A CLOUD-BASED JAVA COMPILER FOR SMART DEVICES

A thesis presented to the Department of

Computer Science

African University of Science and Technology

In Partial Fulfilment of the Requirements for the Degree of

Master of Science

Submitted by

Mohammed Tanko Yahaya

JUNE, 2016

A CLOUD-BASED JAVA COMPILER FOR SMART DEVICES

By

Mohammed Tanko Yahaya

A THESIS APPROVED BY THE COMPUTER SCIENCE DEPARTMENT

RECOMMENDED: ________________________________

Supervisor, Professor Mohamed Hamada

____

Head, Department of Computer Science

APPROVED: ________________________________

Chief Academic Officer

____

Date

Abstract

The Java programming language is widely used in industry and business. Therefore, academic institutions worldwide include Java learning as a basic part of their Computer Science and Engineering curricula. At the same time, smart devices have become popular among university learners. This research tries to take advantage of this fact to promote Java learning. The main problem is that we cannot compile Java programs on smart devices due to the technical limitations of such devices. This research aims to leverage cloud computing, the availability, prevalence and affordability of smart devices and the ever-growing market of Android devices to provide users with text editors to create and modify Java programs and save them to a server. Users can also compile and execute created programs. A web-based version of the application is also provided for users who do not use Android devices that can be accessed via a browser on a PC or Smart device. The system uses an existing online compiler. The developed cloud-based compiler can be integrated into a smart multimedia learning system for learning the Java programming language.

Acknowledgement

Firstly I thank Almighty Allah, to whom all praise is due, the most beneficent, the most merciful, and the lord of the worlds for sparing my life and enabling me complete this Master of Science degree program in good health. You make my impossible possible, you make me determined, I lay under your supportive umbrella I cannot pay for but rather I do what you command me to do.

To my father, Alhaji Tanko Yahaya, my mother, Hajiya Mariam Mohammed, my brothers and sisters, I appreciate your prayers, advices, financial and mental support before and during the course of my program. I pray Allah rewards you all abundantly.

I thank the Nelson Mandela Institution (NMI) considering me worthy of a scholarship for my Master of Science degree program out of many other qualified candidates who merited this award. Your faith in me has and will continue to yield the expected dividends.

I would like to express my deepest appreciation to my supervisor Professor Mohamed Hamada whom without his guidance, encouragement and persistence this thesis would not have been possible. He continually and convincingly a spirit of adventure in regard to research and also find enough time to give me detailed explanations, suggestion and directions. Thank you so much sir.

My profound gratitude goes to all the faculties in Computer Science and Engineering department especially Professor Mamadou Kaba Traore, Professor Lehel Csato, Professor Ben Abdallah and Professor Mohamed Hamada who happens to be my supervisor. I appreciate your constructive criticisms and the knowledge you impacted on me.

I wish to thank my friends namely Mohammed Audu Galadima, Ibrahim Bright Mohammed, Salaudeen Rabiu Niyi, Habib Itopa Mohammed, for their help, words of encouragement and prayers throughout the period of my program.

To the entire management and staffs of African University of Science and Technology, Galadimawa Abuja Nigeria, I say a very big thank you for providing a conducive learning and research environment for me and my colleagues to run our Master of Science program.

Of course I would like to thank my course mates, PhD students in the Department of Computer Science and Engineering and the entire students of the African University of Science and Technology Galadimawa, Abuja, Nigeria. You all made my stay here one I will always remember.

Dedication

This thesis is dedicated to my beloved parents Alhaji Tanko Yahaya and Hajiya Maryam Mohammed.

Table of Contents

List of figures

List of tables

List of abbreviations

AJAX	Asynchronous Javascript and XML
AMD	Asynchronous Module Definition
API	Application Programming Interface
BSD	Berkley Software Distribution
CARET	Center for Applied Research in Educational Technology
COBRA	Common Object Request Broker Architecture
CRM	Customer Relationship Management
CRUD	Create, Read, Update and Delete
CSS	Cascading Style Sheets
DVM	Dalvik Virtual Machine
HTML	HyperText Markup Language
HTTPS	HyperText Transfer Protocol
IaaS	Infrastructure as a service
IDC	International Data Corporation
IDE	Integrated Development Environment
JVM	Java Virtual Machine
LAN	Local Area Network
LGPL	Lesser General Public License
MIT	Massachusetts Institute of Technology
MOOC	Massive Open Online Courses
NMI	Nelson Mandela Institution
OHA	Open Handset Alliance
PaaS	Platform as a Service
PC	Personal Computer
PDA	Personal Digital Assistant
PHP	HyperText Preprocessor
RAM	Random Access Memory
REST	Representational State Transfer

RPC	Remote Procedure Call
SaaS	Software as a Service
SDK	Software Development Kit
SOAP	Simple Object Access Protocol
URL	Uniform Resource Locator
WYSIWYG	What You See is What You Get

CHAPTER ONE: INTRODUCTION

In this chapter we examine learning technologies in the 21st century and how they affect learning among students. We also discuss mobile technology, mobile learning and multimedia learning systems. The chapter also discusses the increased use of smart mobile devices, the future of learning technologies and how we can leverage them to improve the use of mobile learning. These discussions lead us to formulate and present our problem statement as well as our aims and objectives.

1.0 Learning technology in the 21st century

Technology is found in most parts of our lives. Most homes have connected computers or internet-enabled devices (smart devices). Technology plays a very important role in 21st century education. When integrated into the curriculum, technology revolutionizes the learning process. More and more studies show that technology integration in the curriculum improves students' learning processes and outcomes. Teachers who recognize computers as problem-solving tools change the way they teach. They move from a behavioural approach to a more constructivist approach.

Technology and interactive multimedia are conducive to project-based learning. Students are engaged in their learning using these powerful tools, and can become creators and critics instead of just consumers. Technology helps change student/teacher roles and relationships: students take responsibility for their learning outcomes, while teachers become guides and facilitators. Technology lends itself as the multidimensional tool that assists that process. For economically disadvantaged students, the school may be the only place where they will have the opportunity to use a computer and integrate technology into their learning. There is a growing body of evidence that technology integration positively affects student achievement and academic performance. The Center for Applied Research in Educational Technology (CARET) found that, when used in collaborative learning methods and leadership aimed at improving the school through technology planning, technology impacts achievement in content area learning, promotes higher-order thinking and problem-solving skills, and prepares students for the workforce [5].

1.0.1 Mobile technology and mobile learning

The availability, prevalence and affordability of mobile phones, tablets and other connected devices have been on the increase over the last two decades. With this, wireless technology can dramatically improve learning and bring digital content to students. Students engage with mobile technology and use it regularly in their personal lives for taking pictures, playing games, using social media and so on. Effective use of mobile technology is less about the tools and more about learners' digital literacy skills, including the ability to access, manage and evaluate digital resources [7].

Berking et al. in [8] define mobile learning as "leveraging ubiquitous mobile technology for the adoption of or augmentation of knowledge, behaviors, or skills through education, training, or performance support while mobility of the learner may be independent of time, location, and space".

Mobile learning according to the author in [6], "encompasses the usage of portable computing devices (such as iPads, laptops, tablet PCs, PDAs, smart phones) with wireless networks to enable mobility, mobile learning, allowing teaching and learning to extend to spaces beyond the traditional classroom".

As an integral part of students' daily lives, mobile technology has changed how they communicate, gather information, allocate time and attention and potentially how they learn. The mobile platform's unique capabilities, including connectivity, cameras, sensors and GPS have great potential to enrich the academic experience of learners. This is according to the result of a research carried out by [7].

Global mobile device ownership is high and continues to increase. The figures below give the growth rate of mobile devices and an estimated number of users among different age groups.

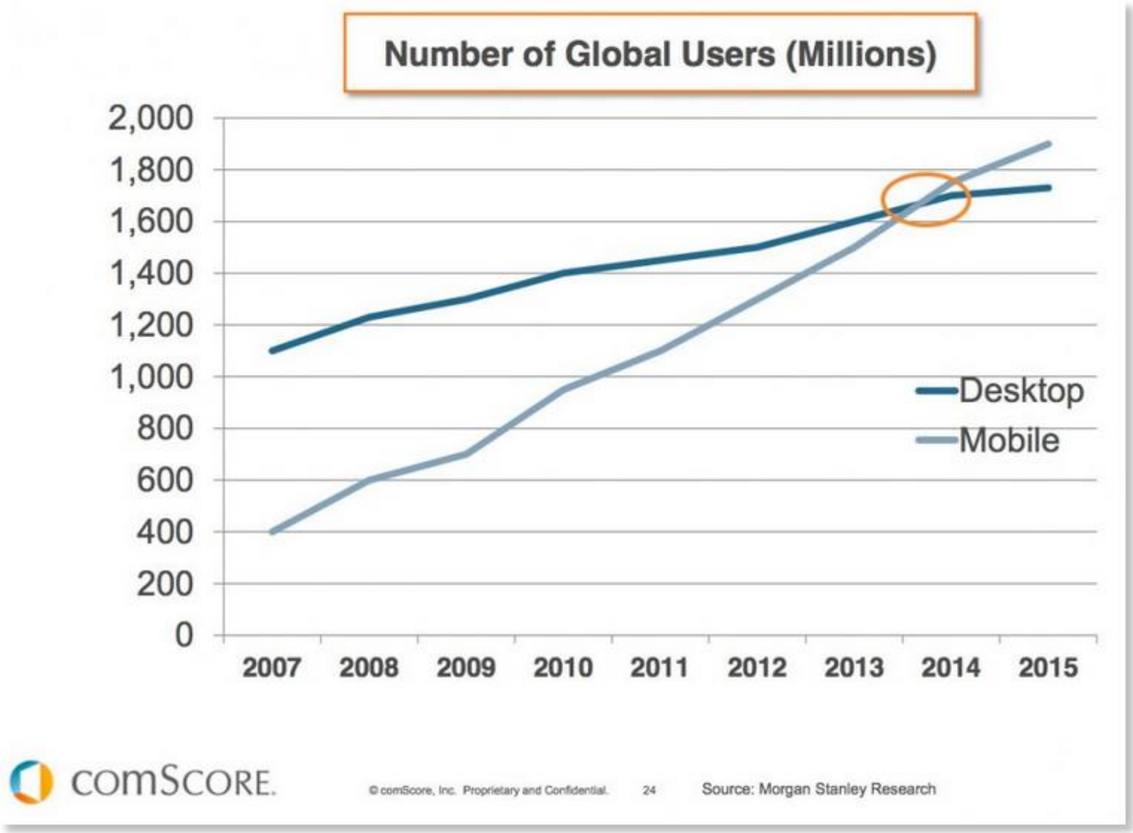
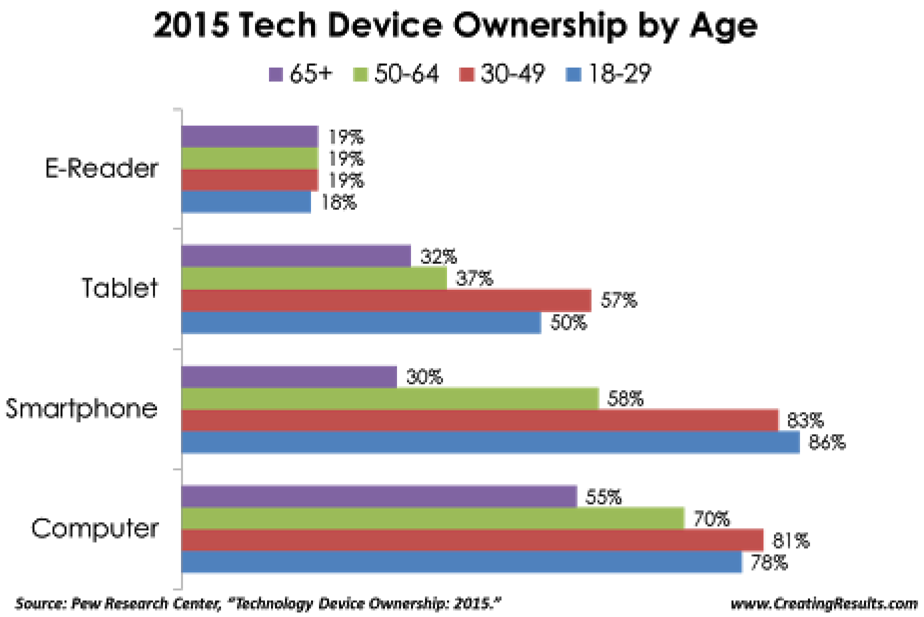


Figure 1.1: Global desktop and mobile device ownership



Source: Pew Research Center, "Technology Device Ownership: 2015." www.CreatingResults.com

Figure 1.2: Global device ownership by age

1.0.2 Smartphone OS market share

In the year 2015, the worldwide smartphone market grew by 13.0% year on year with 341.5 million shipments, according to data from the International Data Corporation (IDC) Worldwide Quarterly Mobile Phone Tracker. In the past few years, the Android operating system has begun to dominate the world of smart devices – phones and tablets [12]. The figure below shows the percentage of smart devices that run on the various operating systems, obtained in the year 2015 according to Forbes.

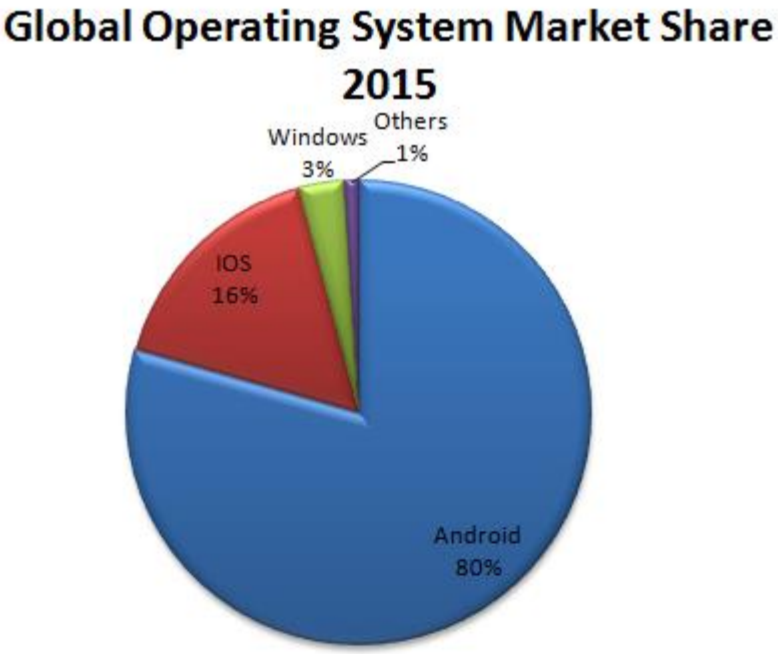


Figure 1.3: Percentage of smart device OS users in 2012

Source: Trefis Team, 2015

As at the first quarter of the year 2015, Android dominated the smartphone market with a share of 82.8% [12]. The Android OS market share has been on the rise over in the past few years and is projected to keep rising.

1.0.3 The future of learning technology

The learning and development world has evolved rapidly during the last four to five years. There are many online learning platforms such as Khan Academy (www.khanacademy.com), Cousera (www.cousera.org) that offer video-based learning and Massive Open Online Courses (MOOCs). Then there is learning delivered through smartphones, tablets and cloud-based applications, not to mention the increase in webinars, podcasts and social media-based offerings across the digital world.

Technological advances, internet access and mobile device usage all play a significant role in enabling scalable access of learning across the globe. However, while organizations have started investing in these areas, there is still a low adoption of e-learning [9].

With the rapid growth of internet-enabled devices, it is important to note the future of learning technology is very bright. By developing more e-learning platforms in addition to the existing ones and motivating both learners and trainees to adopt these platforms, the many benefits associated with learning technologies can be achieved.

1.1 Multimedia learning systems

Mayer and Moreno define multimedia as “any computer-mediated software or interactive application that integrates text, color, graphical images, animation, audio sound, and full motion video in a single application”. Multimedia learning systems consist of animation and narration, which offer potential avenues for improving student understanding [29].

With the rapid advances the internet and information and communications technology have made, it has become extremely important to utilize these technologies to enhance teaching, learning and education. This will be of immense value to both the educator and the learner. For educators, it makes available a convenient platform to present information in interactive and media-enhanced formats in contrast to the usual methods they are used to. For learners, information offered through these channels and methods is easier to understand, deal with and retrieve, and this in turn improves the whole learning process for all parties.

Multimedia has the potential to create high-quality learning environments, with the ability to create a more realistic learning context through different media. It also helps by allowing a teacher to take better control of the classroom, especially when the class size is large.

1. 2 Java programming language

Java is a programming language first developed by James Gosling at Sun Microsystems, which is now a part of Oracle Corporation. It was released in 1995 as a part of Sun Microsystems' Java platform. Much of its syntax is derived from C and C++. Java applications are usually compiled to bytecode (a class file) that can run on any Java Virtual Machine (JVM). Java is currently one of the most popular programming languages in use [10].

1. 3 Problem statement

In view of the important role learning technology plays in education in the 21st century coupled with the relative affordability and affordability of internet-enabled smart devices, it has become imperative to develop a smart multimedia learning system for the Java programming language which comprises a lecture slides module, a reading topics module, a flash cards module, video lectures module, and an integrated development environment (IDE) module for learning, compiling and running Java programs on learners' smart devices.

This research focuses on the development of a cloud-based compiler for smart devices to compile and run Java programs on learners' smart devices which would be incorporated into a smart multimedia learning system for the Java programming language, together with other components in the future.

1.4 Purpose of the research

The research aims to achieve the following:

a. Design and develop an IDE for the Java language to run on smart devices on a chosen operating system.
b. Implement a cloud-based server that hosts the Java compiler.

1.5 Target operating system

Android is an open-source operating system introduced by Google and the OHA (Open Handset Alliance) in September 2008. It is the most widely used mobile operating system, with more than 80% of the world's mobile operating systems being Android. The popularity of the Android OS, its affordability and readily available programming kits makes it an easy choice for this research. Although the application to be developed is targeted for the Android OS, a web-based version is also to be developed for non-Android users.

1.6 Expected results and deliveries

The following are the expected deliverables of this research

1. An Android application via which a user can type a program in Java and press a compile button to send the written code to the cloud server for processing.
2. A web-based version of the developed Android application

1.7 Scope of the work

This research entails the development of a mobile application to compile and run Java on smart devices. A web-based version of the application was also built for those who do not use the target operating system the application was built on. The server side of the project could not be completed in time and as a result an existing online compiler was used to achieve the results.

1.8 Thesis structure

Chapter 2 gives an insight into concepts relating to compilers, cloud-based computing, the Android operating system and a review of current literature.

Chapter 3 discusses the research methodologies, architecture and describes the system.

Chapter 4 gives a detailed discussion of the implementation of the system.

Chapter 5 rounds up by discussing results, summary, conclusions and suggestions for future work.

CHAPTER TWO: LITERATURE REVIEW

2.0 Introduction

There are certain concepts relating to cloud-based compilers for smart devices. This chapter discusses these concepts which include compilers and how they work, cloud computing and the Android operating system.

This chapter also reviews existing work relating to cloud-based compilers, their weaknesses and how the system to be developed intends to solve these weaknesses.

2.1 Compilers

Computer programs are formulated in a programming language and specify classes of computing processes. Computers, however, interpret sequences of particular instructions, but not program texts. Therefore, the program text must be translated into a suitable instruction sequence before it can be processed by a computer. This translation can be automated, which implies that it can be formulated as a program itself. The translation program is called a *compiler*, and the text to be translated is called *source text* (or sometimes *source code*) [13].

2.1.1 Compiler architecture

Compilers are used to compile programs and convert them from written program to executable binaries. In other words, a compiler is a program that reads a program written in one language and translates it into another language. The compiler creates executable files which can then be run in order to execute the program and its instructions.

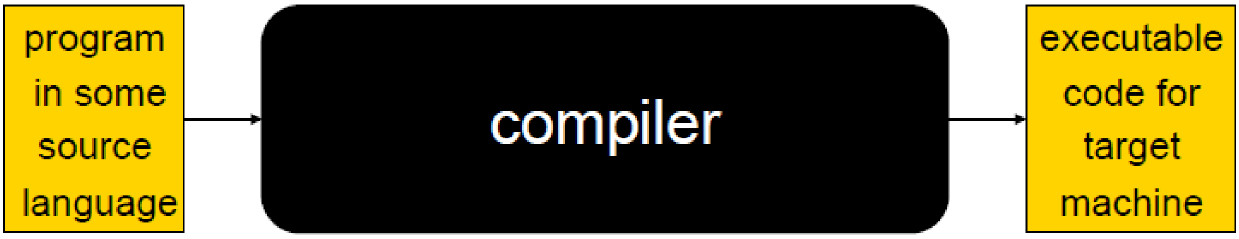

program in some source language
compiler
executable code for target machine

Figure 2.1: How a compiler works

Every compiler primarily consists of three parts:

1. **The Front end:** This checks the semantics and syntax of the higher level code (written by the user). Other functions like type checking and error reporting are also performed by the front end.
2. **The Middle end:** This performs the optimization through removal of redundant code, or relocation of computation depending on the context.
3. **The Back end:** This is the part where the translation of the language actually takes place.

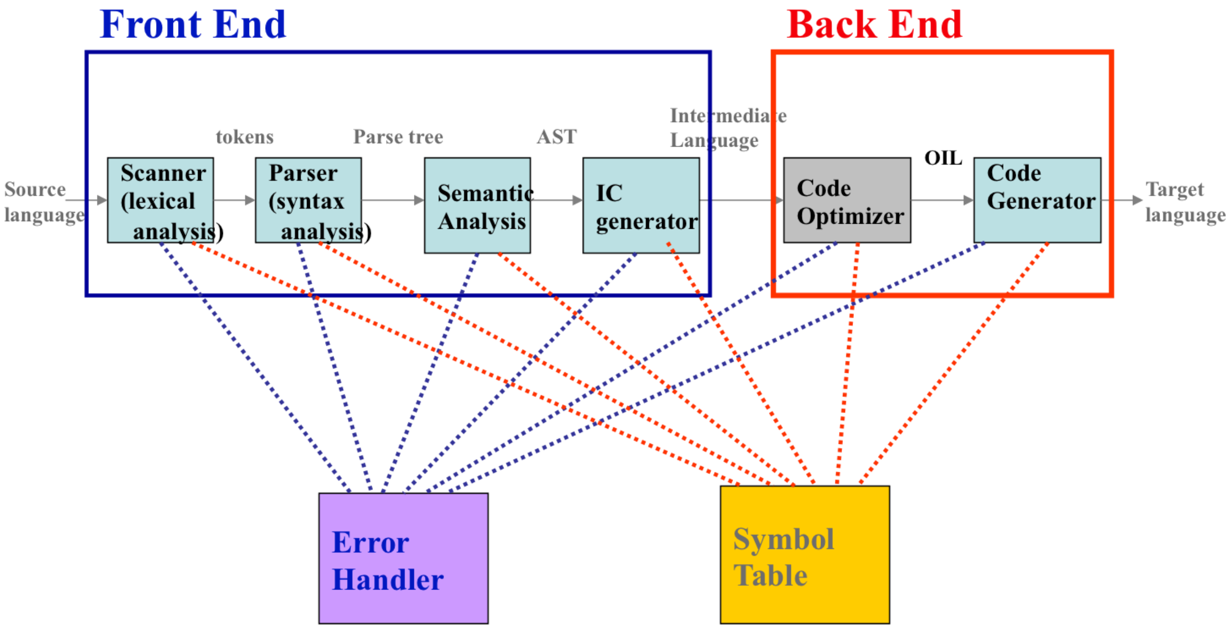


Figure 2.2: Architecture of a compiler

2.1.2 Phases of a compiler

The compiler has a number of phases plus a symbol table manager and error handler. This modularization is typical of many real compilers. The authors in [11] and [14] describe the phases of a compiler which are summarized in the table below.

Table 2.1: Compiler phases and their description

S/N	PHASE	DESCRIPTION
1.	Lexical Analyzer	Break the source file into individual words, or *tokens.*
2.	Syntax Analyzer	Analyze the phrase structure of the program.
3.	Semantics Analyzer	Build a piece of *abstract syntax tree* corresponding to each phrase. Determine what each phrase means, relate uses of variables to their definitions, check types of expressions, and request translation of each phrase.
4.	Intermediate Code Generator	Transforms parse tree into intermediate language which represents source code program
5.	Code Optimizer	Optimizes intermediate codes and produces fast running machine codes
6.	Code Generator	Produces re-locatable machine codes or assembly codes
7.	Target Language	

2.2 Concept of cloud computing

Cloud computing according to the author in [15] refers to “flexible self-service, network-accessible computing resource pools that can be allocated to meet demand”. Services are flexible because the resources and processing power available to each can be adjusted “on the fly” to meet changes in need or based on configuration settings in an administrative interface, without the need for direct IT personnel involvement. These resources are assigned from a larger pool of available capacity (for example memory, storage, CPUs) as needed, allowing an organization to spin up a proof-of-concept application, expand that to a full prototype, and then roll it out for full use without having to consider whether existing hardware, data centre space, power and cooling are capable of handling the load. Cloud computing allows the allocation of resources to be adjusted as needed, creating a hardware-independent framework for future growth and development.

Almost anything can be hosted in the cloud, from databases and applications to complete virtual infrastructures encompassing data storage, networking and all components of the server environment. The cloud can also host virtualized user desktop environments available from any networked client device, whether or not the client has sufficient local resources to host the virtualized desktop environment and its various applications.

Cloud computing goes beyond simply hosting a website or database service on a machine located in a remote data centre, with early cloud services such as Google Gmail and Google Apps demonstrating the power of cloud computing, starting in 2006. Cloud computing solutions have several common characteristics, regardless of their form [15]. They include:

Managed by the provider – cloud computing services are managed by the cloud provider. Once applications and services have been moved to external cloud computing, an organization no longer needs to worry about local data centre issues regarding power, space and cooling, and developers need only know whether their applications will be running on one cloud service platform or another [15].

Flexible resource assignment – the capacity and resources available to cloud computing services can be increased or decreased, with costs adjusted according to actual consumption. This allows an organization to spin up a new offering with only minimal costs for the resources used and then to meet spikes or cyclic use patterns with increased capacity, paying for only the level of use needed. Traditional data centres must always plan for future growth, and a sudden success for a web-based offering can rapidly overrun available server and network capacity unless data centre managers purchase sufficient “spare” resources beforehand. Cloud computing draws resources from a pool as they are needed, based on the level of service consumption. This is similar to the way power companies supply power to individual organizations, billing each according to its individual use. For example, a new cloud application might experience a sudden increase in use following mention on a popular blog and require additional network bandwidth, data storage, server memory or CPU power to keep up with the sudden increase in demand. Traditional data centres would be limited by hardware constraints, while cloud computing alternatives can simply add CPUs or expand available database file storage up to predefined limits when needed and then shrink back after the storm of access has passed to manage on-demand costs [15].

Network accessible – cloud services are available via networked devices and technologies, facilitating rapid access by mobile customers and remote office locations. This provides an "anywhere, anytime" service model not possible in traditional data centres, where service downtime and local area outages in power and networking can impact uptime. Because cloud computing vendors can be located anywhere in the world, they can host organizational services from areas outside geopolitical turmoil or environmental threats. Before a hurricane, for example, a cloud service provider could transfer operations from Florida to Washington transparently to the service consumer [15].

Sustainable – because cloud providers can provide resources at need, it is possible to reduce power and cooling requirements during off-peak times, gaining economies of scale well beyond those available to single-tenanted hardware-based data services, which must stay on, waiting for later use. The flexibility in cloud hosting location allows providers to shift operations without disruption to consumers. They can move data centre activity seasonally to save on cooling costs or transfer operations to areas with excess power production capability, such as Iceland [15].

Managed through self-service on demand – after limits for resource availability are configured within the cloud provider’s systems, available resource capacity can be automatically expanded or managed by the client with minimal effort. Bringing up a test server no longer requires access to the physical system, loading software, and configuring networking by hand; instead, the customer needs only to access their cloud provider and request a new resource allocation using the self-service user interface. As long as the organization’s contractual limits on resources allow the addition, it is managed automatically without the need for further technical assistance. [15]

2.2.1 Types of cloud computing

Kirk Hausman et al. in [15] give a classification of cloud computing by location while a classification by type of services offered is given in [16]. The two classifications are discussed below.

2.2.1.1 Location of the cloud

Cloud computing is typically classified in the following three ways:

1. **Public cloud**: The computing infrastructure is hosted by the cloud vendor at the vendor’s premises. The customer has no knowledge of or control over where the computing infrastructure is hosted. The computing infrastructure is shared between many organizations.
2. **Private cloud**: The computing infrastructure is dedicated to a particular organization and not shared with other organizations. Some experts consider that private clouds are not real examples of cloud computing. Private clouds are more expensive and more secure when compared to public clouds.
3. **Hybrid cloud**: Organizations may host critical applications on private clouds and applications with relatively fewer security concerns than on the public cloud. The usage of both private and public clouds together is called a hybrid cloud. A related term is cloud bursting. In cloud bursting, organizations use their own computing

infrastructure for normal usage, but access the cloud using services like Salesforce cloud computing for high/peak load requirements. This ensures that a sudden increase in computing requirement is handled seamlessly.

4. **Community cloud:** This involves sharing of computing infrastructure in between organizations of the same community. For example, all government organizations within the state of California may share computing infrastructure on the cloud to manage data related to citizens residing in California.

2.2.1.2 Classification based upon service provided

Based upon the services offered, clouds are classified in the following ways:

1. **Infrastructure as a service (IaaS)** involves offering hardware-related services using the principles of cloud computing. These could include some kind of storage services (database or disk storage) or virtual servers. Leading vendors that provide IaaS are Amazon EC2, Amazon S3, Rackspace Cloud Servers and Flexiscale.
2. **Platform as a Service (PaaS)** involves offering a development platform in the cloud. Platforms provided by different vendors are usually not compatible. Typical players in PaaS are Google's Application Engine, Microsoft's Azure, Salesforce.com's force.com.
3. **Software as a Service (SaaS)** includes a complete software offering in the cloud. Users can access a software application hosted by the cloud vendor on pay-per-use basis. This is a well-established sector. The pioneer in this field has been Salesforce.com's offering in the online customer relationship management (CRM) space. Other examples are online email providers like Google's Gmail and Microsoft's Hotmail, Google docs and Microsoft's online version of office called BPOS (Business Productivity Online Standard Suite).

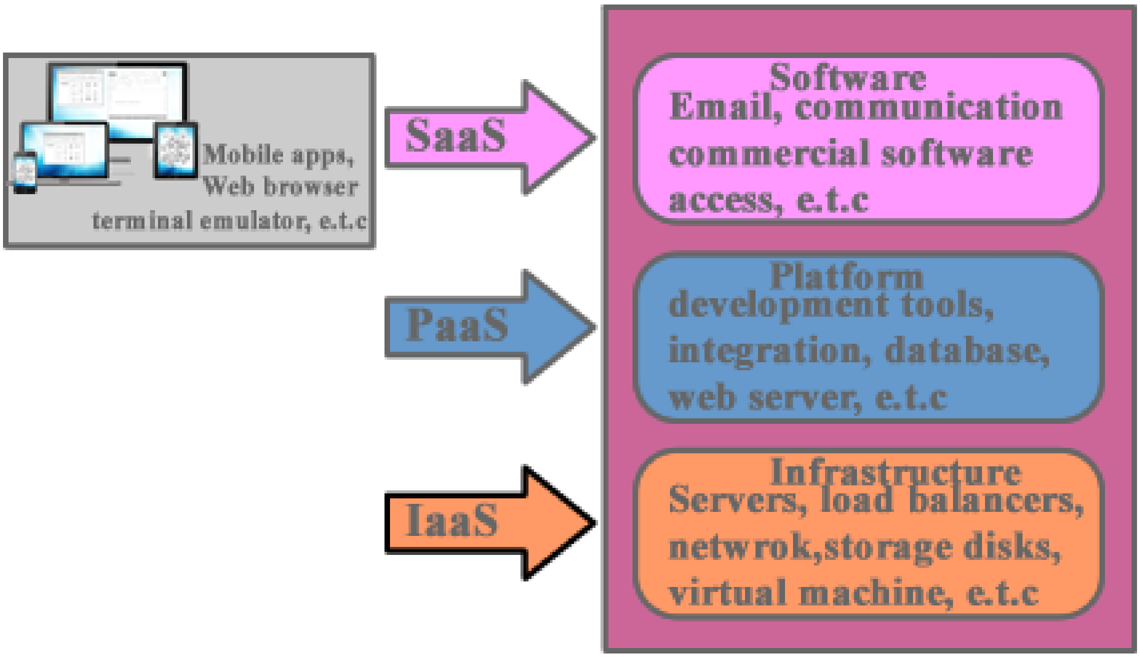


Figure 2.3: Cloud services

2.3 Android operating system

Android is a software stack for mobile devices that includes an operating system, middleware and key applications with the aim of all-time high performance by optimizing memory with faster and more accurate response. All applications are written using the Java language by enabling and simplifying the reuse of components, i.e. full access to the same framework.

Application programming interfaces (APIs) are used by the core applications and can be replaced or reused. They also include a set of C/C++ libraries used by components through the Android application framework. The Core Libraries provide most of the functionality of the Java language like Data Structures, Utilities, File Access, Network Access, Graphics, etc.

The Dalvik Virtual Machine provides an environment in which every Android application runs in its own process, with its own instance with multiple efficient register-based VMs. The Dalvik Executable (.dex) format, is optimized for minimal memory footprint and compilation.

The Linux Kernel provides threading, low memory and process management, network stack, drive model and security. Unlike PC operating systems, mobile phone operating systems are constrained by their hardware, memory, power dissipation and mobility conditions [17].

The most recent Android version is Android 6.0, known as Android Marshmallow.

Google provides the Android SDK for developers for developing applications for Android easily. The Android operating system provides users and developers with a complete suite of software for mobile devices such as an operating system, middleware and key mobile applications.

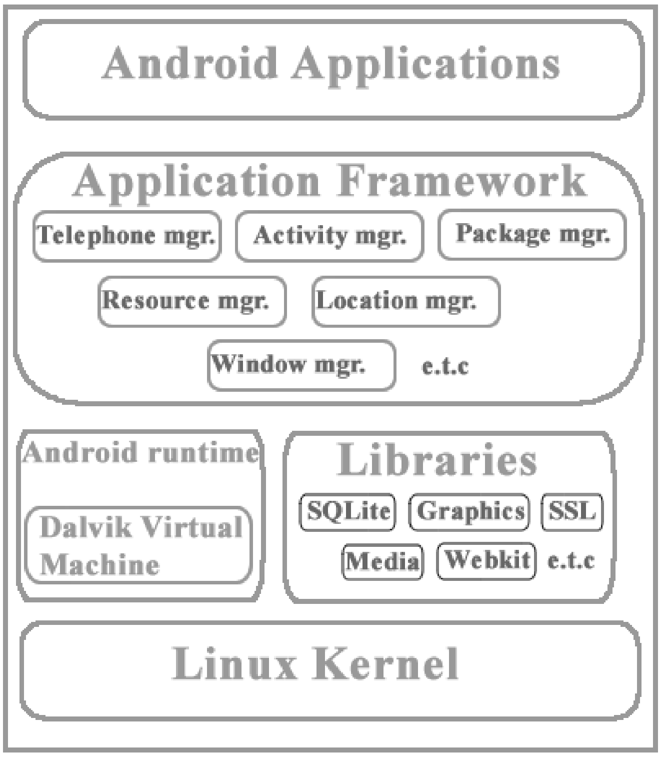


Figure 2.4: Layers of the Android OS

2.4 Review of existing works

There is much research on online-based compilers. This review discusses some previous work on online compilers and how they execute. It also discusses the limitations of each.

2.4.1 Online C/C++ compiler using cloud computing

In 2011, Aamir, et al. [1] developed an *Online C/C++ Compiler Using Cloud Computing*. The aim of their project was to have a system where online programming examinations with C/C++ could be conducted in their school. This project implemented the following functionalities

a. **Compile option:** This functionality allows the user to compile a program typed in the editor area. Upon clicking this option, the user's program is submitted to the cloud server. The cloud server does the compilation and returns the appropriate result.
b. **Execute option:** When this option is clicked, the user is provided with the links to the executable file of the program for him or her to download and subsequently execute. The user has a folder where all programs previously compiled at least once and without errors are stored.
c. **Start test option:** This option allows the user to start writing code. Unless this button is clicked the user cannot start writing code.

All users' programs together with the timestamps of when they were compiled are stored at the server-side database. One notable feature of this work is that users are not allowed to execute their code on the server. Instead a URL is provided for the user to download the executable file. The feature of downloading the executable file onto the user's terminal ensures that malicious code (for example code to format the C: drive on the server itself) written on the server will not execute on the server itself (thereby keeping the server intact and safe).

2.4.1.1 Project architecture

The system uses a dual-layered architecture. The lower layer consists of clients, which are of lower configuration. The upper layer consists of the server. The important components of the upper layer are described as below:

1. A web framework, Visual Studio 2010, which handles the work of scripting and compilation of code;
2. IIS server which handles the client request;
3. Database which stores the client information; and
4. The “cloud hard disk” as a shared resource.

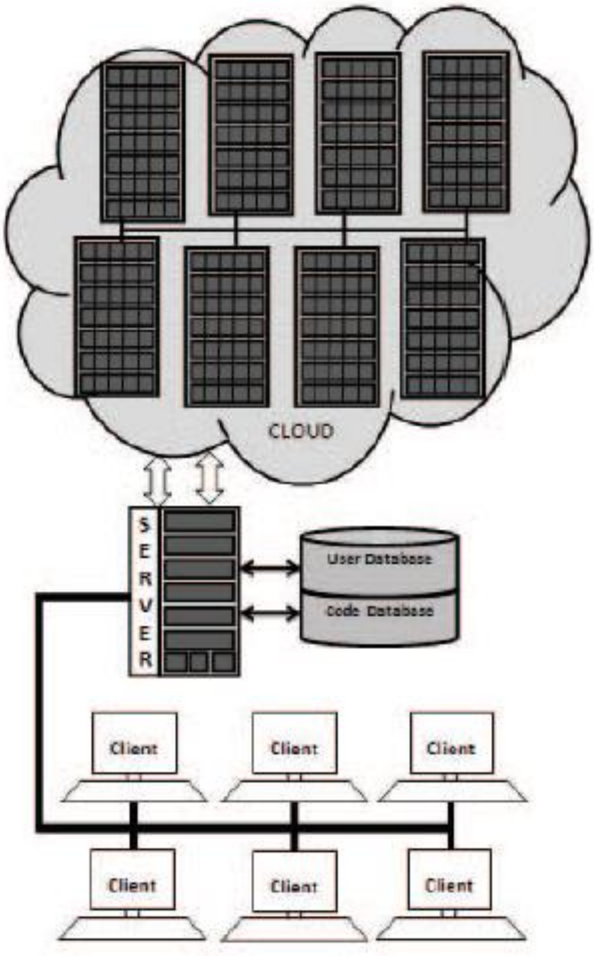


Figure 2.5: Architecture of the project

2.4.1.2 Implementation

The user interface of this research was programmed in HTML and enhanced with ASPX. It was assumed that the user would use his or her favourite text editor to create and correct program files. This assumption allowed the creation of a very simple front end that loads quickly and is platform independent. Although the front end is designed to be as simple as possible with only a few commonly used options, it is sufficiently functional and can be used quickly. The server-side part of the application was implemented using ASPX written in ASP.NET that handles the communication between a user and compiler. The script does the file managing, runs compilers and processes the compilation results. The result is the source code listing or a list of errors sent back to the user.

2.4.1.3 Limitation of the system

The major setback of this system developed by the authors in [1] is that the system only returns the executable (.exe) of any program successfully compiled which requires the user to carry out some installation before being able to execute the executable returned.

Secondly this implementation was designed only for the C programming language and does not provide a mobile application, leaving the user with the single option of accessing the system only via a web browser.

2.4.2 Cloud Compiler Based on Android

In 2014, Vijay et al. [2] developed a *Cloud Compiler Based on Android*. Unlike Aamir et al., Vijay et al. developed an Android-based IDE that could detect the programming language of the code typed.

Vijay et al. identified two simple but substantial problems with today's IDEs. Firstly they require intensive CPU and memory usage which is not available all the time and since these applications are installed on a specific system, it prevents portability. By combining cloud computing and Android technologies, their project aimed to remove the requirement for powerful systems and provide portability to the developer considering that the Android application provides much better functionality than other heavy programming kits for the Android. The compiler they built was embedded in the cloud and used SaaS. The compiler responds by providing the output of the program if successfully compiled or returns errors and warnings.

2.4.2.1 Architecture of the project

The system was designed to work for three fields, which they named zones. So the system was divided into three zones:

Application zone: The application zone consists of the interface from which a client can interact with the proposed system. The modules included in this zone are the Android application and the browser. The Android application is only for versions 2.3 (Gingerbread) and above. Any person who does not have Android can also use the proposed system through a web browser. These interfaces will provide the user with editors and various options through which user can access functions needed such as compile and file upload. The application zones must be provided with an internet connection. Without internet connection, the compilers cannot be used. Users write and store their codes using the editor provided. Whenever the device gets an internet connection, the files will be uploaded automatically if the user chooses.

Communication zone: When code is being sent for compilation, the flow moves into the communication zone. The communication zone is the core part of the model. First, the code's language is detected so that the code should be sent to appropriate compiler. The communication zone also includes scheduling the compilation queue and checking whether the compiler is idle or not; if it is not then the code goes to a wait state. After the compiler is detected in the idle state, the code is sent directly for execution. For getting access to the workspace, the user has to register for the first time and then log in. This transferring of user name and passwords in an encrypted format is included under the communication zone.

Database zone: The database zone consists of total backend contents such as workspace, user name and passwords. These passwords are saved in encrypted format in the database.

The users are provided with a limited workspace for storing their codes or projects. Whenever any particular user logs in, he or she will be provided with their workspace only. These files are accessible either from the Android application or from the web browser.

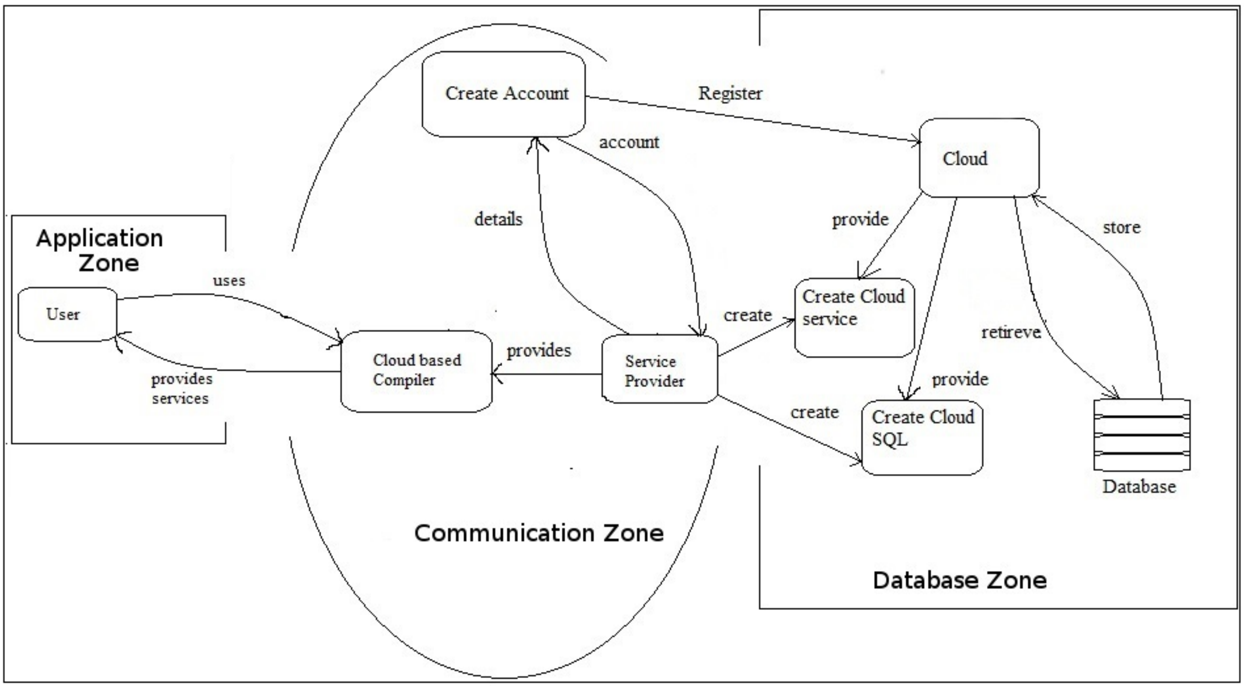


Figure 2.6: **Architecture of the system**

2.4.2.2 Dataflow diagram of the system

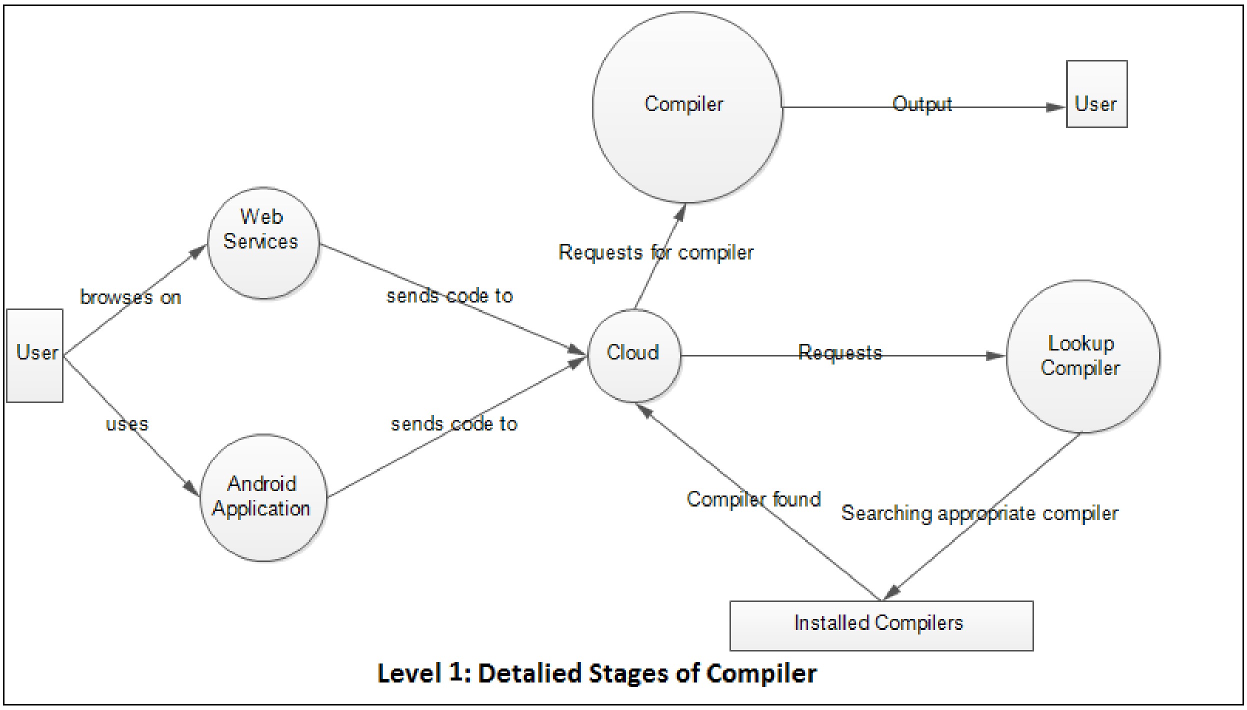


Figure 2.7: Dataflow diagram of the system

2.4.2.3 Implementation of the project

The mobile application was developed using Android programming. The web version of the application was developed with HTML and PHP.

2.4.2.4 Limitations of the system

The limitations of this system are that it does not accept runtime inputs for programs that require such interaction, and this system does not have a provision for running GUI-based programs in any way.

2.4.3 Cloud-based “C - Programming” Android application framework

In 2015, Sonali et al. [3] developed a *Cloud based “C - Programming” Android Application Framework*. In their work, they built an Android-based IDE for running C programs on users’ smart devices.

2.4.3.1 Functionality of the system

For a user to use this system, he or she has to create an account. Upon doing this, the user is assigned a user ID by the system which is stored in the database. With the user ID, a user can log into the system and use the services provided. The Android application is made up of the editor, submit and save, error box and result box (output field). The C code will be edited in the editor of mobile application after which the code must be submitted to the server for further processing. The server hosted in the cloud has the GCC compiler installed on it. Submitted code is compiled and the result is returned to the user’s mobile device. The output is displayed in the output panel for successfully compiled programs, otherwise appropriate error messages are displayed in the error panel on the user’s device. The system can also be assessed from the browser.

2.4.3.2 System architecture

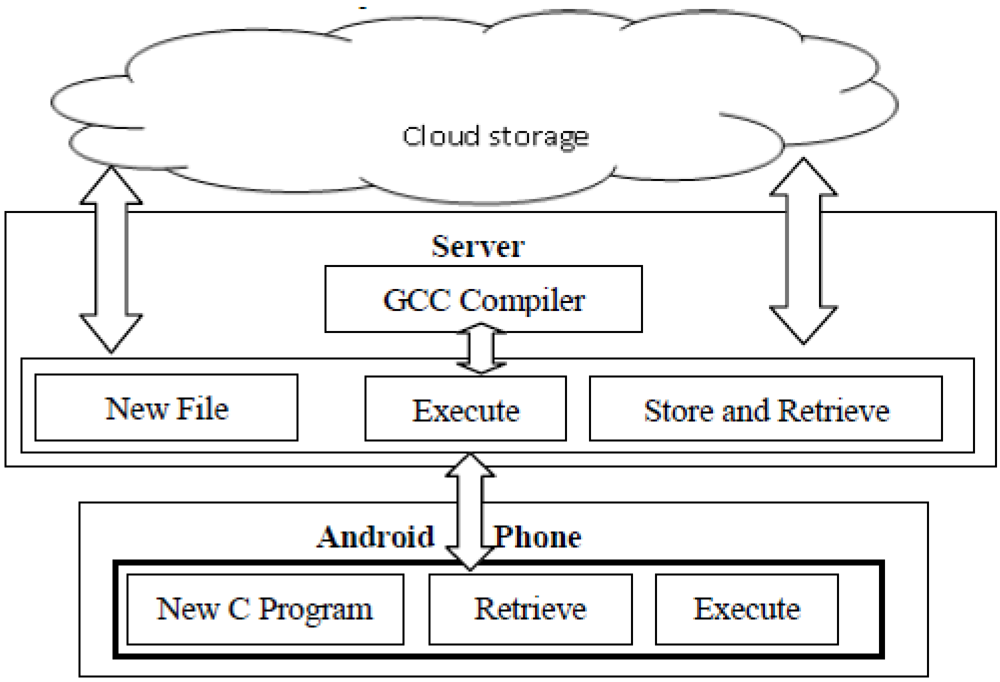


Figure 2.8: System Architecture

DNS3 C PRO Cloud: This module consists of a Java-based tool for configuring the GCC compiler. The tool can compile and execute the C programs and the results will be shown to the user. It is a browser-based app that can be accessible from any browser on the network and also on any smartphone device having groups or Wi-Fi support.

Android application: The user interface on an Android phone in which user can type the C program is developed here. Also compile and execute functionality will be separately provided on the phone, and the result will be displayed in the result box. The phone will be connected to the cloud server via a Wi-Fi network.

Server Application: This service acts as an intermediate layer between a C pro application and an Android phone. Users' requests for compilation and execution of C programs are forwarded to these services and the results returned to the phone.

2.4.3.3 Dataflow diagrams of the system

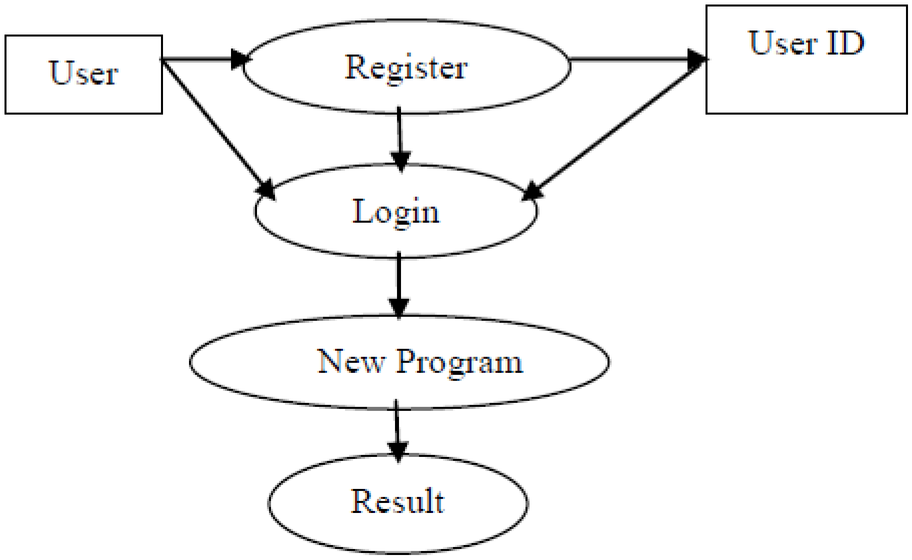


Figure 2.9: Dataflow diagram of the system

2.4.3.4 Implementation

The system was implemented with for the Android mobile operating system using the Android SDK 20 with minimum Android OS API2.2. The database system used was MySQL.

In general it is worthwhile to note that aside from providing a programmer the opportunity to code on the go with a smart device, there are quite a number of other advantages that cloud-based compilers have to offer. Some of these advantages are:

1. User's device memory to install a compiler is saved.
2. There is no need to upgrade the compiler, as every part of maintenance is handled by the cloud provider.
3. The user will not have to install compiler on different devices, just connect to the service and use it.

2.4.3.5 Limitations of the system

This system has no support for any other language than the C programming language.

2.5 Proposed solution to limitations of the existing works

To address the limitation posed by Aamir et al. in [1], the proposed system is built for the Java programming language and allows users to execute programs on their smart devices. Also a mobile application is to be developed for the system with a web version of the system that can be accessed via the browser for users who do not use the Android platform

The system developed by Vijay et al. [2] could accept runtime inputs for programs. The proposed system is to be developed to allow users to send inputs required by their programs for compilation and execution. Also the proposed system will attempt to allow users to compile simple GUI-based programs and display the output as an image embedded in HTML in a browser.

The work of Sonali et al. [3], has the limitation of only compiling and executing C programs. The new system will implement a Java-based compiler.

CHAPTER THREE: RESEARCH METHODOLOGY, SYSTEM DESCRIPTION AND ARCHITECTURE

3.0 Introduction

In developing the system, the Sphere Engine Online Compiler API was used. In view of this, this chapter discusses how the Sphere Engine works and the technologies it uses. The chapter also gives a detailed description and architecture of the proposed system. The proposed system has a database component for user registration, authentication, code compilation and execution. The chapter also gives a detailed description of the database design. It discusses the database system used, the database schema and the relational model.

The server-side technology used for communication with the Sphere Engine API is PHP and the database system used is MySQL. PHP's strong integration with MySQL makes it desirable for this project and to programmers generally.

In the course of developing the system from the beginning right up to the point of development and deployment, the following steps were taken.

Requirement assessment

The requirements of the new system were clearly defined by carrying out a thorough investigation and assessment of the existing system. The strengths and weaknesses of the existing systems were identified and the new sets of requirements to address these weaknesses were formed.

Work flow design

A system workflow was designed based on the system requirements definition to serve as a guide to developing the new system.

Coding

The workflow of the new system was converted into code and debugged.

Testing and deployment

The developed application was tested and deployed ready for use.

3.1 Sphere Engine

Sphere Engine [20] is an online compiler that offers a clean and efficient way to compile and run source code of a programming language within an application. Sphere Engine executes code on remote servers with the aid of a simple easy-to-use API. It has support for over 60 programming languages and libraries.

3.1.1 Sphere Engine API

This section describes the Sphere Engine Compiler's webservice [20], including the use of methods and how to interpret returned data.

3.1.1.1 Functionality

Sphere Engine Compiler's API allows the user to:

1. Upload a source code;
2. Run the program with input data on server side in more than 60 programming languages; and
3. Finally download results of the execution (output, standard error, compilation information, execution time, memory usage, etc.).

3.1.1.2 The webservice

To use the Sphere Engine API, a user is required to register. The Sphere Engine developers offer users a premium service, but upon registration, a user is given a certain number (1000 as at the time of writing this report) of free submissions for a start.

The Sphere Engine Compiler API is a mostly RESTful API. REST stands for Representational State Transfer. It relies on a stateless, client-server, cacheable communication protocol – and in virtually all cases, the HTTP protocol is used. REST is an *architectural style* for designing networked applications. The idea is that rather than using COBRA, RPC or SOAP to connect between machines, simple HTTP is used to make calls between machines.

RESTful applications use HTTP requests to post data (create and/or update), read data (e.g. make queries), and delete data. Thus, REST uses HTTP for all CRUD (Create, Read, Update and Delete) operations [19].

All Sphere Engine API access is over HTTPS and accessed via the http://api.compilers.sphere-engine.com/api/v3/domain.

3.1.1.3 Passing request data to the Sphere Engine API

Request data is passed to the Sphere Engine API by sending the parameters via the POST method in a way similar to how a regular HTML form is submitted. Each function takes at least one parameter (access_token). An access_token is a unique identifier attached to each user to be used when compiling codes. Each user has at least one access_token attached to his or her account but has the privilege to create more access_tokens associated with the account.

3.1.2 Sphere Engine API methods

The Sphere Engine API has four known methods as at the time of writing this report. Each method, as earlier stated, takes at least on parameter (i.e. access_token). If a wrong access_token is provided, an AUTH_ERROR (authentication error) error code is returned.

3.1.2.1 Test connection method

The method name is test. It is a GET method used for testing purposes. With a valid access_token, it returns the same data every time it is called. Using a sample access_token with value d033e22ae348aeb5660fc2140aec3585, the following example request is considered.

```
<?php
$url = "'http://api.compilers.sphere-engine.com/api/v3/test?access_token=
d033e22ae348aeb5660fc2140aec3585"
      $ch = curl_init();
      curl_setopt($ch,CURLOPT_URL, $url);
      curl_setopt( $ch, CURLOPT_RETURNTRANSFER, true );
      $result = curl_exec($ch);
      echo $result;
curl_close($ch);
?>
```

Example response is:

```
{
    "moreHelp": "sphere-engine.com",
    "pi": 3.14,
    "answerToLifeAndEverything": 42,
    "oOok": true
}
```

3.1.2.2 List all supported languages method

This method name is languages. It is a GET method. When supplied a valid access_token, the method returns an associative array of a list of programming languages supported by Sphere Engine. The return value is an associative consisting of:

Table 3.1: Return value for supported languages

Key integer	language id
Value String	language name and version

Example request:

```
<?php
$url = "http://api.compilers.sphere-
engine.com/api/v3/languages?access_token=d033e22ae348aeb5660fc2140aec3585"
      $ch = curl_init();
      curl_setopt($ch,CURLOPT_URL, $url);
      curl_setopt( $ch, CURLOPT_RETURNTRANSFER, true );
      $result = curl_exec($ch);
      echo $result;
curl_close($ch);?>
```

Example response:

```
{
   "7":"Ada (gnat-4.9.2)",
   "45":"Assembler (gcc-4.9.2)",
   "13":"Assembler (nasm-2.11.05)",
   "104":"AWK (gawk) (gawk-4.1.1)",
   "105":"AWK (mawk) (mawk-3.3)",
   "28":"Bash (bash 4.3.30)",
   "110":"bc (bc-1.06.95)",
   "11":"C (gcc-4.9.2)",
   "27":"C# (mono-3.10)",
   "41":"C++ 4.3.2 (gcc-4.3.2)"
}
```

3.1.2.3 Create submissions method

The method name is submissions. It is a POST method. This method is used to submit source code in a programming language to the Sphere Engine API. In addition to the access_token, this method takes the following post parameters.

Table 3.2: Parameters for the Submission Method

sourceCode string	Source code of the submission
Language Integer	Language identifier. Can be retrieved with the languages method described above
Input String	stdin data (if any) that the program requires

A call to this method with the correct parameters returns an id (identifier) of the new submission or an error in the case where an invalid access_token is supplied. The returned id of the new submission can be used to fetch the details of the execution of the submitted source code.

Return value:

Table 3.3: Return value for the submission method

Id string	id (identifier) of the new submission

Errors:

Table 3.4: Error returned by the submission method

400	The provided API token is not a valid Sphere Engine Compilers API token

Example Request:

```
<?php
```

```
$url = "http://api.compilers.sphere-
engine.com/api/v3/submissions/44379157?access_token=d033e22ae348aeb5660fc21
40aec3585&withCmpinfo=true&withOutput=true";
      $fields = json_encode(array(
      'language' => 10,
      'sourceCode' => "class Hello{public static void main(String[]
args){System.out.println(\"Hello World\");}}"
      ));
      $headers= array('Content-Type: application/json');
      $ch = curl_init();
      curl_setopt($ch,CURLOPT_URL, $url);
      curl_setopt($ch,CURLOPT_POSTFIELDS, $fields);
      curl_setopt( $ch, CURLOPT_RETURNTRANSFER, true );
      curl_setopt($ch, CURLOPT_HTTPHEADER, $headers);
      $result = curl_exec($ch);
echo $result;
curl_close($ch);?>
```

Example Response:

```
{
    "id": 123456
}
```

3.1.2.4 Fetch submission status method

The method name is submissions, the same as the one used to submit codes. The difference between the two is that fetching the information about a submission requires an additional mandatory id (identifier) parameter whose information is to be fetched. The parameters it accepts are:

Table 3.5: Parameters accepted by the fetch submission method

Id Integer	
withSource Boolean	Determines whether source code of the submission should be returned.
withInput Boolean	Determines whether input data of the submission should be returned.
withOutput	Determines whether output produced by the program should be returned.

Boolean	
withStderr Boolean	Determines whether stderr should be returned.
withCmpinfo Boolean	Determines whether compilation information should be returned.

The fetch submission method if supplied valid parameters returns the following data:

Table 3.6: Return values for the fetch submission method

langId integer	Submission's language identifier.
langName string	Submission's language name.
langVersion string	Submission's language version.
Time Float	Execution time in seconds.
Date String	Server date and time of submission's creation in the following format: YYYYMMDD HHMMSS; for example: 20090519 023456.
Status Integer	Submission's current status. (Discussed below under Status and Result).
Result Integer	Submission's current result. (Discussed below under Status and Result).
Memory Integer	Memory used by the program.
Signal Integer	Signal raised by the program when an error had occurred.
Source String	Source code of the submission. This value is returned if the withSource parameter is set to true.
Input String	Input data of the submission. This value is returned if the withInput parameter is set to true.
Output String	Output produced by the program. This value is returned if the withOutput parameter is set to true.
Stderr	Stderr produced by the program. This value is returned if the withStderr

String	parameter is set to true.
Cmpinfo String	Compilation information regarding the program. This value is returned if the withCmpinfo parameter is set to true.

Example request:

```
$url = "http://api.compilers.sphere-
engine.com/api/v3/submissions/123456?access_token=d033e22ae348aeb5660fc2140a
ec3585"
      $ch = curl_init();
      curl_setopt($ch,CURLOPT_URL, $url);
      curl_setopt( $ch, CURLOPT_RETURNTRANSFER, true );
      $result = curl_exec($ch);
      echo $result;
curl_close($ch);?>
```

Example response:

```
{
  "langId": 10,
  "langName": java,
  "langVersion": jdk 8u51,
  "time": 5,
  "date": 2016-03-17 10:49:07,
  "status": 0,
  "result": 13,
  "memory": 189056,
  "signal": 24,
  "output": "hello world",
  "cmpinfo": ""
}
```

3.1.3 Status and result

Variables status and result are integer values returned when the details of a submission are fetched. The tables below summarizes the meaning of the integer values returned by status and results variable as well as the meaning of returned error codes.

Table 3.7: Status variable values

Value	Meaning
< 0	waiting for compilation – the submission awaits execution in the queue
0	done – the program has finished
1	compilation – the program is being compiled
3	running – the program is being executed

The Sphere Engine API requires that when the getSubmissionStatus method is used and the status is not equal to 0, then one should wait for 35 seconds and call the method again. When the status is 0, how the program finished can be found out by checking the result variable.

Table 3.8: Result variable values

Value	Meaning
11	compilation error – the program could not be executed due to compilation error
12	runtime error – the program finished because of the runtime error, for example: division by zero, array index out of bounds, uncaught exception
13	time limit exceeded – the program did not stop before the time limit
15	**success – everything went ok**
17	memory limit exceeded – the program tried to use more memory than it is allowed to
19	illegal system call – the program tried to call illegal system function
20	internal error – some problem occurred on Sphere Engine; try to submit the program again and if that fails too, then contact the Sphere Engine developers

Table 3.9: Error codes meaning

Value	Meaning
200	Everything went ok.
401	User name or user's password are invalid.
404	Submission with a specified link could not be found.
400	Language with a specified id does not exist.
401	Access to the resource is denied for the specified user.

3.2 Description of the system

In the section the system's functionalities, architecture, functional system requirements, dataflow diagram, detailed database design and the overall behaviour of the system are discussed.

3.2.1 Functionalities of the system

To use the system developed, a user must be connected to the internet with his or her device through a Wi-Fi network or any other network. The user is then required to first register. During the course of registration, in addition to providing some personal details including email address and a password, the user is required to choose a unique name as his or her root folder. This name is used to create a folder for the user on the system server where all folders and code are stored and can be managed. The root folder name is unique to all users. The system ensures this during the course of registration. With a combination of email address and password, the user can log into the system where he or she directly accesses the compile/editor area and can perform the following functions.

1. **Create folders and files:** The system provides a user with a tree-like directory for managing folders and files. The user can create as many folders and files as desired. The system automatically renames a file folder if a file or folder with the same name already exists in the same directory. The created files and folders are stored on the server for future use by the user.
2. **Modify folders and files:** A user can modify files and folders. The system allows a user to rename a folder and file while maintaining the uniqueness of the names of the files and folders in any directory. A user can also delete any folder or file as desired.
3. **Load files into editor:** This feature allows the user to open files in the editor to enable him or her to edit the content and also to run or compile.
4. **Edit and save file content:** A user can open one or more files in the editor, edit the contents of files and save any changes made.

5. **Compile:** This option allows the user to compile a program (file) currently open in the editor. The resulting compilation information is returned to the user.
6. **Run/execute:** This is a function which when used returns the output of a program (if any) currently open in the editor. If the program contains error(s), they are displayed to the user.

3.2.2 Architecture of the system

The system uses server/client architecture. The client requests services from the server which forwards the request to the cloud Sphere Engine API. The response is returned to the server which interprets it and sends the appropriate result back to the client. The figure below describes the architecture of the system.

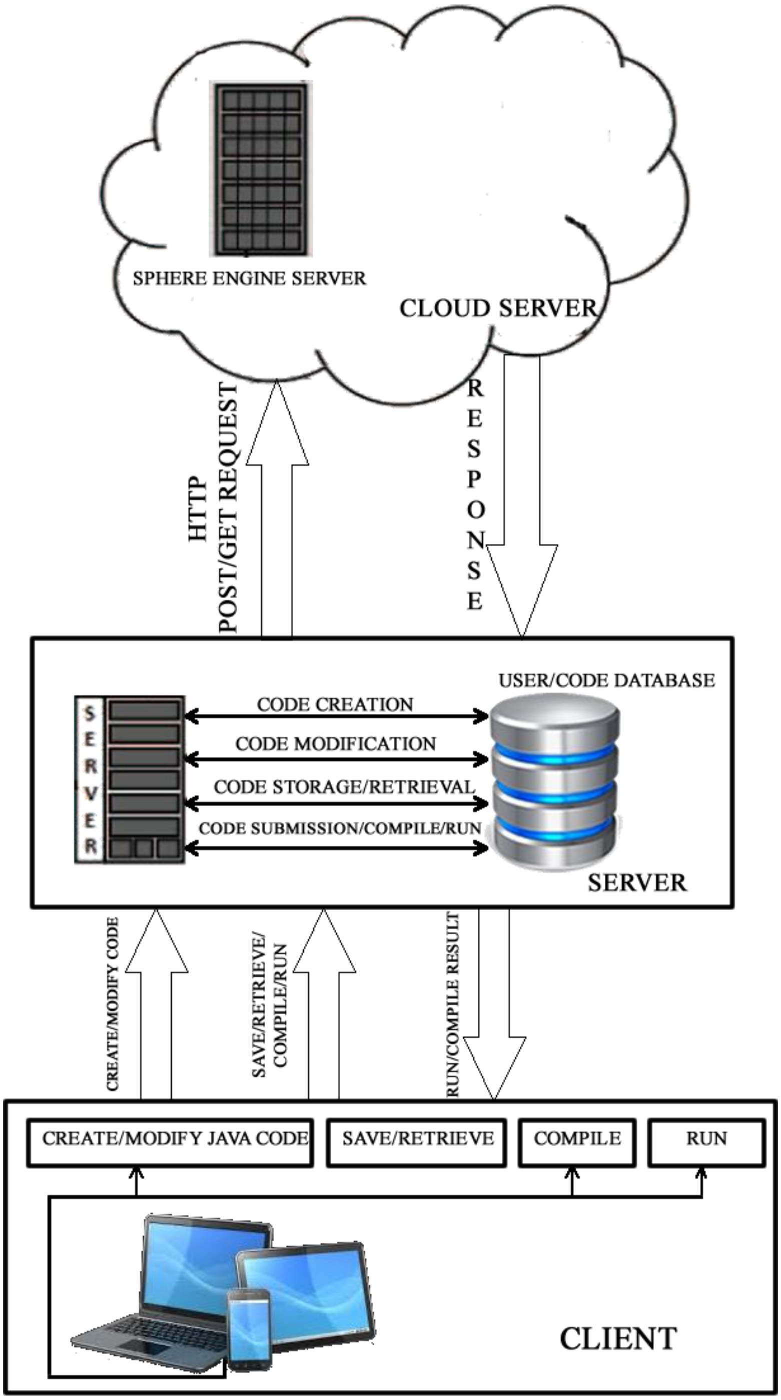


Figure 3.1: **Architecture of the system**

The diagram above shows a brief overview of how the system works and what it does. The client is a user with a mobile device running on the Android operating system (minimum version 4.0, or Ice Cream Sandwich) or a non-Android mobile device or a PC with a browser installed. The user is required to register to use the system. With details from a successful registration, a user can log into the system and beginning writing and compiling code. User-written code is saved in the database on the server and can be retrieved at any time. User requests for compilation or execution are forwarded to the cloud server (Sphere Engine, [20]) and the response returned to the server is sent back with appropriate results and displayed on the user's device. The details of execution and compilation are also stored in the server's database for future referencing.

3.2.3 Dataflow diagram

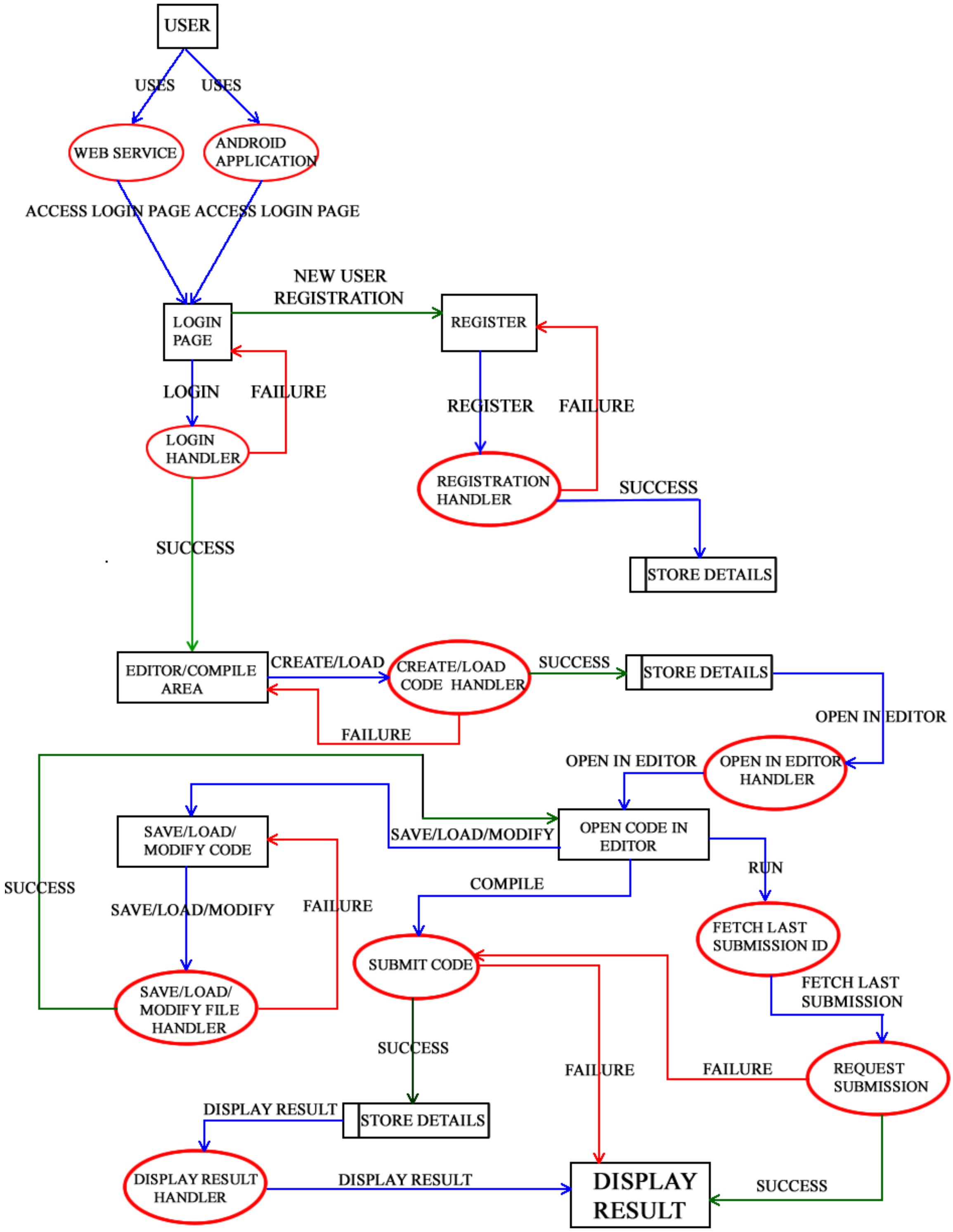


Figure 3.2: Dataflow diagram of the system

The figure above is the dataflow diagram for the compiler we are building. A number of entities and processes are involved. From the point of registration and login until the point of creating and managing code as well as compiling and executing it, the interaction and flow of data among these entities and process are clearly shown.

3.2.4 Functional requirements

3.2.4.1 Development of the Android and web application

The system requires an Android device or a web browser on a PC and non-Android devices, a web server to host the server-side program that interfaces between the user device and the cloud server and finally a cloud server which does the main processing. The user writes and edits the Java programs on his or her device and sends them to the server. The server forwards the request to the cloud server which does the processing and returns the response to the web server which interprets the result and sends it for display to the user's device.

3.2.4.2 Development of the server

The server is created for interfacing between the user and the cloud server. The server requests the cloud server to execute user's code and interprets the returned response and sends the result to the user. The server also handles storage of user's code, compilation and execution information.

3.2.5 Other requirements

3.2.5.1 User interface requirements

Two user interfaces were developed, one for the Android devices developed using the Android SDK with minimum API level 16 and the other was developed for the web users accessing the system via a browser with HTML, CSS and JQuery.

3.2.5.2 Communication requirement

A user is required to have access to the internet via Wi-Fi networks, LAN networks or any other network to use the system.

3.2.6 Database design

A database was designed for the system to store user's information, as well as compilation and execution information. Each user that registers on the system has a unique root folder where folders and files (programs) are stored on the server. Users have access only to their folders and files on the server, which are displayed using a directory-like structure.

3.2.6.1 Entities created

The following entities with the information they hold were created.

USERS_tbl: This entity stores the personal information of any registered user.

ROOT_FOLDER_tbl: The unique root folder chosen by a user during registration is stored here.

CODES_tbl: Details of files (programs) created by a user such as file name, folder level it belongs to and the date and time of creation are stored by this entity.

ROOT_FOLDER_CODES_tbl: Files (programs) belonging to a particular root folder are stored here.

CODE_SUBMISSION_tbl: Whenever a user compiles a program, the submission ID returned by the cloud server is stored here.

SUBMITTED_CODES_tbl: This entity stores the details of program execution and compilation.

3.2.6.2 Entity relationship model

The entity relationship diagram for the system was modelled using the modelling tool MySQL Workbench. The figure below shows the resulting model.

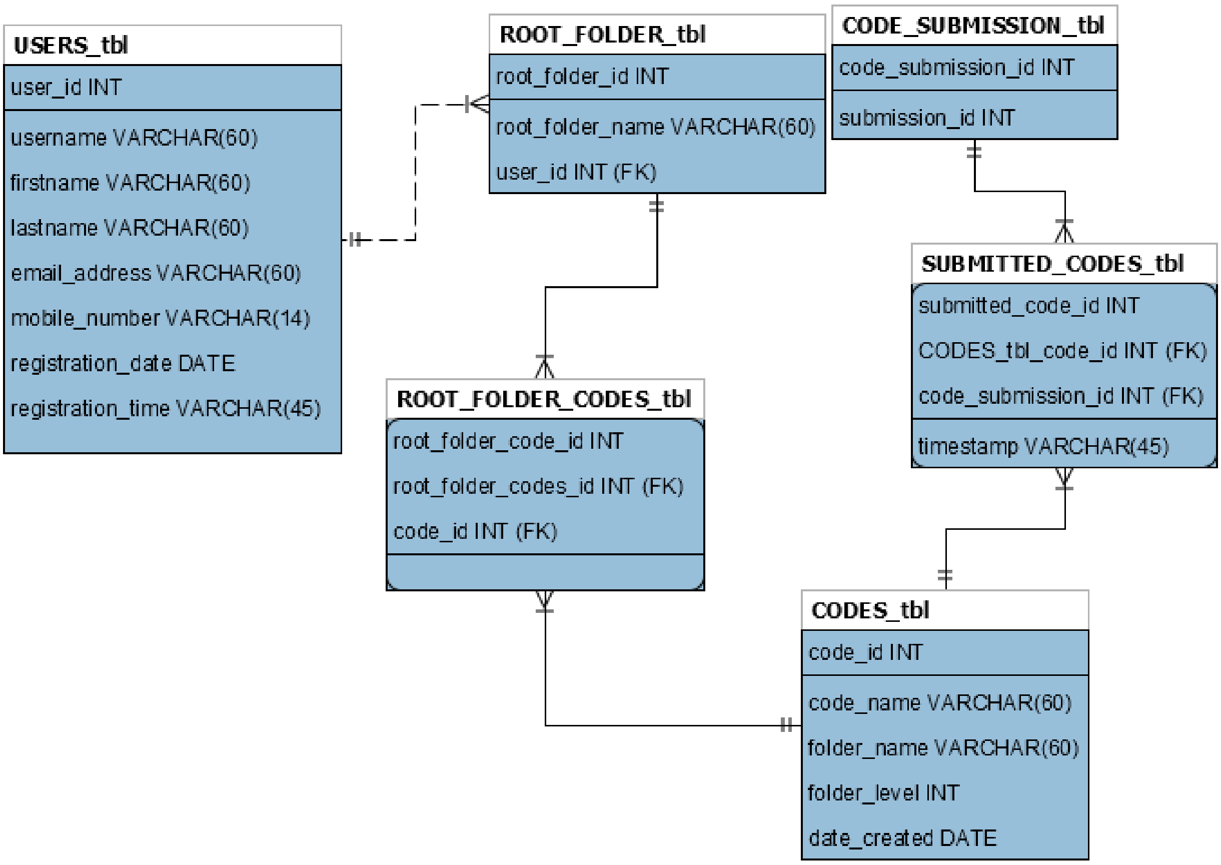


Figure 3.3: Entity relationship diagram of the system

3.2.6.3 Relational model with referential integrity constraints

USERS_tbl

user_id	firstname	Lastname	email_address	mobile_number	Password	registration_date	registration_time

ROOT_FOLDERS_tbl

root_folder_id	root_folder_name	user_id

CODES_tbl

code_id	code_name	folder_name	folder_level	date_created

ROOT_FOLDER_CODES_tbl

root_folder_codes_id	root_folder_id	code_id

CODE_SUBMISSION_tbl

code_submission_id	submission_id

SUBMITTED_CODES_tbl

submitted_codes_id	code_id	code_submission_id	date_submitted

CHAPTER FOUR: IMPLEMENTATION OF THE SYSTEM

4.0 Introduction

In this chapter, the various components of the implemented system are discussed. A directory-like structure was used to display the user folders and files in the root directory. The system also comprises an in-built editor with functionalities such as save, load, find, replace, go-to-line and some others for editing program code. The system also has a compile and execute region (where the compile and run button are situated), the input region (for programs that require stdin input), the compilation information region (to display the compilation information of any program) and the stdout region (to display the output of any program). This section discusses the various components used in developing the system and how they are laid out to produce the compiler interface of the system.

4.1 jsTree

The jsTree [24] is a free, open-source plugin written in jQuery distributed under the MIT license. It is easily extensible, customizable, configurable and easy to theme with support for HTML and JSON data sources as well as AJAX loading.

The jsTree functions properly in either of the two box-models, content-box or border-box. It can be loaded as an AMD module and has a built-in mobile theme for responsive design that can easily be customized. It uses jQuery's event system, so binding callbacks on various events in the tree is familiar and easy. Some of the notable features of jsTree include:

1. Drag and drop support;
2. Keyboard navigation;
3. Inline edit, create and delete;
4. Tri-state checkboxes;
5. Fuzzy searching; and

6. Customizable node types.

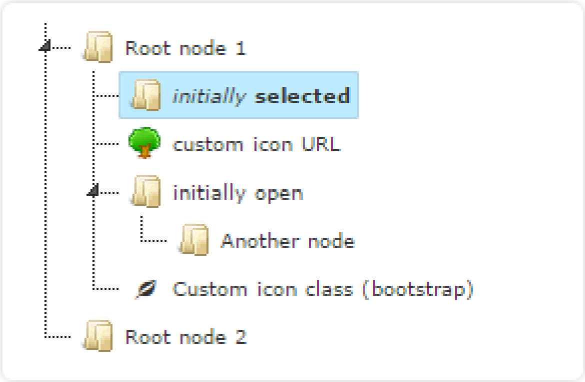
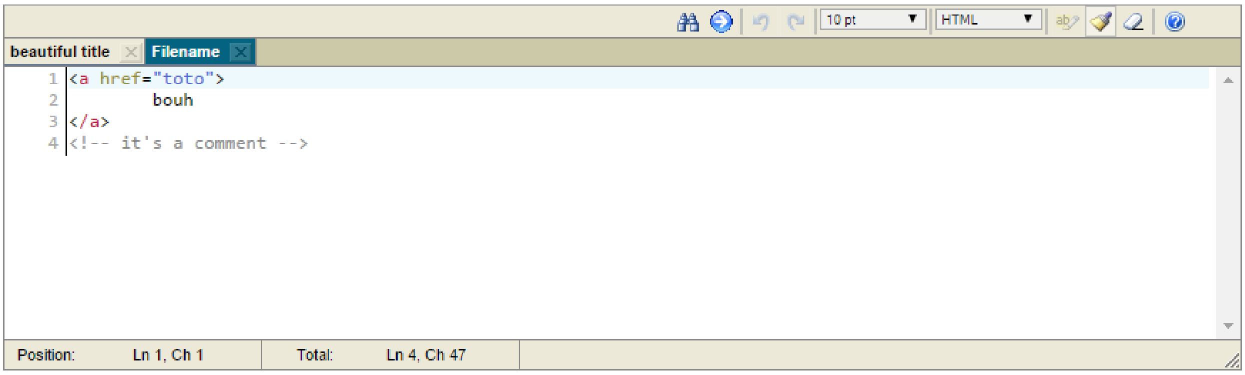


Figure 4.1: jsTree directory

According to the developers, jsTree is compatible with Chrome 14+, Firefox 3.5+, Opera 12+, Safari 4+ and IE8+. It could work with older versions but it has not been tested.

4.2 EditArea Javascript editor

EditArea [25] is a free Javascript editor for source code. (That is not a WYSIWYG editor). This editor is designed to edit source code files in a text area. The main goal is to allow text formatting, search and replace and real-time syntax highlight (for not too heavy text). This editor is free and freely distributable (released under LGPL, Apache and BSD licenses).



```
<a href="toto">
        bouh
</a>
<!-- it's a comment -->
```

Position: Ln 1, Ch 1   Total: Ln 4, Ch 47

Figure 4.2: EditArea Javascript editor

The main features of the EditArea include:

1. Easy to integrate, only one script include and one function call;
2. Tabulation support (allow to write well-formatted source code);
3. Customizable real-time syntax highlighting (currently: PHP, CSS, Javascript, Python, HTML, XML, VB, C, CPP, SQL, Pascal, Basic, etc.);
4. Word-wrap support;
5. Search and replace (with regexp);
6. Auto-indenting new lines;
7. Line numeration;
8. Multilanguage support (currently: Croatian, Czech, Danish, Dutch, English, Esperanto, French, German, Italian, Japanese, Macedonian, Polish, Portuguese, Russian, Slovak, Spanish, and probably more ...);
9. Possible PHP gzip compression;
10. Allow multiple instances;
11. Full screen mode;
12. Possible plugin integration;
13. Possible save and load callback functions;
14. Possible dynamic content management; and
15. Can work in the same environment as “prototype” and “mootools” libraries.

The EditArea has some limitations. Notable among these limitations are:

1. Automatic focus is on the textarea on page load.
2. It can be slow when editing large files (Javascript is not a fast language).
3. Only one syntax language is supported at the same time (no HTML and PHP syntax highlight at the same time).

4.3 The interfaces

Two interfaces, the web browser interface and the Android application interface were developed for the system. Both interfaces consist of a number of components. Some of these components are made up of other subcomponents. Among the components of the system, the most important are:

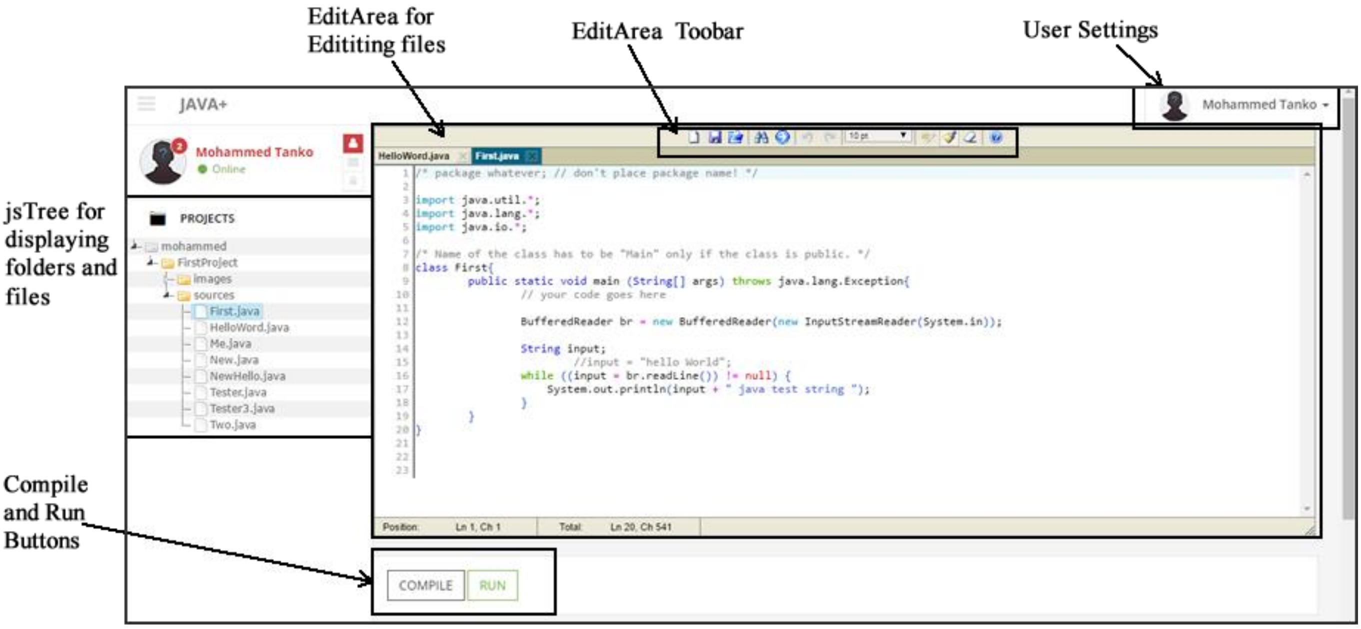
LOGIN AREA: A user wishing to use the system logs into it using login credentials via this area.

REGISTRATION AREA: A user is required to first register before accessing services rendered by this system. This registration process can be done via this component.

COMPILE AREA: This is the area where a user creates, edits, compiles and executes code.

4.3.1 Web browser interface

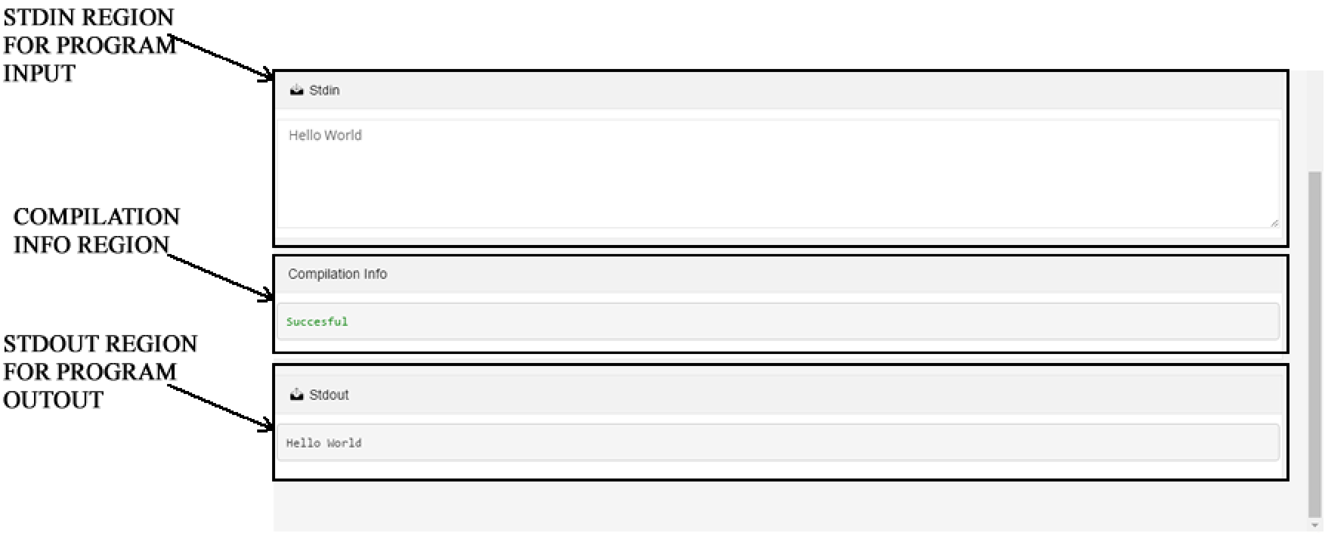
The web browser interface was designed using HTML, CSS and JQuery. The jsTree and EditArea plugin were integrated with other HTML elements to produce the interface for the system. A responsive HTML theme was used to make the system easy to use regardless of the device (non-Android devices) used to access the system.



```
/* package whatever; // don't place package name! */

import java.util.*;
import java.lang.*;
import java.io.*;

/* Name of the class has to be "Main" only if the class is public. */
class First{
        public static void main (String[] args) throws java.lang.Exception{
                // your code goes here

                BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

                String input;
                        //input = "hello World";
                while ((input = br.readLine()) != null) {
                    System.out.println(input + " java test string ");
                }
        }
}
```

Position: Ln 1, Ch 1 Total: Ln 20, Ch 541

Figure 4.3: Components of the Compile area



STDIN REGION FOR PROGRAM INPUT
COMPILATION INFO REGION
STDOUT REGION FOR PROGRAM OUTOUT
Stdin
Compilation Info
Stdout

Hello World

```
Succesful
```

```
Hello World
```

Figure 4.4: Execution region of the Compile Area

4.3.2 Android interface

The Android interface was developed using Android SDK tools provided by Google (the developers of Android). A minimum SDK version of 16 (for Android operating system 4.1 – Jelly Bean) was set. As at the time of implementation, the latest version of the Android operating system available was 5.0 (Lollipop) with SDK version 23, hence the target SDK version was set to 23.

Rather than creating an entirely different interface from scratch for the Android interface, the WebView widget provided by Android was leveraged to bypass the challenges of creating the editor area and the ensure uniformity with the web version of the system.

The WebView class is a view that displays web pages or simply displays some online content within an application. It uses the WebKit rendering engine to display web pages [27]. The URL to the web version of the system was passed to the WebView class to be displayed on the user's device. Javascript was used to detect mobile devices which in order to provide a separate implementation for some features (like right-click) that are not supported by mobile devices. The figures below show the Android interface developed by the system.

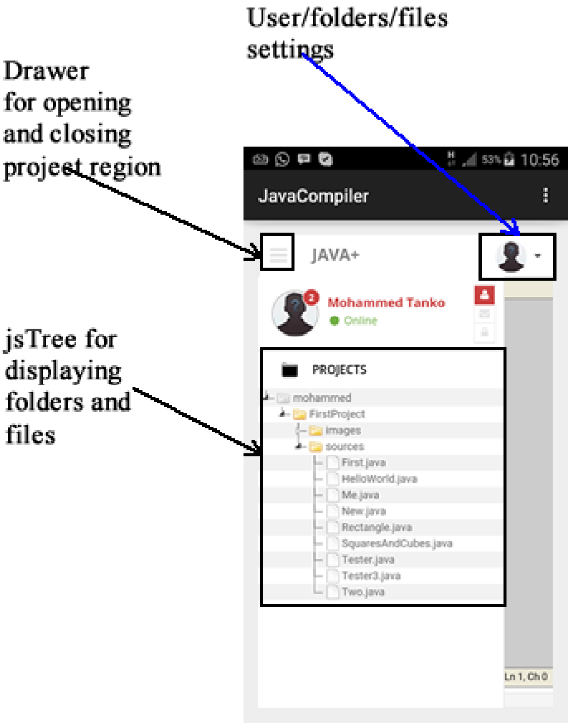


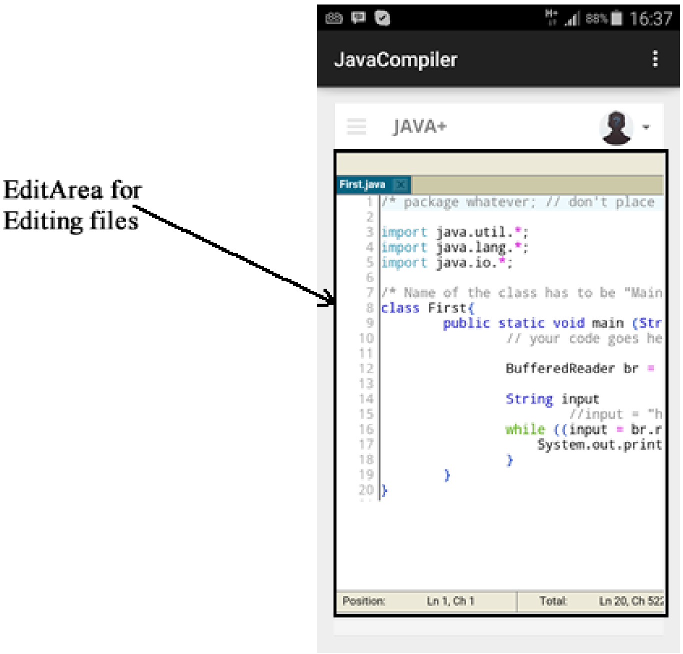
Figure 4.5: **Components of the Android application interface**

The use of the regions highlighted in the figure above is described below.

Drawer: The drawer region allows users to open or close the region containing the folders and files. This feature is very useful to users with small screen devices as it allows them have more space for the editor to occupy.

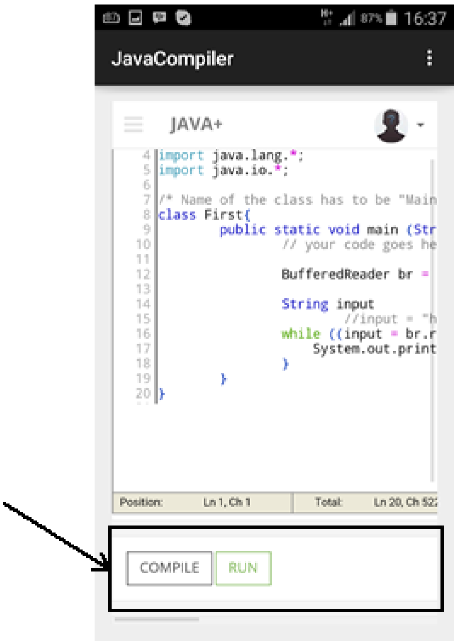
User/folders/files settings: This region contains a set of menu items for managing personal user details, file creation and modification, help and a link to logout.

jsTree region: Users' folders and files created and managed by the jsTree plugin are situated here.



```
/* package whatever; // don't place

import java.util.*;
import java.lang.*;
import java.io.*;

/* Name of the class has to be "Main
class First{
        public static void main (Str
                // your code goes he

                BufferedReader br =

                String input
                        //input = "h
                while ((input = br.r
                    System.out.print
                }
        }
}
```

Figure 4.6: **Editor Area of the Android Interface**



```
import java.lang.*;
import java.io.*;

/* Name of the class has to be "Main
class First{
        public static void main (Str
                // your code goes he

                BufferedReader br =

                String input
                        //input = "h
                while ((input = br.r
                    System.out.print
                }
        }
}
```

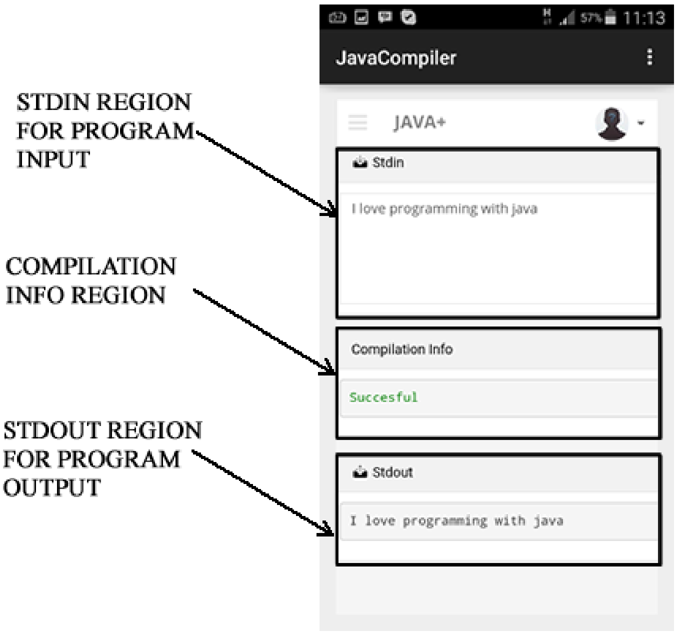
Compile and
Run Buttons

Figure 4.7: Run/Compile region of the Android interface

Figures 4.6 and 4.7 show the following regions:

Edit area: This is the region containing the EditArea editor with which users edit and save their programs.

Compile and Run region: This area has two buttons: compile for compiling a program and run for executing it.



I love programming with java

Succesful

```
I love programming with java
```

Figure 4.8: Execution region of the Android interface

In the figure above, we have:

Stdin region: This is a text area for users to supply inputs for programs that require such inputs.

Compilation info region: Compilation information for a program, i.e. successfully compiled, error during compilation, and details of errors are displayed here.

Stdout region: The output of successfully compiled programs is displayed here.

4.4 Software tools used

In developing this system from the requirement analysis stage up to the development phase, the following software tools were used.

WampServer

WampServer (Windows, Apache, MySQL, PHP) is an all-in-one package that provides a Windows web development environment. It allows one to create web applications. It also comes with phpMyAdmin to easily manage your database. This software was used in developing the web aspect of the system.

Android Studio

Android studio is the official IDE for Android app development, based on intelliJ IDEA. According to [29], on top of intelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance productivity when building Android apps such as a flexible Gradle-based build system, a rich layout editor with support for drag and drop theme editing, etc. This tool was used in developing the Android application of the system.

Notepad++ Text Editor

Notepad++ is a text editor and a source code editor for Windows. Apart from its simplicity, Notepad++ has support for many languages, syntax highlighting and tabbed editing, which allows working with multiple open files. Notepad++ was used in writing the code for the web system.

MySQL Workbench

MySQL Workbench is a unified visual tool that provides data modelling, SQL development and comprehensive administration tools for server configuration, user administration, backup, and much more [28]. This tool was used in modelling all database aspects of the system.

Adobe Fireworks

Adobe Fireworks is a discontinued bitmap and vector graphics editor for websites and apps that provides designers with a lightweight, effective means of creating graphics without getting deep into code. This tool was used to create the majority of the images required for the system. It was also used to create diagrams for this report.

Browsers

A web browser is a software application for retrieving, presenting and traversing information resources on the web. The most commonly used browsers are Microsoft Edge, Mozilla Firefox, Chrome, Opera and Safari. The web version of the system was successfully tested on Microsoft Edge 25+, Mozilla Firefox 4.0+ Chrome 50+, Opera 12+ without any bugs detected. The system is expected to work perfectly with lower versions of these browsers but it has not been tested.

Other tools used include:

Android Device

An Android device with the Android operating system 5.0 (Android Lollipop) was used to test the developed Android application.

4.5 System testing

Testing is a process of performing a number of tests on a system to discover problems before it is delivered to its users. The testing of a system is usually necessary before and after the system is delivered to its potential users.

The server side that mediates between the client (user device or web browser) and the cloud-based Java server (Sphere Engine, [20]) was developed on a 500 GB hard disk, 4 GB RAM PC with a Windows 10 operating system. The server side was deployed to an Ubuntu server.

The web-based system was accessed and tested in the Windows 10 environment using Chrome 14+, Firefox 3.5+, Opera 12+, Safari 4+ and IE8+, with no bugs detected. The system is expected to work on other operating systems with the same browsers.

The Android application was tested an Android mobile device with a 5.10-inch screen display and a 1080 pixel resolution, 2 GB of RAM and 16 GB of internal memory with Android version 5.0 (Lollipop) with no known bugs. The web system was also accessed on the mobile device via Chrome 50+ and Firefox 46+ browsers for Android. The Android application was designed for devices of all screen sizes and resolution with minimum OS version 4.0 (Jelly Bean). The application is expected to work on these devices.

CHAPTER FIVE: SUMMARY, CONCLUSIONS AND RECOMMENDATIONS

This chapter discusses the summary of results obtained, conclusions and recommendations (future works) of the "Cloud-based Java Compiler for Smart Devices"

5.0 Summary of results obtained

The proposed system had a number of aims and objectives as discussed in Chapter One of this thesis. Among the set out objectives and from the analysis of the implementation of the system from the previous chapter, the following is a summary of results obtained:

Design of an Android-based IDE: An Android-based IDE was successfully developed which allows users to create and edit Java programs on their smart devices.

Design of a web-based IDE: A web-based equivalent of the Android-based IDE was also developed to enable users who do not use the Android operating system platform access the system from the mobile devices or PCs via a web browser.

Other functionalities of the system successfully implemented include:

Create, modify and save files: The system allows users to not only create files (Java programs), but they can also save them to the server so that when next they log into the system, they can access their files and continue from where they stopped.

Compile: Users can successfully compile written programs and have the compilation information displayed on their device or PC.

Run: Users can successfully execute their written programs. The result of programs that do not contain any error(s) is displayed to the user or appropriate error messages are displayed.

5.1 Conclusions

In conclusion, the system developed affords a user the opportunity to write and execute Java programs on a device where there is internet connectivity. This makes it possible for a programmer to easily move around with a programming kit on the go. The developed “cloud-based Java compiler for smart devices” can now be integrated into a smart multimedia learning system for Java programming language to allow users who are learners to program on the go with their smart devices.

5.2 Limitations of the system

We were faced with certain challenges during this research, the most obvious of them all being the short period of time allocated. As a result our system has some features we could not build as we would have wished or did not include them at all. Below are the limitations of our system.

Limited amount of compile time: We were unable to build our own cloud-based server for compilation. We had to use an existing server (Sphere Engine, [20]) which offered us only a limited number of compilation times.

No Execution for GUI programs: We were unable to solve the problem posed by the work of Vijay et al. [2]. The approach we initially wanted to use did not work. We discovered a new approach at the end of the work but could not implement it due to time constraints. Hopefully we can include it in a future work.

5.3 Recommendations and future work

The time frame for this research was too short to achieve some of the desired features of the system. Initially we were supposed to implement the cloud-based server that hosts the Java compiler but this could not be done; hence we used an existing online compiler (Sphere Engine, [20]). In view of this, the following are suggested for consideration in future works.

1. **Implement a cloud-based server to host the Java compiler:** A cloud-based server should be developed in the future. As it is we used an existing Java compiler which restricted us to their functionalities.
2. **Handle GUI programs:** During the planning stage, we intended to allow users to execute GUI programs and return an image of the GUI program embedded into Javascript. We started this process but could not complete it before the end of the research. This should be considered for implementation in the future or better still other methods to allow users to execute GUI programs from their smart devices could be explored.
3. **Implement a smart multimedia learning system for Java programming language:** The "Cloud-based Java Compiler for Smart Devices" developed is one component of a proposed smart multimedia learning system for Java programming language. The other components of this smart learning system could be developed and integrated with the already developed Cloud-based Java Compiler for Smart Devices.
4. **Extend to include other programming languages:** The implemented system was strictly for the Java programming language. In future works compilers for other programming languages such as C, C++, Python, etc. could be developed and integrated into this system.

References

[1] Aamir, N.A., Siddharth, P., Arundhati, N., Aditya, P., Venkatesh, B. 2011. “Online C/C++ Compiler Using Cloud Computing”, IEEE Spectrum. DOI 978-1-61284-774-0/11/$26.00

[2] Vijay, R.S., Guruprasad, S.I., Dilip, K.J. (2014). “Cloud Compiler Based on Android”. International Journal of Science and Research (IJSR), 3(9), 2342-2346.

[3] Sonali, S.P., Vinod, B.I. (2015). “Cloud based C - Programming Android Application Framework”. International Journal of Computer Applications (IJCA), 115(12), 20-23.

[4] Utkrash, L. (2013, April 14). Technology and its Role in 21st Century Education. Retrieved April 1, 2016, from http://edtechreview.in/trends-insights/insights/277-role-of-technology-in-21st-century

[5] Why do we need Technology Integration? (2007, November 7). Retrieved April 2, 2016, from http://www.edutopia.org/technology-integration-guide-importance

[6] Mobile Learning. Retrieved April 3, 2016, from http://library.educause.edu/topics/teaching-and-learning/mobile-learning

[7] Baiyun, C., Ryan, S., Luke, B., Sue, B. (2015, June 22). Students’ Mobile Learning Practices in Higher Education: A Multi-Year Study. Retrieved April 3, 2016, from http://er.educause.edu/articles/2015/6/students-mobile-learning-practices-in-higher-education-a-multiyear-study

[8] Berking et al., Mobile Learning Survey Report, 2013, 5.

[9] Amit, N. (2015, July 28). Technology and the Future of Learning. Retrieved April 3, 2016 from https://www.td.org/Publications/Blogs/Global-HRD-Blog/2015/07/Technology-and-the-Future-of-Learning

[10] Java (Programming Language). (2016, April 23). Retrieved April 3, 2016, from https://simple.wikipedia.org/wiki/Java_(programming_language).

[11] Andrew, W. A., (1997). Modern Compiler Implementation in C – Basic Techniques. Cambridge University Press, pp.3-6.

[12] Smartphone OS market Share, 2015 Q2. Retrieved April 3, 2016, from http://www.idc.com/prodserv/smartphone-os-market-share.jsp

[13] Niklaus, W., (2005). Compiler Construction. Addison-Wesley, pp.6-10.

[14] Phases of Compiler. Retrieved April 3, 2016, from http://www.personal.kent.edu/~rmuhamma/Compilers/MyCompiler/phase.htm

[15] Kirk, H., Susan, L.C., Telmo, S. (2013). Cloud Essentials. John Wiley & Sons, Inc., Indianapolis, Indiana, pp.1-47.

[16] Types of Cloud Computing. Retrieved April 6, 2016, from http://www.thecloudtutorial.com/cloudtypes.html.

[17] Nisarg, G., Rahila, S. (2010). Google Android: An Emerging Software Platform For Mobile Devices, International Journal on Computer Science and Engineering (IJCSE), Special issue, ISSN: 0975-3397. 12-17.

[18] Ideone is powered by: Sphere Engine. Retrieved April 24, 2016, from http://www.ideone.com/sphere-engine.

[19] Elkstein, M. Learn REST: A Tutorial (2008, February 9). Retrieved April 26, 2016, from http://rest.elkstein.org/2008/02/what-is-rest.html

[20] API Documentation. Retrieved February 22, 2016, from http://sphere-engine.com/services/compilers/docs

[21] Ravishanker, K. (2014, January 1). PHP CURL POST & GET Examples – Submit form using PHP CURL. Retrieved March 5, 2016, from http://hayageek.com/php-curl-post-get/

[22] Elmasri, R. (2002). Fundamentals of Database Systems. (3rd Edition), University of Texas Press, Texas. pp 165 - 180

[23] Ian, S. (2011). Software Engineering (8th Edition), China Machine Press, China. pp.119 - 170

[24] What is jsTree? Retrieved May 6, 2016, from http://www.jstree.com.

[25] Edit Area. Retrieved May 9, 2016, from http://www.cdolivet.com/editarea/

[25] Edit Area Examples. Retrieved May 9, 2016, from http://www.cdolivet.com/editarea/editarea/exemples/exemple_full.html

[25] WebView. Retrieved May 13, 2016, from http://developer.android.com/reference/android/webkit/WebView.html

[26] MySQL Workbench. Retrieved May 14, 2016, from http://www.mysql.com/products/workbench

[27] Android Studio Overview. Retrieved May 14, 2016, from http://developer.android.com/tools/studio/index.html

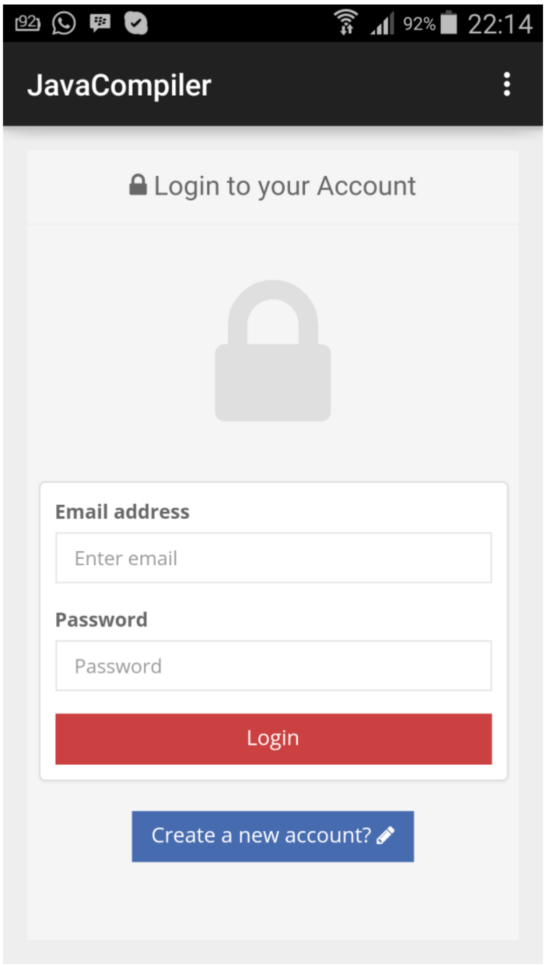
[28] Trefis T. (photographer). (2015). *Reasons Why Nokia May Be Planning to Re-enter The Snartphone Business.* [Digital image] retrieved May 26, 2016 from http://www.forbes.com/sites/greatspeculations/2015/07/14/reasons-why-nokia-may-be-planning-to-re-enter-the-smartphone-business/#747af80f4869.

[29] Mayer, R.E., Moreno, R. (2002). Aids to Computer-based Multimedia Learning. In Learning and Instruction, volume 12, 107-119

Appendix A

Images of the Developed Android Application

1.0



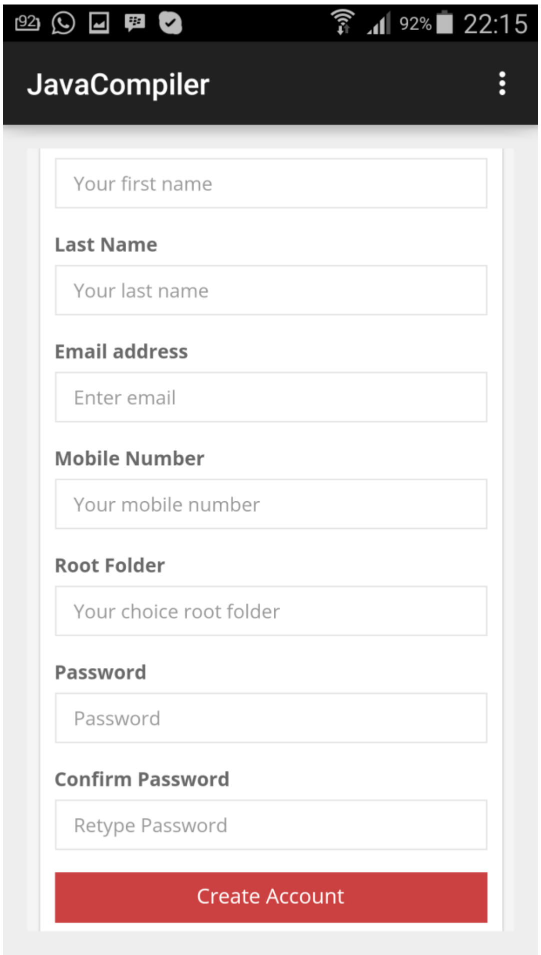
Email address

Enter email

Password

Password

2.0



Your first name

Last Name

Your last name

Email address

Enter email

Mobile Number

Your mobile number

Root Folder

Your choice root folder

Password

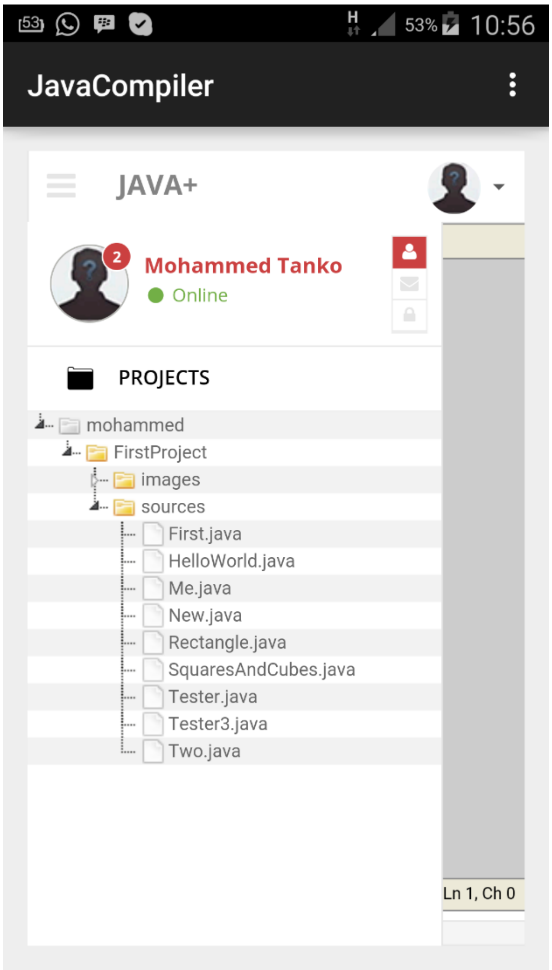
Password

Confirm Password

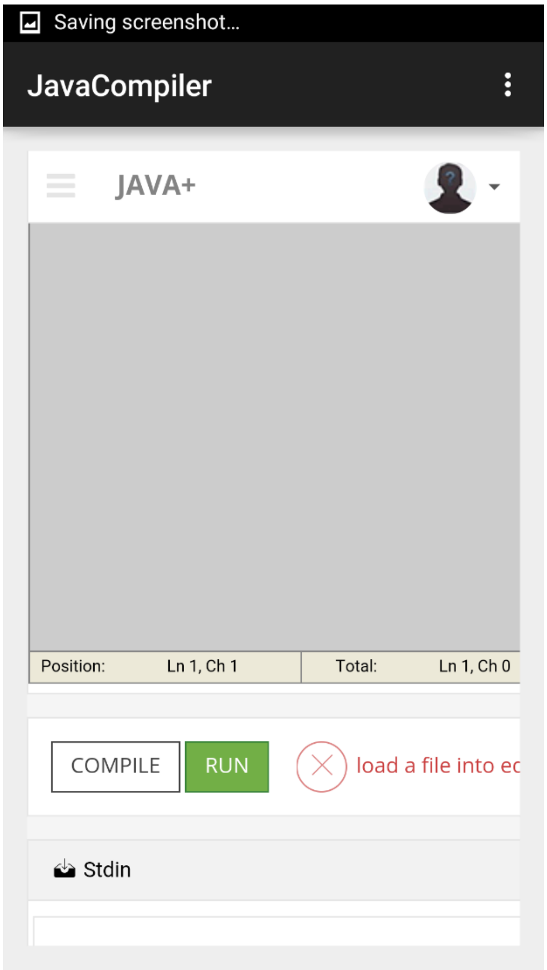
Retype Password

LOGIN PAGE

REGISTRATION PAGE

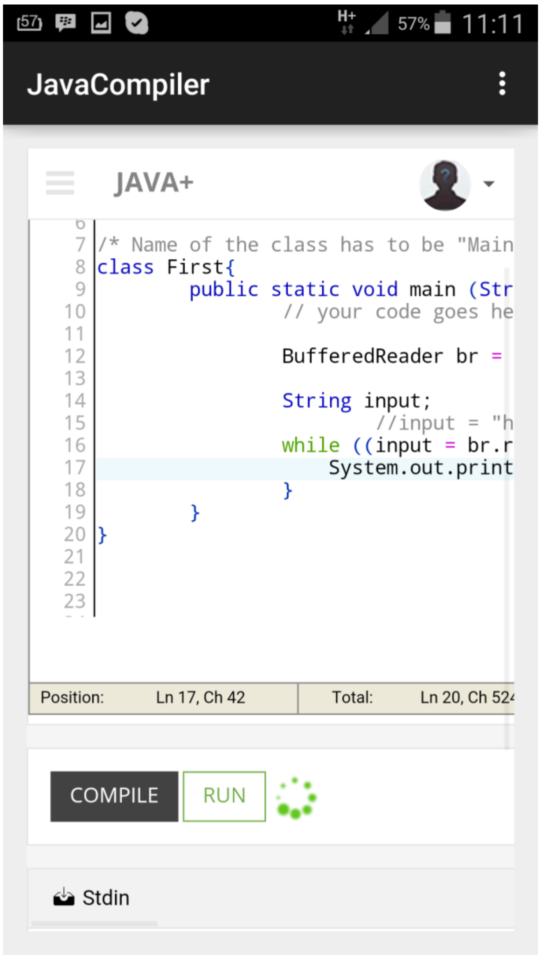


COMPILE AREA WITH OPEN DRAWER SHOWING FOLDERS AND FILES



load a file into ec

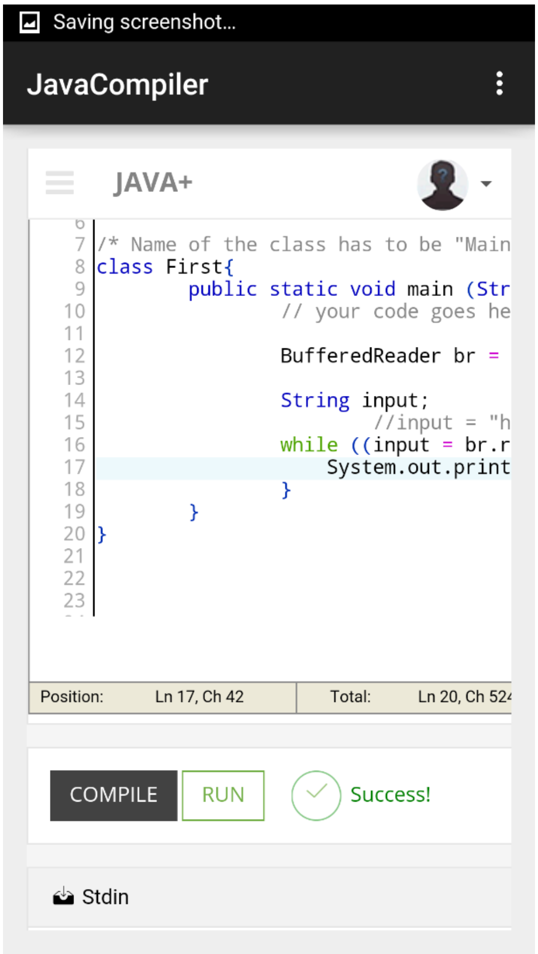
ERROR MESSAGE DISPLAYED WHILE ATTEMPTING TO RUN OR COMPILE WITHOUT OPENING A FILE IN THE EDITOR



```
/* Name of the class has to be "Main
class First{
        public static void main (Str
                // your code goes he

                BufferedReader br =

                String input;
                        //input = "h
                while ((input = br.r
                    System.out.print
                }
        }
}
```

Position: Ln 17, Ch 42   Total: Ln 20, Ch 524

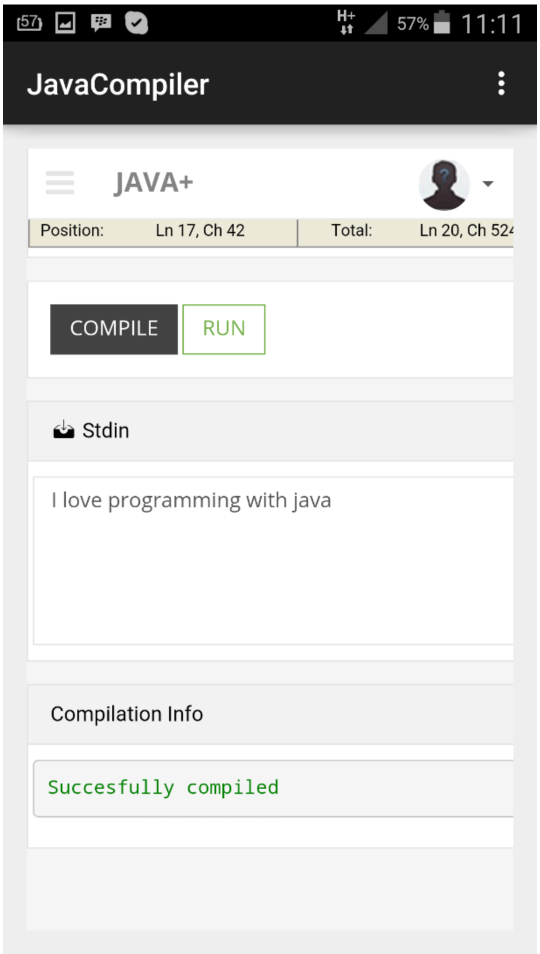
COMPILING OR RUNNING
A PROGRAM



```
/* Name of the class has to be "Main
class First{
        public static void main (Str
                // your code goes he

                BufferedReader br =

                String input;
                        //input = "h
                while ((input = br.r
                    System.out.print
                }
        }
}
```

Position: Ln 17, Ch 42   Total: Ln 20, Ch 524

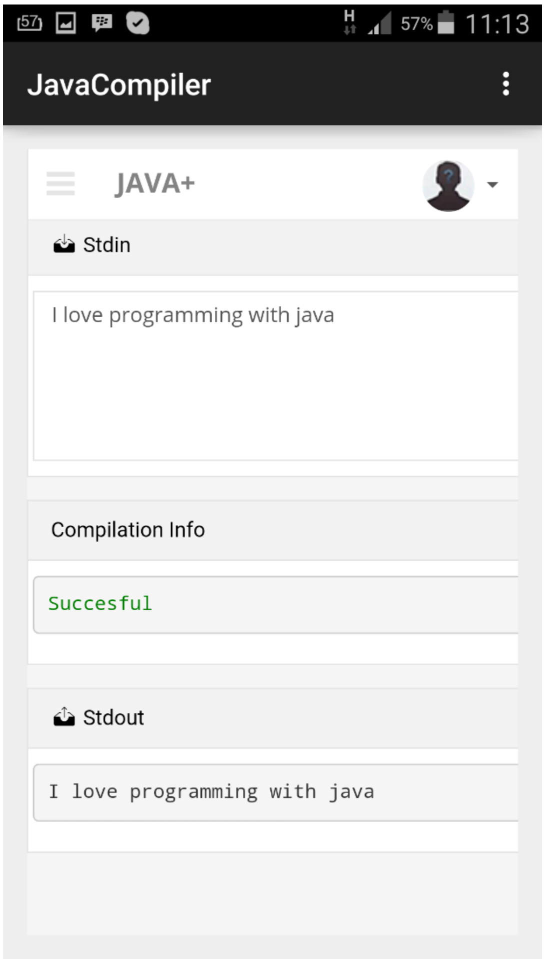
PROGRAM SUCCESSFULLY
COMPILED OR EXECUTED



Position: Ln 17, Ch 42 Total: Ln 20, Ch 524

```
I love programming with java
```

```
Succesfully compiled
```



```
I love programming with java
```

```
Succesful
```

```
I love programming with java
```

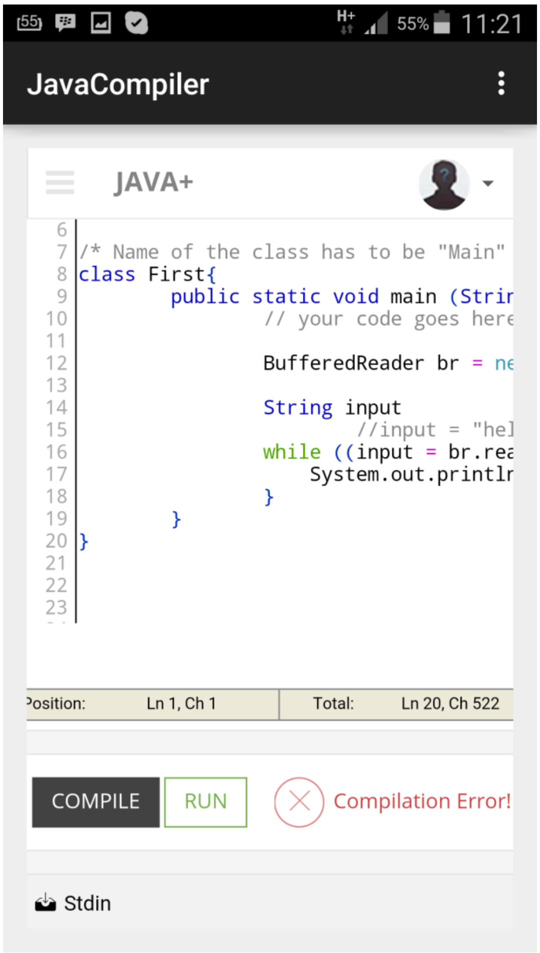
COMPILATION INFORMATION

OF PROGRAMS SUCCESSULLY

COMPILED

OUTPUT OF A PROGRAM BASED

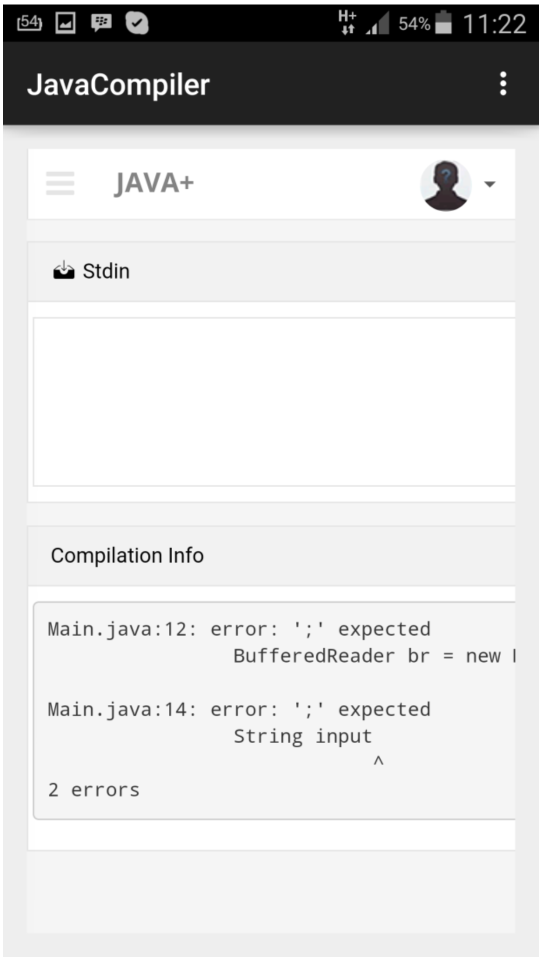
ON USER INPUT SUPPLIED



```
6
7  /* Name of the class has to be "Main"
8  class First{
9          public static void main (Strir
10                 // your code goes here
11
12                 BufferedReader br = ne
13
14                 String input
15                         //input = "hel
16                 while ((input = br.rea
17                     System.out.printlr
18                 }
19         }
20 }
21
22
23
```

A PROGRAM WITH COMPILATION

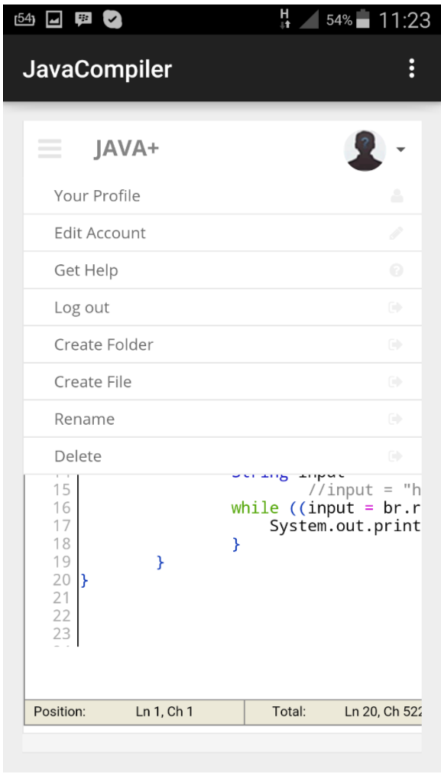
ERROR



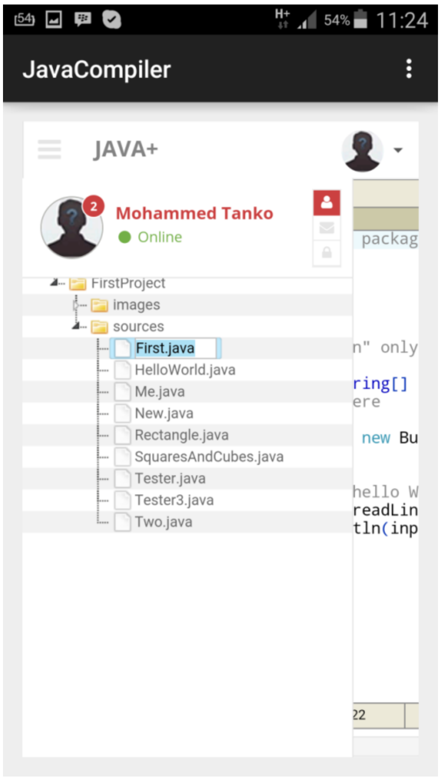
```
Main.java:12: error: ';' expected
                BufferedReader br = new I

Main.java:14: error: ';' expected
                String input
                            ^
2 errors
```

COMPILATION DETAILS OF

PROGRAM WITH ERROR

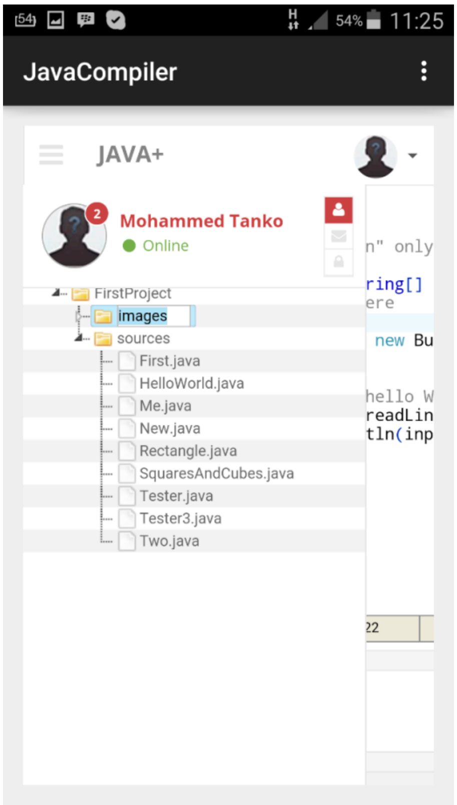


```
                    //input = "h
              while ((input = br.r
                  System.out.print
              }
        }
}
```



MENU TO MANAGE FOLDERS, FILES AND PERSONAL SETTINGS

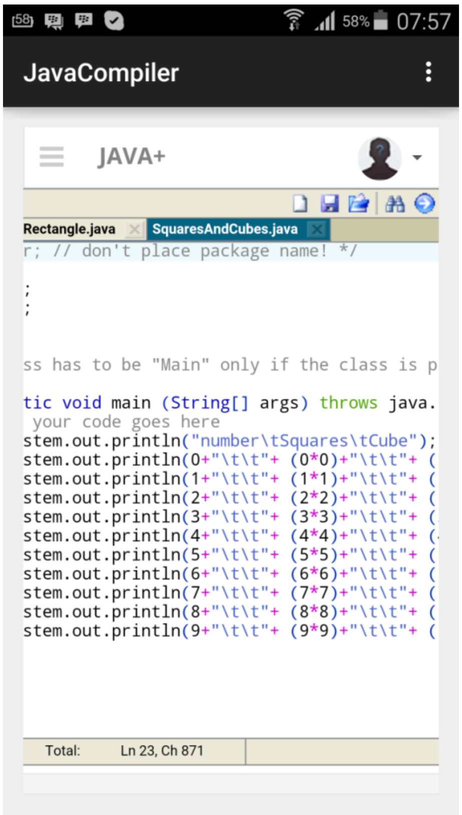
CREATING OR RENAMING AN EXISTING FILE



```
FirstProject
    images
    sources
        First.java
        HelloWorld.java
        Me.java
        New.java
        Rectangle.java
        SquaresAndCubes.java
        Tester.java
        Tester3.java
        Two.java
```

```
n" only
ring[]
ere
 new Bu
hello W
readLin
tln(inp
```

CREATING OR RENAMING AN EXISTING FOLDER

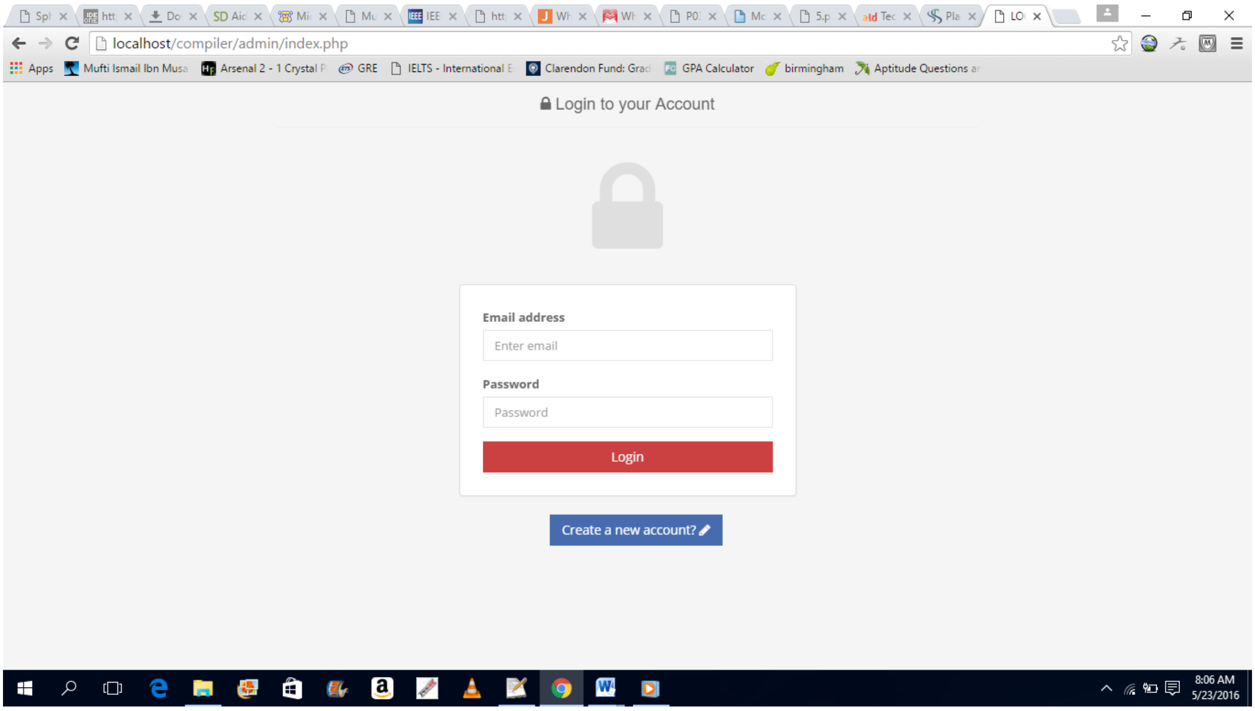


```
r; // don't place package name! */

;
;


ss has to be "Main" only if the class is p

tic void main (String[] args) throws java.
 your code goes here
stem.out.println("number\tSquares\tCube");
stem.out.println(0+"\t\t"+ (0*0)+"\t\t"+ (
stem.out.println(1+"\t\t"+ (1*1)+"\t\t"+ (
stem.out.println(2+"\t\t"+ (2*2)+"\t\t"+ (
stem.out.println(3+"\t\t"+ (3*3)+"\t\t"+ (
stem.out.println(4+"\t\t"+ (4*4)+"\t\t"+ (
stem.out.println(5+"\t\t"+ (5*5)+"\t\t"+ (
stem.out.println(6+"\t\t"+ (6*6)+"\t\t"+ (
stem.out.println(7+"\t\t"+ (7*7)+"\t\t"+ (
stem.out.println(8+"\t\t"+ (8*8)+"\t\t"+ (
stem.out.println(9+"\t\t"+ (9*9)+"\t\t"+ (
```

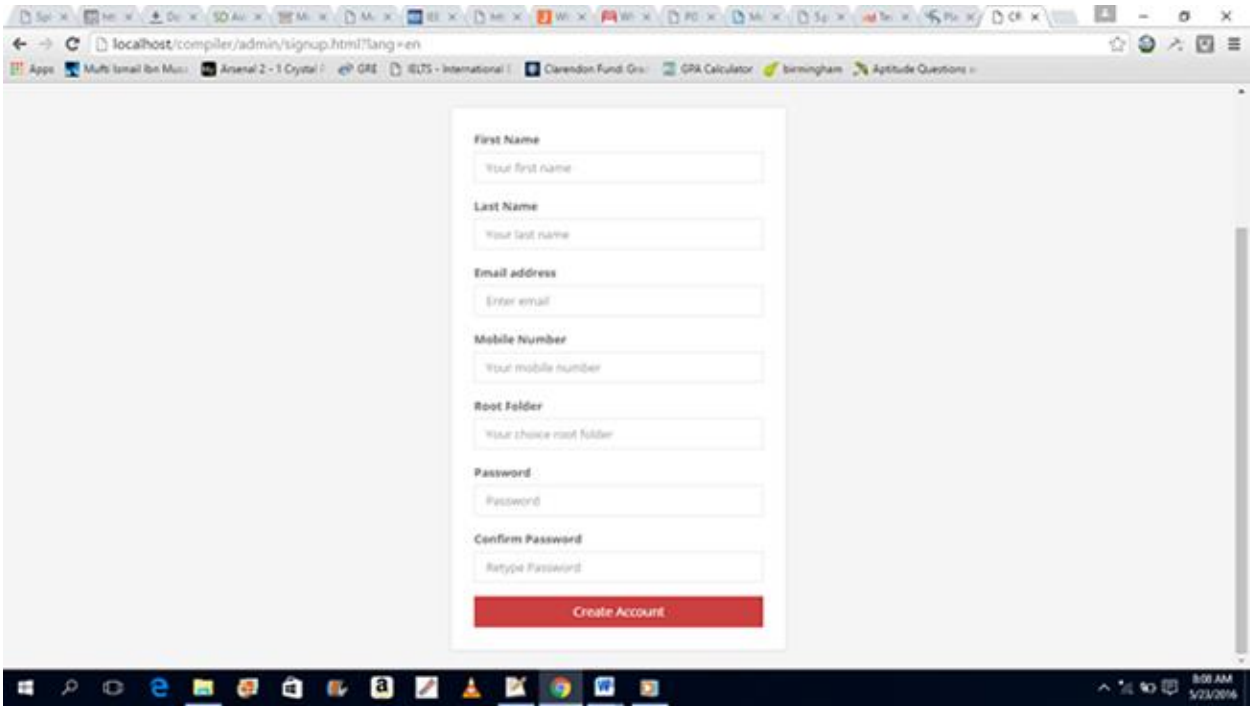
Total: Ln 23, Ch 871

EDITOR SHOWING MULTIPLE TABS OPEN AND TOOLBAR

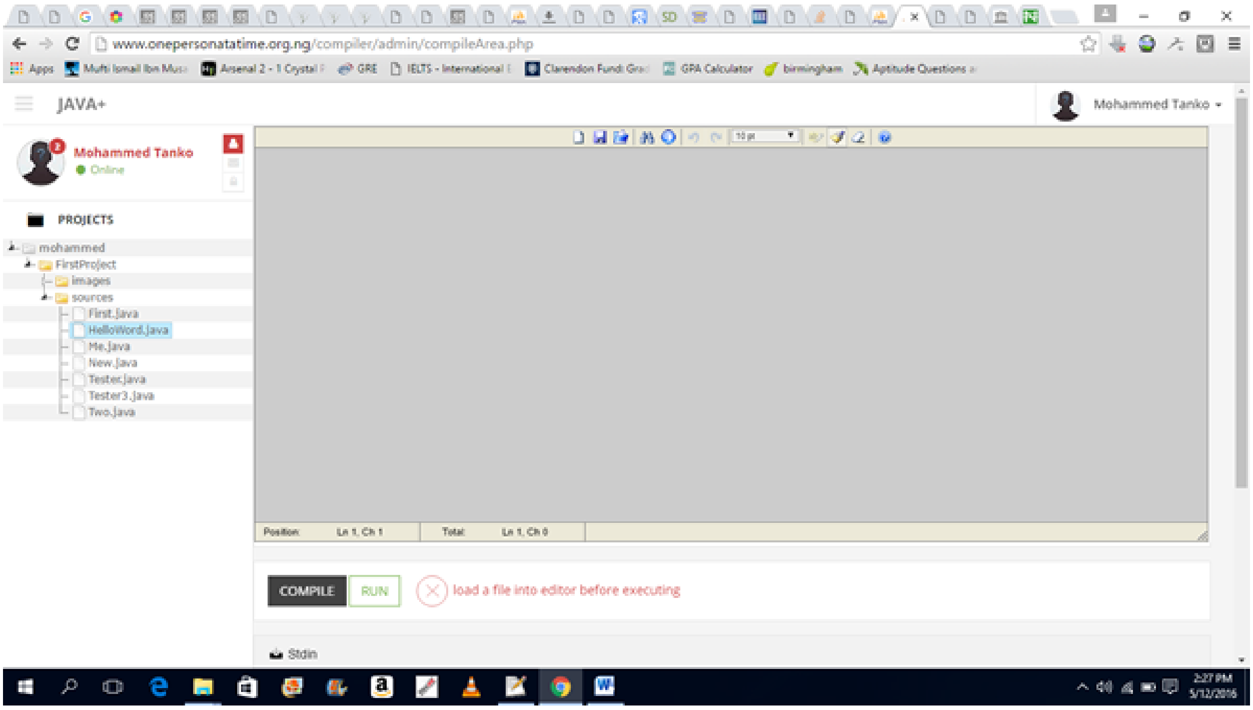
Images of the WEB-UI of the System



LOGIN PAGE

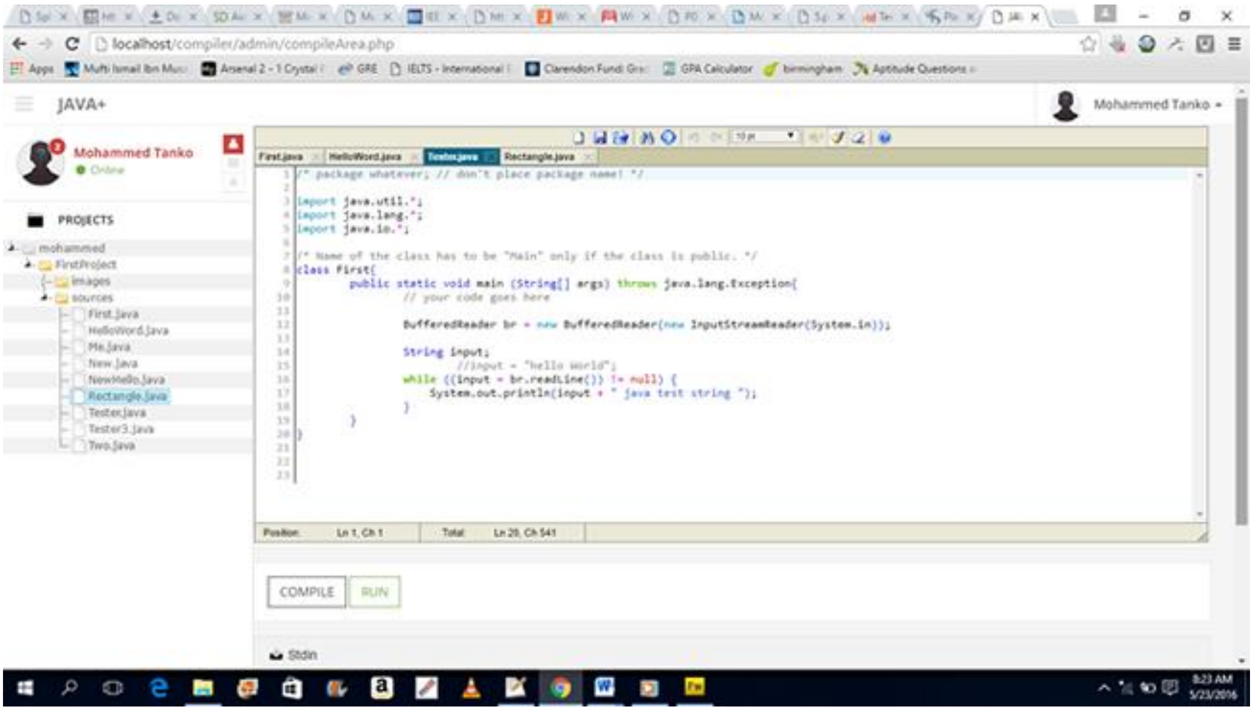


REGISTRATION PAGE



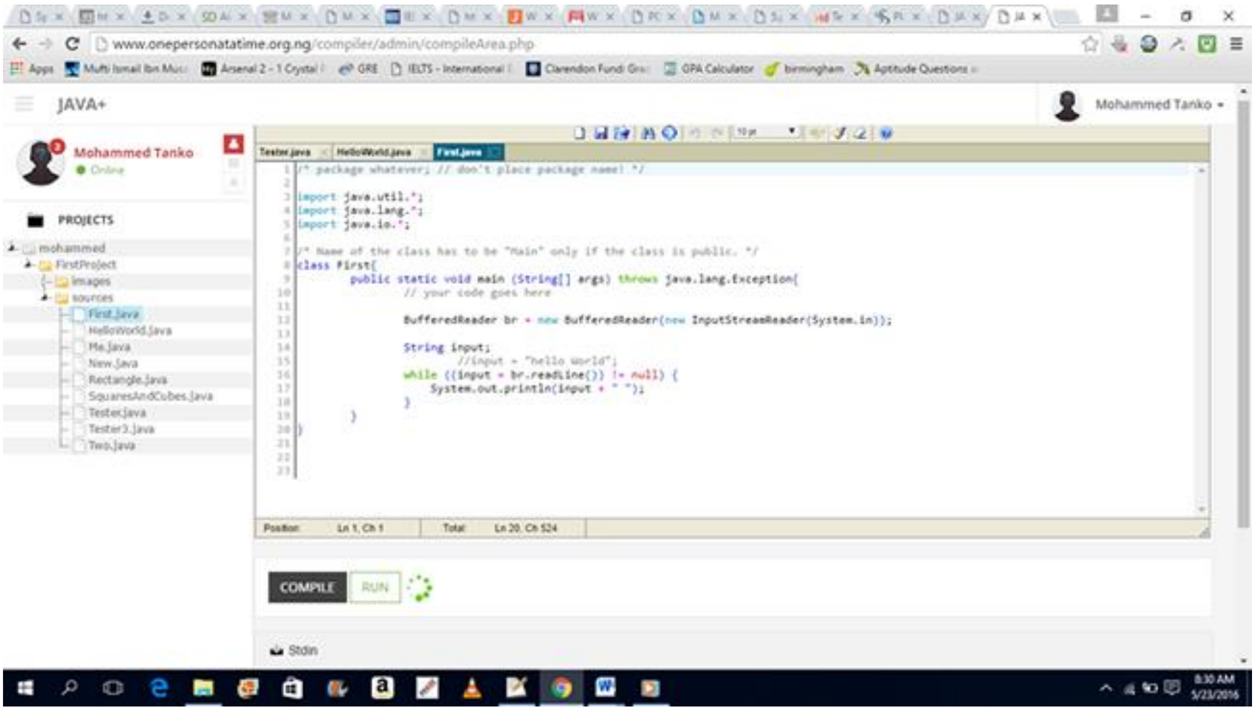
load a file into editor before executing

COMPILE AREA SHOWING FOLDERS AND FILES AND ERRORS WHILE ATTEMPTING TO COMPILE OR RUN WITHOUT OPENING A FILE IN EDITOR



```
/* package whatever; // don't place package name! */

import java.util.*;
import java.lang.*;
import java.io.*;

/* Name of the class has to be "Main" only if the class is public. */
class First{
        public static void main (String[] args) throws java.lang.Exception{
                // your code goes here

                BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

                String input;
                        //input = "hello world";
                while ((input = br.readLine()) != null) {
                    System.out.println(input + " java test string ");
                }
        }
}
```

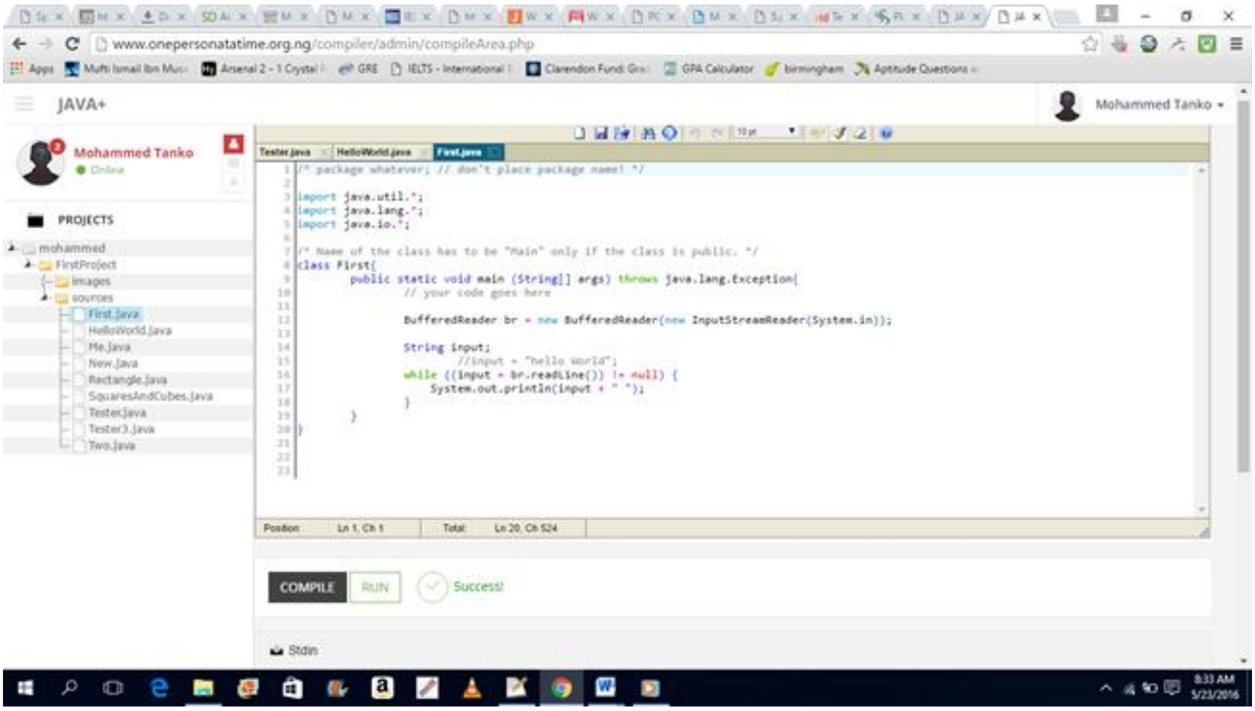
EDITOR SHOWING MULTIPLE OPEN TABS AND TOOLBAR



```java
/* package whatever; // don't place package name! */

import java.util.*;
import java.lang.*;
import java.io.*;

/* Name of the class has to be "Main" only if the class is public. */
class First{
        public static void main (String[] args) throws java.lang.Exception{
                // your code goes here

                BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

                String input;
                        //input = "hello world";
                while ((input = br.readLine()) != null) {
                    System.out.println(input + " ");
                }
        }
}
```

Position Ln 1, Ch 1 Total Ln 20, Ch 524

COMPILING OR RUNNING A PROGRAM

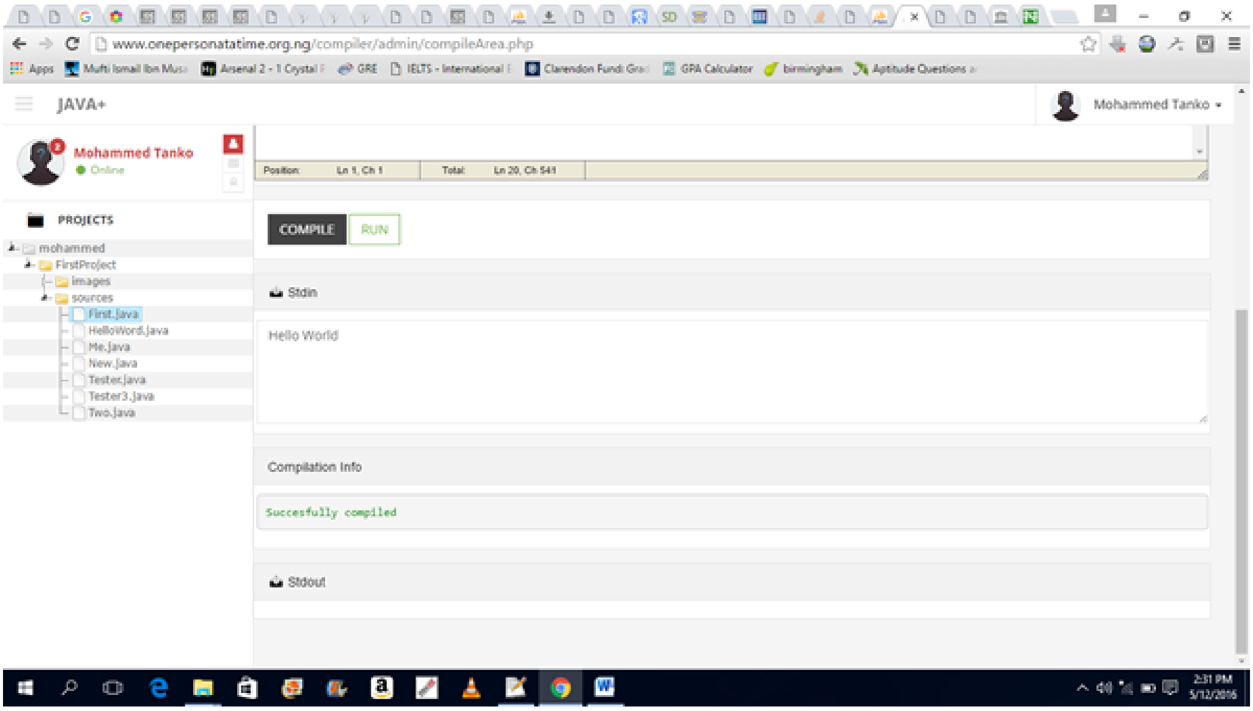


```java
/* package whatever; // don't place package name! */

import java.util.*;
import java.lang.*;
import java.io.*;

/* Name of the class has to be "Main" only if the class is public. */
class First{
        public static void main (String[] args) throws java.lang.Exception{
                // your code goes here

                BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

                String input;
                        //input = "hello world";
                while ((input = br.readLine()) != null) {
                    System.out.println(input + " ");
                }
        }
}
```

Position Ln 1, Ch 1 Total Ln 20, Ch 524

Success!

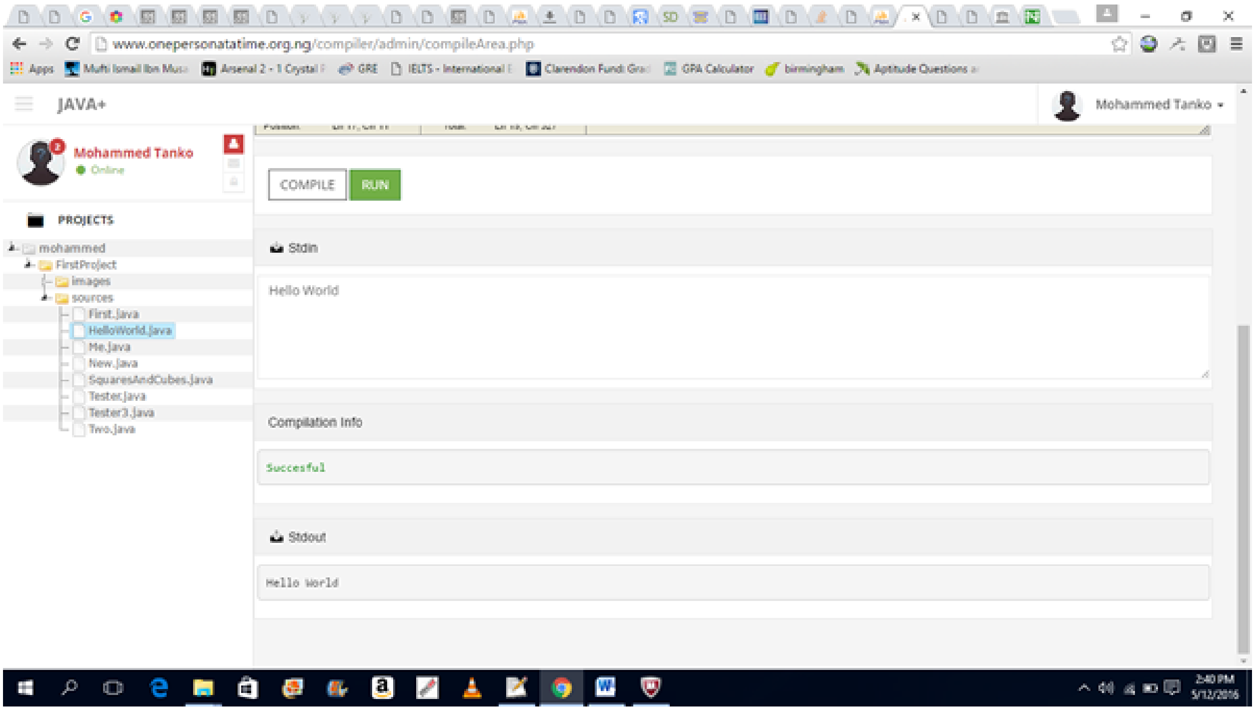
PROGRAM SUCCESSFULLY COMPILED OR EXECUTED



```
Hello World
```

```
Succesfully compiled
```

COMPILATION INFORMATION OF PROGRAMS SUCCESSFULLY COMPILED

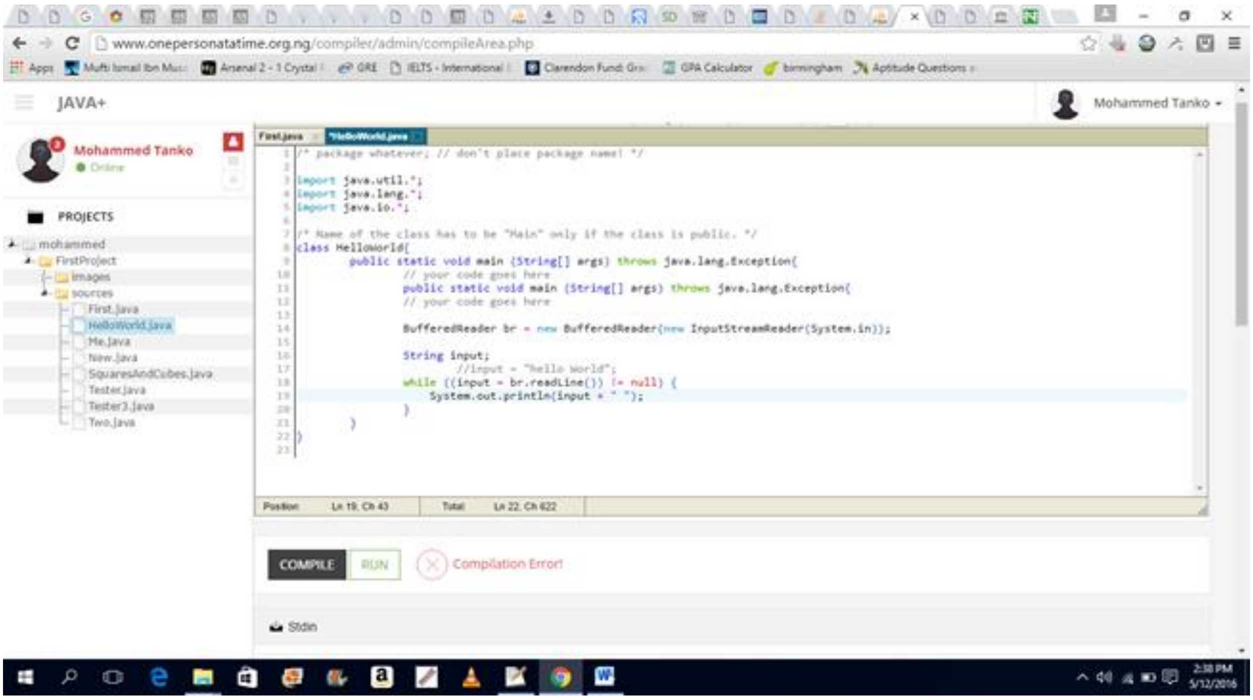


```
Hello World
```

```
Succesful
```

```
Hello World
```

OUTPUT OF A PROGRAM BASED ON USER INPUT SUPPLIED

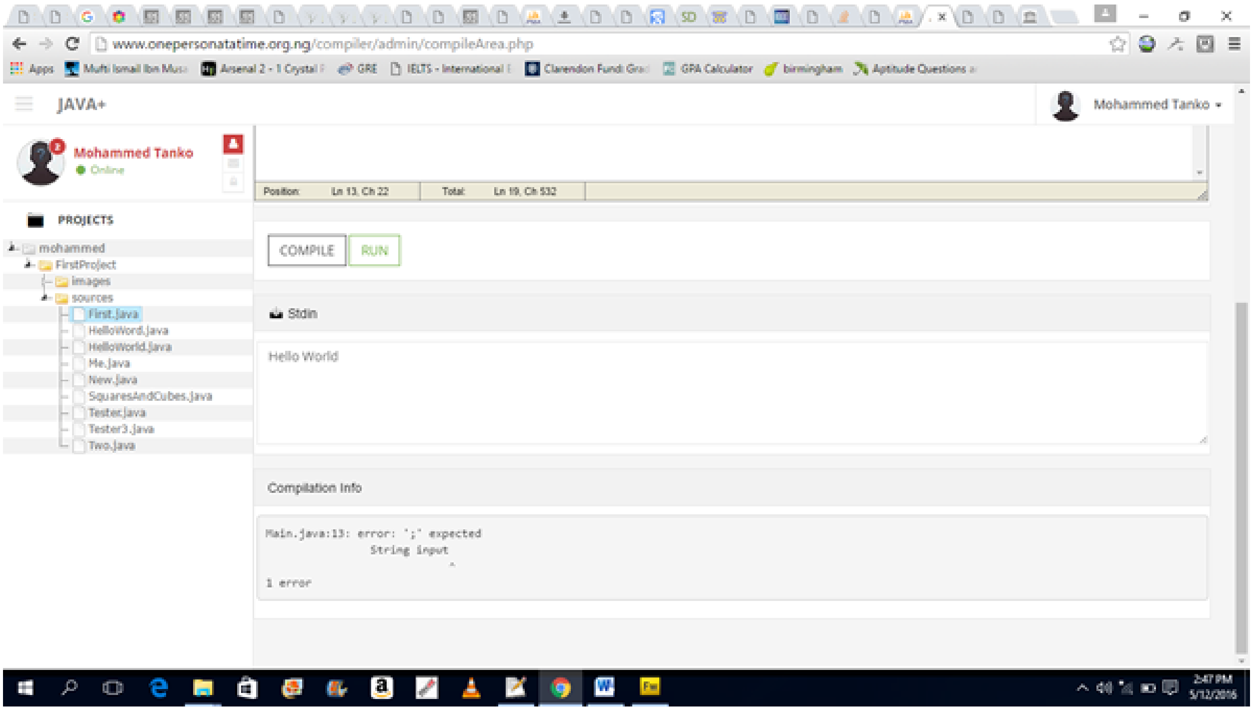


```
/* package whatever; // don't place package name! */

import java.util.*;
import java.lang.*;
import java.io.*;

/* Name of the class has to be "Main" only if the class is public. */
class HelloWorld{
        public static void main (String[] args) throws java.lang.Exception{
                // your code goes here
                public static void main (String[] args) throws java.lang.Exception{
                // your code goes here

                BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

                String input;
                        //input = "hello World";
                while ((input = br.readLine()) != null) {
                    System.out.println(input + " ");
                }
        }
}
```

Position: Ln 19, Ch 43 Total: Ln 22, Ch 622

Compilation Error!

A PROGRAM WITH COMPILATION ERROR



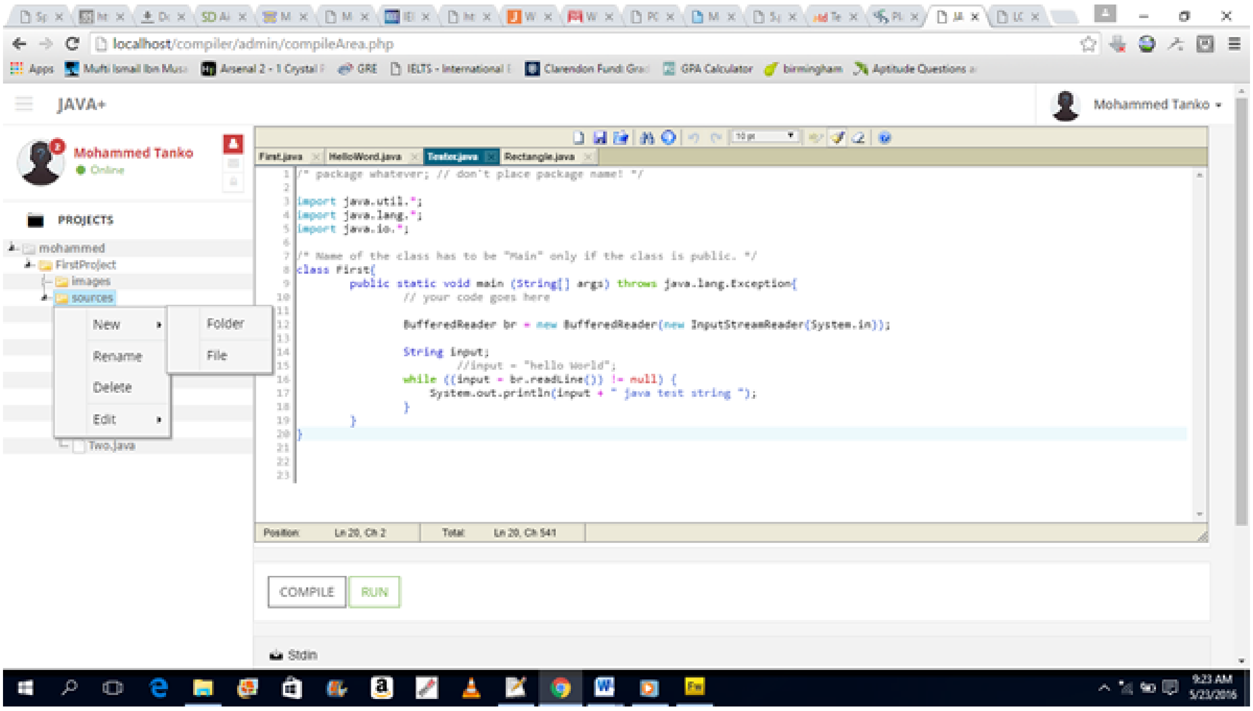
Position: Ln 13, Ch 22 Total: Ln 19, Ch 532

Hello World

Compilation Info

```
Main.java:13: error: ';' expected
                String input
                            ^
1 error
```

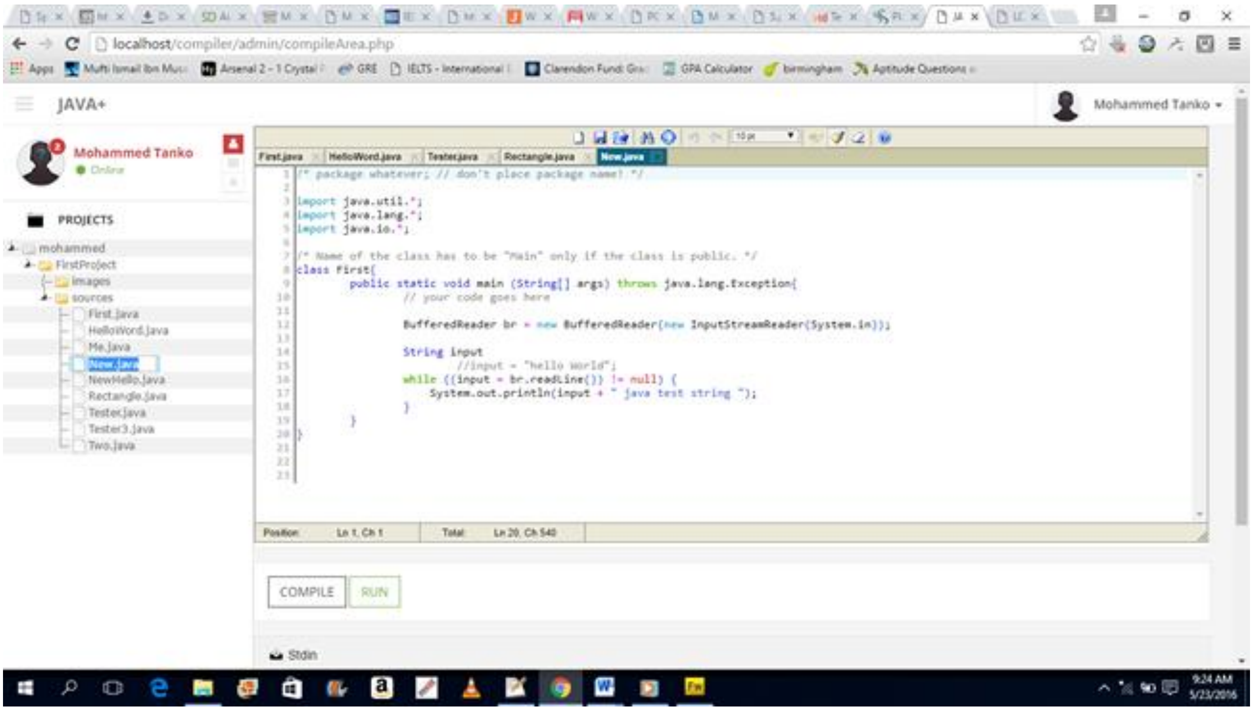
COMPILATION DETAILS OF PROGRAM WITH ERROR



```
/* package whatever; // don't place package name! */

import java.util.*;
import java.lang.*;
import java.io.*;

/* Name of the class has to be "Main" only if the class is public. */
class First{
        public static void main (String[] args) throws java.lang.Exception{
                // your code goes here

                BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

                String input;
                        //input = "hello World";
                while ((input = br.readLine()) != null) {
                    System.out.println(input + " java test string ");
                }
        }
}
```

Position Ln 20, Ch 2 Total Ln 20, Ch 541

MENU FOR MANAGING FOLDERS AND FILES



```
/* package whatever; // don't place package name! */

import java.util.*;
import java.lang.*;
import java.io.*;

/* Name of the class has to be "Main" only if the class is public. */
class First{
        public static void main (String[] args) throws java.lang.Exception{
                // your code goes here

                BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

                String input
                        //input = "hello World";
                while ((input = br.readLine()) != null) {
                    System.out.println(input + " java test string ");
                }
        }
}
```

Position Ln 1, Ch 1 Total Ln 20, Ch 540

CREATING OR RENAMING A FILE OR FOLDER

Appendix B

Some code Fragments of the developed Android Application and Web Version.

Here is a code fragment from the Android application main activity class called **MainCompilerActivity.java**

```
package com.myapps.mtystyle.javacompiler;
import android.app.Dialog;
import android.app.ProgressDialog;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.Toast;
import java.util.ArrayList;
import java.util.List;
public class MainCompilerActivity extends AppCompatActivity {
    private WebView webView;
    private FoldersAndFilesManager fManager;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main_compiler);

        fManager = new FoldersAndFilesManager(this);

        webView = (WebView) findViewById(R.id.webView);

        webView.setWebViewClient(new MyWebViewClient());
        webView.getSettings().setJavaScriptEnabled(true);

webView.loadUrl("http://www.onepersonatatime.org.ng/compiler/admin");
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.menu_main_compiler, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {

        switch (item.getItemId()){
            case R.id.menu_create_folder:
                fManager.createFolderDialog();
                return true;
            case R.id.menu_rename_folder:
                fManager.renameFolderDialog();
                return true;
            case R.id.menu_delete_folder:
                fManager.deleteFolderDialog();
                return true;
            case R.id.menu_create_file:
                fManager.createFileDialog();
                return true;
            case R.id.menu_rename_file:
                fManager.renameFileDialog();
                return true;
            case R.id.menu_delete_file:
                fManager.deleteFileDialog();
                return true;
            default:
                return super.onOptionsItemSelected(item);
        }
    }

    private class MyWebViewClient extends WebViewClient {
        ProgressDialog progressDialog;
        @Override
        public boolean shouldOverrideUrlLoading(WebView view, String url) {
            if
(!url.contains("http://www.onepersonatatime.org.ng/compiler/admin")) {
                view.loadUrl(url);
                return true;
            }
            return super.shouldOverrideUrlLoading(view, url);
        }
        public void onLoadResource (WebView view, String url) {
            if (progressDialog == null) {
                progressDialog = new
ProgressDialog(MainCompilerActivity.this);
                progressDialog.setMessage("Loading...");
                progressDialog.show();
            }
        }
        public void onPageFinished(WebView view, String url) {
            try{
                //work on this later
                progressDialog.dismiss();
                if(progressDialog.isShowing()) {
                    progressDialog.dismiss();
                    progressDialog = null;
                }
            }catch(Exception exception){
                exception.printStackTrace();
            }
        }
    }
}
```

The file **filehandler.php** handles database file creation, deletion, modification on the server side.

```
<?php
session_start();
```

```
require_once('includes/functions.php');
date_default_timezone_set('Africa/Lagos');
//echo $_POST['path'];
if($_POST){
    if(isset($_POST['rename']) && isset($_POST['path']) &&
isset($_POST['newName'])){
        //echo $_POST['path'];
        $folders = explode('/', $_POST['path']);
        $folder_level = count($folders) - 1;
        $oldName = $folders[count($folders)-1];
        $folder_name = $folders[count($folders)-2];
        $rf = $_SESSION['root_folder'];

        $qF = "SELECT root_folder_id FROM root_folders_tbl WHERE
root_folder_name = '".$rf."'";
        $rqF = mysql_query($qF, $link);
        if(mysql_num_rows($rqF) == 1){
            $row = mysql_fetch_array($rqF);
            $rfID = $row['root_folder_id'];

            $newName = filter($_POST['newName']);
            $query = "UPDATE codes_tbl SET code_name = '".$newName."' WHERE
codes_tbl.code_name = '".$oldName."' AND codes_tbl.folder_name =
'".$folder_name."' AND codes_tbl.folder_level = '".$folder_level."' AND
codes_tbl.root_folder_id = '".$rfID."'";

            $result = mysql_query($query, $link) or die(mysql_error());

            if(mysql_affected_rows($link)==1){
                echo 1;
            } else {
                echo $result + "";
            }
        } else{
            echo 3;
        }
    }

    if(isset($_POST['create']) && isset($_POST['path']) &&
isset($_POST['name'])){
        //echo $_POST['path'];
        $folders = explode('/',$_POST['path']);
        $rootFolder = $_SESSION['root_folder'];
        $folder_level = count($folders);
        $folder_name = $folders[count($folders)-1];
        $name = filter($_POST['name']);
        //$errors = array();
        $data = "";
        $qF = "SELECT root_folder_id FROM root_folders_tbl WHERE
root_folder_name = '".$rootFolder."'";
        $rqF = mysql_query($qF, $link);
        if(mysql_num_rows($rqF) == 1){
            $row = mysql_fetch_array($rqF);
            $rfID = $row['root_folder_id'];
            $date = date('d-m-y H:i:s');
            $iCQuery = "INSERT INTO codes_tbl VALUES('','$name', '$rfID',
'$folder_name', '$folder_level','$date')";
            $riCQuery = mysql_query($iCQuery, $link);
            if(mysql_affected_rows($link) == 1){
                $code_id = mysql_insert_id();
                $qry = "INSERT INTO root_folder_codes_tbl VALUES('',
'$rfID', '$code_id')";
                $rslt = mysql_query($qry, $link);
                if(mysql_affected_rows($link) == 1){
                    $data = 1;
                } else {
                    //$errors[] = "problem storing root folder code";
                    $data = 2;
                }
            }else{
                //$errors[] = "problem storing code";
                $data = 3;
            }
        }else {
            //$errors[] = "root folder not found";
            $data = 4;
        }

        echo $data;
    }

    if(isset($_POST['delete']) && isset($_POST['filename'])){
        //echo $_POST['filename'];

        $folders = explode('/', $_POST['filename']);
        //$oldName = $folders[count($folders)-1];
        $folder_name = $folders[count($folders)-2];
        $folder_level = count($folders) - 1;
        $rf = $_SESSION['root_folder'];
        $outData = "";
        $qF = "SELECT root_folder_id FROM root_folders_tbl WHERE
root_folder_name = '".$rf."'";
        $rqF = mysql_query($qF, $link);
        if(mysql_num_rows($rqF) == 1){
            $row = mysql_fetch_array($rqF);
            $rfID = $row['root_folder_id'];
            $name = $folders[count($folders)-1];
            $query = "DELETE FROM codes_tbl WHERE codes_tbl.code_name =
'".$name."' AND codes_tbl.folder_name = '".$folder_name."' AND
codes_tbl.folder_level = '".$folder_level."' AND codes_tbl.root_folder_id =
'".$rfID."'";

            $result = mysql_query($query, $link) or die(mysql_error());
            if(mysql_affected_rows($link)==1){
                $outData = 1;
            } else {
                $outData = $result. mysql_affected_rows($link);
            }
        } else {
            $outData = 3;
        }
        echo $outData;
    }
}
?>
```

SphereEngine.php handles program submission to the Sphere Engine Online API

```
<?php
require_once('includes/functions.php');
if($_POST){
    if(isset($_POST['runCode']) && isset($_POST['sourceCode'])  &&
$_POST['runCode'] == 1){
        $spEngine = new SphereEngine();
        $input = $_POST['stdInput'];
        echo $spEngine->submitCode($_POST['sourceCode'], $input);
    }else if(isset($_POST['fetchSubmission']) &&
isset($_POST['submissionID']) && $_POST['fetchSubmission'] == 1){
        $submissionID = filter($_POST['submissionID']);
        $spEngine = new SphereEngine();
        echo $spEngine->fetchSubmission($submissionID);
    }else{
        echo 9876;
    }
}

class SphereEngine{

    public function __construct() {

    }

    public function submitCode($source, $input){
        $url = 'http://api.compilers.sphere-
engine.com/api/v3/submissions?access_token=2bf91fed33b7fbc3255b7ef37579ed16
';
        $fields = json_encode(array(
        'language' => 10,
        'sourceCode' => $source,
        'input' => $input
        ));

        $headers= array('Content-Type: application/json');

        $ch = curl_init();
        curl_setopt($ch,CURLOPT_URL, $url);
        curl_setopt($ch,CURLOPT_POSTFIELDS, $fields);
        curl_setopt( $ch, CURLOPT_RETURNTRANSFER, true );
        curl_setopt($ch, CURLOPT_HTTPHEADER, $headers);

        $result = curl_exec($ch);

        curl_close($ch);

        return $result;
    }

    public function fetchSubmission($submissionID){
        $base = "http://api.compilers.sphere-
engine.com/api/v3/submissions/";
        $url =
$base.$submissionID."?access_token=2bf91fed33b7fbc3255b7ef37579ed16&withCmp
info=true&withOutput=true&withStderr=true";

        $ch = curl_init();
        curl_setopt($ch,CURLOPT_URL, $url);
        curl_setopt( $ch, CURLOPT_RETURNTRANSFER, true );

        $result = curl_exec($ch);
        curl_close($ch);

        return $result;
    }
}


?>
```

handleSubmission.php handles program compilation and running based on the previous time the program was compiled or executed

```
<?php
require_once('includes/functions.php');
date_default_timezone_set('Africa/Lagos');
    if($_POST){
        if(isset($_POST['code']) && isset($_POST['folder_level']) &&
isset($_POST['getSub']) && $_POST['getSub'] == 1 &&
isset($_POST['folder_name'])){
            $code = filter($_POST['code']);
            $folder_level = filter($_POST['folder_level']);
            $folder_name = filter($_POST['folder_name']);

            $qry = "SELECT code_id FROM codes_tbl WHERE
codes_tbl.code_name='".$code."' AND
codes_tbl.folder_name='".$folder_name."' AND
codes_tbl.folder_level='".$folder_level."'";
            $rslt = mysql_query($qry) or die(mysql_error());

            if(mysql_num_rows($rslt)==1){
                $row = mysql_fetch_array($rslt);
                $code_id = $row['code_id'];
                $query = "SELECT * FROM  codes_tbl,
submitted_codes_tbl,code_submission_tbl WHERE codes_tbl.code_id =
'".$code_id."' AND submitted_codes_tbl.code_id = '".$code_id."' AND
submitted_codes_tbl.code_submission_id =
code_submission_tbl.code_submission_id ORDER BY
submitted_codes_tbl.date_submitted DESC";
                $result = mysql_query($query, $link);
                if(mysql_num_rows($result) == 0){
                    $data = 1;
                }else if(mysql_num_rows($result) >= 1){
                    $row = mysql_fetch_array($result);
                    $data = $row['submission_id'];
                } else {
                    $data = 2. "me";
                }
            }else{
                $data = 4;
            }

            echo $data;
        }

        if(isset($_POST['submissionID']) && isset($_POST['code']) &&
isset($_POST['folder_level']) && isset($_POST['folder_name'])){
            $submissionID = filter($_POST['submissionID']);
            $code = filter($_POST['code']);
            $folder_level = filter($_POST['folder_level']);
            $folder_name = filter($_POST['folder_name']);

            $cquery = "SELECT code_id FROM codes_tbl WHERE code_name =
'".$code."' AND folder_name = '".$folder_name."' AND
codes_tbl.folder_level='".$folder_level."'";
            $rcquery = mysql_query($cquery, $link);
            if(mysql_num_rows($rcquery) == 0){
                $out = 3;
            } else if(mysql_num_rows($rcquery) == 1){
                $row = mysql_fetch_array($rcquery);
                $code_id = $row['code_id'];
                $subQ = "INSERT INTO code_submission_tbl VALUES('',
'$submissionID')";
                $rsubQ = mysql_query($subQ, $link);

                if(mysql_affected_rows($link) == 1){
                    $subID = mysql_insert_id();
                    $date = date('d-m-y H:i:s');
                    $codeSubQ = "INSERT INTO submitted_codes_tbl VALUES('',
'$code_id', '$subID', '$date')";
                    $rcodeSubQ = mysql_query($codeSubQ, $link);

                    if(mysql_affected_rows($link) == 1){
                        $out = 1;
                    }else{
                        $out = 5;
                    }

                } else {
                    $out = 4;
                }


            } else {
                $out = 2;
            }
            echo $out;
        }

    }
?>
```

JQUERY function to open file in editor

```
function loadFile(fname, content, folder){
            //alert(folder);
            var folders = folder.toString().split('/');
            var folder_level = folders.length;
            var folder_name = folders[folders.length - 1];
            var fileExt = fname.toString().split('/');
            var num = fileExt.length;
            var fileName = fileExt[num-1].split('.');
            var new_file= {id: fileName[0]+"", text: content+"", syntax:
'Java', title: fileExt[num-1]+"", folderLevel: folder_level+"", folderName:
folder_name+"", path: folder+""};
            editAreaLoader.openFile('textarea_1', new_file);
        }
```

JQUERY AJAX function to request to save changes to a file

```
function doSaveChanges(data){
            $.ajax({
                type: "POST",
                url: "do_actions.php",
                data: data,
                success: function(result){
                    //alert(result);
                    if(result==1){
                        $('#errSuccessContent').addClass('alert-
success').html("<button type=\"button\" class=\"close\" data-
dismiss=\"alert\">&times;</button><strong>Saved!</strong> Changes
successfully saved.");

$('#errSuccessDiv').attr('display','block').slideDown(500);

                            setTimeout(function() {

$('#errSuccessDiv').attr('display','none').slideUp(500);
                                $('#errSuccessContent').removeClass('alert-
success',5000).html('&nbsp;');
                        }, 5000 );
                    }else if(result==2){
                        $('#errSuccessContent').addClass('alert-
danger').html("<button type=\"button\" class=\"close\" data-
dismiss=\"alert\">&times;</button><strong>Error!</strong> changes could not
be saved, try again later.");

$('#errSuccessDiv').attr('display','block').slideDown(500);

                            setTimeout(function() {

$('#errSuccessDiv').attr('display','none').slideUp(500);
                                $('#errSuccessContent').removeClass('alert-
danger',5000).html('&nbsp;');
                        }, 5000 );
                    }
                },
                error: function(error) {
                    alert("error" + error.status + "occured");
                }
            });
        }
```

JQUERY AJAX REQUEST FUNCTION TO SUBMIT CODE FOR COMPILATION OR RUNNING

```
function submitCode(runCompile){
                var curFile1 = editAreaLoader.getCurrentFile("textarea_1");
                var content1 = curFile1.text;
                var file_name1 = curFile1.id + ".Java";
                var folder_level1 = curFile1.folderLevel;
                var folderName =  curFile1.folderName;
                if(content1 == ""){
                    //alert("empty");
                    $('#err-btn').attr('display','inline').show();
                    $('#err-span').html('load a file into editor before
executing').attr('display','inline').show();
                    setTimeout(function() {
                            $('#err-btn').attr('display','none').hide();
                            $('#err-span').html('&nbsp;').hide();
                    }, 5000 );
                }else{
                    $('#loader').attr('display','inline').show();
                    var a = $('#stdInput').val();
                    var stdin = $.trim(a);
                    var sourceCode = encodeURIComponent(content1);
                    data =
'runCode=1'+'&sourceCode='+sourceCode+'&stdInput='+stdin;
                    $.ajax({
                        type: "POST",
                        url: "sphereEngine.php",
                        data: data,
                        success: function(result){
                            d = $.parseJSON(result);
                            if(d===null){
                                $('#loader').attr('display','none').hide();
                                $('#err-
btn').attr('display','inline').show();
                                $('#err-span').html('Operation failed! try
again later').attr('display','inline').show();
                                setTimeout(function() {
                                        $('#err-
btn').attr('display','none').hide();
                                        $('#err-
span').html('&nbsp;').hide();
                                }, 5000 );
                            }else{
                                var submissionID = d.id;
                                insertSubmission(submissionID, file_name1,
folder_level1, folderName);
                                fetchSubmission(submissionID, runCompile);
                            }
                        },
                        error: function(error) {
                            alert("error " + error.status + "occured");
                        }
                    });

                }
            }
```

JQUERY AJAX REQUEST FUNCTION TO INSERT SUBMISSION DETAILS INTO THE DATABASE

```
function insertSubmission(submissionID, code, folder_level, folder_name){
                data =
'submissionID='+submissionID+'&code='+code+'&folder_level='+folder_level+'&
folder_name='+folder_name;
                var outResult;
                $.ajax({
                    type: "POST",
                    url: "handleSubmissions.php",
                    async: false,
                    data: data,
                    success: function(result){
                        //successful insertion
                        outResult;
                    },
                    error: function(error) {
                        alert("error " + error.status + "occured");
                        outResult = 0;
                    }
                });
            }
```

JQUERY AJAX REQUEST FUNCTION TO FETCH SUBMISSION DETAILS FROM CLOUD SERVER

```
function fetchSubmission(submissionID, runCompile){
                data = 'fetchSubmission=1'+'&submissionID='+submissionID;
                if(runCompile == 2){
                    var btn = "compile";
                }else{
                    var btn = "run";
                }
                $.ajax({
                    type: "POST",
                    url: "sphereEngine.php",
                    data: data,
                    success: function(resultReturned){
                        if(resultReturned===null || resultReturned == ""){
                            $('#loader').attr('display','none').hide();
                            $('#err-btn').attr('display','inline').show();
                            $('#err-span').html('Operation failed! try
again later').attr('display','inline').show();
                            setTimeout(function() {
                                    $('#err-
btn').attr('display','none').hide();
                                    $('#err-span').html('&nbsp;').hide();
                            }, 5000 );
                        } else {
                                var dt = $.parseJSON(resultReturned);
                                if(dt.status!= 0){

                                    var curFile =
editAreaLoader.getCurrentFile("textarea_1");
                                    var content = curFile.text;
                                    var file_name = curFile.id + ".Java";
                                    var folder_level = curFile.folderLevel;
                                    var folder_name = curFile.folderName;
                                    var d = getLastSubID(file_name,
folder_level, folder_name);
                                    setTimeout(function() {
                                            fetchSubmission(d, runCompile);
                                    }, 1000 );
                                }else{
                                    switch(dt.result){
                                        case 11:

$('#loader').attr('display','none').hide();

//$("#"+btn).attr('disabled','false');
                                            $('#err-
btn').attr('display','inline').show();
                                            $('#err-
span').html('Compilation Error!').attr('display','inline').show();
                                            setTimeout(function() {
                                                    $('#err-
btn').attr('display','none').hide();
                                                    $('#err-
span').html('&nbsp;').hide();
                                            }, 5000 );

$('#compInfo').html("<pre>"+dt.cmpinfo+"</pre>");

$('#compInfoDiv').attr('display', 'block').show();
                                            break;

                                        case 12:
                                            //alert("12 happened");
                                            break;
                                        case 13:
                                            //alert("13 happened");
                                            break;
                                        case 15:


$('#loader').attr('display','none').hide();

//$("#"+btn).attr('disabled','false');
                                            $('#succ-
btn').attr('display','inline').show();
                                            $('#succ-
span').html('Success!').attr('display','inline').show();
                                            setTimeout(function() {
                                                    $('#succ-
btn').attr('display','none').hide();
                                                    $('#succ-
span').html('&nbsp;').hide();
                                            }, 5000 );
                                            var out = "";
                                            if(runCompile === 1){
                                                out = "<span style =
\"color: green\">Succesful</span>";
                                            } else {
                                                out = "<span style =
\"color: green\">Succesfully compiled</span>";
                                            }
                                            $('#compInfo').html("<pre>" +
out +"</pre>");

$('#compInfoDiv').attr('display', 'block').show();
                                            if(runCompile === 1){

$('#outputData').html("<pre>"+dt.output+"</pre>");

$('#outputDiv').attr('display','block').show();
                                            }
                                            break;
                                        case 17:
                                            //alert("17 happened");
                                            break;
                                        case 19:
                                            //alert("19 happened");
                                            break;
                                        case 20:
                                            //alert("20 happened");
                                            break;
                                    }
                                    //alert(89);
                                }
                            //}


                        }

                    },
                    error: function(error) {
                        alert("error " + error.status + "occured");
                    }
                });
            }
```