/purchase-options/pay-as-you-go [date of access: 31-10-2023]

[18]https://learn.microsoft.com/en-us/azure/architecture/example-scenario/infrastructure/wordpress-iaas [date of access: 08-09-2023]

[19] https://www.python.org [date of access: 08-09-2023]

[20] https://www.djangoproject.com [date of access: 12-09-2023]

[21] https://www.javascript.com [date of access: 13-09-2023]

[22] https://react.dev [date of access: 13-09-2023]

[23] https://www.docker.com [date of access: 17-09-2023]

[24] https://www.codedamn.com [date of access: 15-08-2023]

[25] https://lab.pwskills.com [date of access: 15-08-2023]

[26] https://code.visualstudio.com [date of access: 17-08-2023]

major project

ORIGINALITY REPORT

3%	2%	1%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.coursehero.com Internet Source	<1%
2	open-innovation-projects.org Internet Source	<1%
3	vdoc.pub Internet Source	<1%
4	researchr.org Internet Source	<1%
5	packages.nuget.org Internet Source	<1%
6	vocal.media Internet Source	<1%
7	Jacek Cała. "Automatic Software Deployment in the Azure Cloud", Lecture Notes in Computer Science, 2010 Publication	<1%
8	Microsoft Azure, 2015. Publication	<1%
	aaltodoc.aalto.fi	

9	Internet Source	<1 %
10	Submitted to Georgia State University Student Paper	<1 %
11	www.geeksforgeeks.org Internet Source	<1 %
12	Submitted to University of Greenwich Student Paper	<1 %
13	Brian L. Gorman. "Chapter 5 Azure Container Ecosystem: Azure Container Registry, Azure Container Instances, and Azure Container Apps", Springer Science and Business Media LLC, 2023 Publication	<1 %
14	Submitted to University of Westminster Student Paper	<1 %
15	ntnuopen.ntnu.no Internet Source	<1 %
16	Submitted to Australian School of Accounting Student Paper	<1 %
17	Submitted to CSU, San Jose State University Student Paper	<1 %
18	Submitted to Barnet and Southgate College Student Paper	<1 %
	Submitted to Florida Gulf Coast University	

19	Student Paper	<1 %
20	Submitted to The University of Memphis Student Paper	<1 %
21	Submitted to University of the Sunshine Coast Student Paper	<1 %
22	cyberpanel.net Internet Source	<1 %
23	www.frontiersin.org Internet Source	<1 %
24	www.inovex.de Internet Source	<1 %
25	www.opensourceforu.com Internet Source	<1 %
26	downloads.ixiacom.com Internet Source	<1 %
27	ijesc.org Internet Source	<1 %
28	researchrepository.wvu.edu Internet Source	<1 %
29	stormotion.io Internet Source	<1 %
30	www.mactores.com Internet Source	<1 %

31	Max Goldman. "Role-based interfaces for collaborative software development", Proceedings of the 24th annual ACM symposium adjunct on User interface software and technology - UIST '11 Adjunct, 2011 Publication	<1%
32	Minoru Uehara. "JavaScript Development Environment for Programming Education Using Smartphones", 2019 Seventh International Symposium on Computing and Networking Workshops (CANDARW), 2019 Publication	<1%
33	bournemouthtown.co.uk Internet Source	<1%
34	essay.utwente.nl Internet Source	<1%
35	fastercapital.com Internet Source	<1%
36	huggingface.co Internet Source	<1%
37	Ümit Demirbaga, Gagangeet Singh Aujla, Anish Jindal, Oğuzhan Kalyon. "Chapter 4 Cloud Computing for Big Data Analytics", Springer Science and Business Media LLC, 2024 Publication	<1%

38	Tran, Hai T., Hai H. Dang, Kha N. Do, Thu D. Tran, and Vu Nguyen. "An interactive Web-based IDE towards teaching and learning in programming courses", Proceedings of 2013 IEEE International Conference on Teaching Assessment and Learning for Engineering (TALE), 2013. Publication	<1 %
39	John Warsinkse. "CISSP: Certified Information Systems Security Professional", Wiley, 2019 Publication	<1 %
40	Submitted to QA Learning Student Paper	<1 %

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: ....

Type of Document (Tick): PhD Thesis | M.Tech Dissertation/ Report | B.Tech Project Report | Paper

Name: ______________________________ __**Department:** __________________ **Enrolment No** _________

Contact No. _________________________________**E-mail.** ____

Name of the Supervisor: ____

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): __________________________

____

____

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

Complete Thesis/Report Pages Detail:

- Total No. of Pages =
- Total No. of Preliminary pages =
- Total No. of pages accommodate bibliography/references =

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at ....(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor) **Signature of HOD**

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	Excluded	Similarity Index (%)	Generated Plagiarism Report Details (Title, Abstract & Chapters)	
	• All Preliminary Pages • Bibliography/Images/Quotes • 14 Words String		Word Counts	
Report Generated on			Character Counts	
		Submission ID	Total Pages Scanned	
			File Size	

Checked by
Name & Signature **Librarian**

....

Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com