**ABSTRACT**

The number of smart phone users and mobile applications are growing rapidly. Though smart phones are expected to have PC-like functionality, hardware resources such as CPUs, memory and batteries are still limited .To solve this resource problem, connect the phones to adjacent powerful cloud servers to throw their computational burden to the servers ,many researchers have proposed architectures to use server resources in the cloud for mobile Computing devices.

To solve this resource problem, many researchers have proposed architectures to use server resources in the cloud for mobile Computing devices. We propose a conceptual architecture of android as a server Platform, which enables user Android applications on cloud server via network.

Android is mainly designed for physical smart phone, Android’s two other features are useful to construct a server platform. In our project we proposed a system that without installing software in mobile device accessing that software through cloud server. In our project we are using java software in server and accessing that software through the mobile device, improves the performance of our mobile cloud computing significantly in terms of execution time and energy consumption.

There are two key tasks involved before remote execution: code partitioning and state migration. And we are using software as a service. SAAS is a software delivery method that provides access to software and its functions remotely as a web-based service.

|  |  |  |
| --- | --- | --- |
|  | **TABLE OF CONTENTS** |  |
| **CHAPTER** | **TITLE** | **PAGE NO** |
| **I.** | **ABSTRACT** | **iv** |
| **II.** | **LIST OF TABLES** | **xi** |
| **III.** | **LIST OF FIGURES** | **x** |
| **IV.** | **LIST OF ABBREVIATIONS** | **xi** |
|  |  |  |
| **1.** | **INTRODUCTION** | **1** |
|  | 1.1 OBJECTIVE OF THE PROJECT | 1 |
|  | 1.2 OVERVIEW OF THE PROJECT | 1 |
|  | 1.3 EXISTING SYSTEM | 1 |
|  | 1.3.1 Drawbacks | 2 |
|  | 1.4 PROPOSED SYSTEM | 2 |
|  | 1.4.1 Advantages | 2 |
|  | 1.5 LITERATURE SURVEY | 3 |
|  | 1.5.1 Private Cloud With PC | 3 |
|  | 1.5.1.1 Drawbacks | 3 |
|  | 1.5.1.2 Proposed Work | 3 |
|  | 1.5.2 C# Compiler Using Cloud Computing | 4 |
|  | 1.5.2.1 Drawbacks | 4 |
|  | 1.5.2.2 Proposed Work | 4 |
|  | 1.5.3 An Effective C, C++ And Using Cloud Computing | 4 |
|  | 1.5.3.1 Drawbacks | 4 |
|  | 1.5.3.2 Proposed Work | 5 |
|  | 1.5.4 Performance Calculation Using Cloud Computing | 5 |
|  | 1.5.4.1 Drawbacks | 5 |
|  | 1.5.4.2 Proposed Work | 5 |
|  | 1.5.5 Integrated Development Environment | 5 |
|  | 1.5.5.1 Drawbacks | 6 |
|  | 1.5.5.2 Proposed Work | 6 |
| **2.** | **SYSTEM SPECFICATION** | **7** |
|  | 2.1 SOFTWARE REQIREMENTS | 7 |
|  | 2.2 HARDWARE REQIUREMENTS | 7 |
|  | 2.3 EXTERNAL INTERFACE REQUIREMENTS | 7 |
|  | 2.3.1 User Interface | 8 |
|  | 2.3.2 Hardware Interface | 9 |
|  | 2.3.3 Software Interface | 9 |
|  | 2.3.4 Communication Interface | 9 |
|  | 2.4 SYSTEM FEATURES | 9 |
|  | 2.4.1 Software As A Service | 10 |
|  | 2.4.2 Mobility | 11 |
|  | 2.4.3 Working Environment In Application | 11 |
|  | 2.4.4 Debugging Environment | 12 |
|  | 2.5 NON FUNCTIONAL REQIUREMENTS | 13 |
|  | 2.5.1 Performance Requirements | 13 |
|  | 2.5.2 Safety Requirements | 13 |
|  | 2.5.3 Security Requirements | 13 |
|  | 2.5.4 Software Quality Attributes | 14 |
|  | 2.6 ALGORITHM DESCRIPTION | 15 |
|  | 2.6.1 Load Balancing Algorithm | 15 |
| **3.** | **SYSTEM DESIGN** | **16** |
|  | 3.1 SYSTEM DESGIN ARCHITECTURE | 16 |
|  | 3.2 MODULE DESCRIPTION | 17 |
|  | 3.2.1 Mobile Client | 17 |
|  | 3.2.2 Cloud Server | 17 |
|  | 3.2.3 Software As A Service | 18 |
|  | 3.3 DATA DESIGN | 19 |
|  | 3.3.1 Data Description | 19 |
|  | 3.3.2 Data Dictionary | 19 |
|  | 3.4 COMPONENT DESIGN | 19 |
|  | 3.4.1 Authentication | 19 |
|  | 3.4.2 Implementation Of Software As A Service | 20 |
|  | 3.4.3 Working Environment In Application | 20 |
|  | 3.4.4 Debugging Environment | 20 |
|  | 3.5 HUMAN INTERFACE DESIGN | 20 |
|  | 3.5.1 Overview Of Interface | 20 |
|  | 3.5.2 Screen Objects And Its Action  3.6 DFD DIAGRAM | 21 |
|  | 3.7 USE-CASE DIAGRAM | 26 |
|  | 3.8 CLASS DIAGRAM | 27 |
|  | 3.9 SEQUENCE DIAGRAM | 28 |
|  | 3.10 ACTIVITY DIAGRAM | 29 |
| **4.** | **IMPLEMENTAION AND TESTING** | **30** |
|  | 4.1 CODING | 31 |
|  | 4.2 TEST PLAN | 38 |
|  | 4.2.1 UNIT TESTING | 38 |
|  | 4.2.2 INTEGRATION TESTING | 39 |
|  | 4.2.3 VALIDATION TESTING | 40 |
|  | 4.3 TESTING STRATE | 40 |
|  | 4.3.1 INTEGRATION TESTING | 42 |
|  | 4.3.2 WHITE BOX TESTING | 42 |
|  | 4.3.3 INTERFACE TESTING | 42 |
|  | 4.3.4 MODULE TESTING | 42 |
|  | 4.3.5 SMOKE TESTING | 42 |
|  | 4.4 MAINTANENCE | 43 |
| **5.** | **CONCLUSION AND FUTURE WORK** | **44** |
|  | 5.1 CONCLUSION | 44 |
|  | 5.2 FUTURE WORK | 44 |
|  | **APPENDICES** | **45** |
|  | **REFERENCES** | **55** |

|  |  |  |
| --- | --- | --- |
|  | **LIST OF TABLES** |  |
| **TABLE NUMBER** | **TABLE NAME** | **PAGE NO** |
| **5.1** | **Unit Testing** | **39** |
| **5.2** | **Integration Testing** | **40** |
| **5.3** | **Validation Testing** | **41** |
|  |  |  |

|  |  |  |
| --- | --- | --- |
|  | **LIST OF FIGURES** |  |
| **FIGURE** | **FIGURE NAME** | **PAGE NO** |
| **3.1** | **System Architecture** | **16** |
| **3.2** | **Mobile Client** | **17** |
| **3.3** | **Cloud Server** | **18** |
| **3.4** | **Software As A Service** | **18** |
| **3.5** | **Create Java Program** | **22** |
| **3.6** | **Compile Java Program** | **23** |
| **3.7** | **Debugging Interface** | **24** |
| **3.8** | **Running Program** | **25** |
| **3.9** | **Use Case Diagram** | **26** |
| **3.10** | **Class Diagram** | **27** |
| **3.11** | **Sequence Diagram** | **28** |
| **3.12** | **Activity Diagram** | **29** |
| **A1** | **Register Page** | **45** |
| **A2** | **Service Options** | **46** |
| **A3** | **Creating A File** | **47** |
| **A4** | **Compiling A File** | **48** |
| **A5** | **Execution And Output** | **49** |
| **A6** | **Debugging Environment** | **50** |
| **A7** | **Online Compiler** | **51** |
| **A8** | **Source Code** | **52** |
| **A9** | **Compile Code** | **53** |
| **A10** | **Run code** | **54** |

|  |  |
| --- | --- |
| **LIST OF ABBREVATIONS** | |
| SAAS | SOFTWARE AS A SERVICE |
| JDK | JAVA DEVELOPMENT KIT |
| CODE DOM | CODE DOCUMENT OBJECT |
| IDE | INTEGRATED DEVELOPMENT ENVIRONMENT |
| URL | UNIVERSAL RESOURCE LOCATOR |
| GPRS | GENERAL PACKET RADIO SERVICE |
|  | HYPERTEXT PREPROCESSOR |
| HTTP | HYPERTEXT TRANSFER PROTOCOL |

**CHAPTER 1**

**INTRODUCTION**

1. **INTRODUCTION**

World is contracting with the growth of mobile phone technology. As the number of users is increasing day by day, facilities are also increasing. Starting with simple regular handsets which were used just for making phone calls, mobiles have changed our lives and have become part of it. Now they are not used just for making calls but they have innumerable uses and can be used as a Camera, Music player, Tablet PC, T.V. , Web browser etc. And with the new technologies, new software and operating systems are required. [Operating Systems](http://www.engineersgarage.com/articles/operating-systems-tutorial) have developed a lot in last 15 years. Starting from black and white phones to recent smart phones or mini computers, mobile OS has come far away. Especially for smart phones, Mobile OS has greatly evolved from Palm OS in 1996 to Windows pocket PC in 2000 then to Blackberry OS and Android.

One of the most widely used mobile OS these days is **ANDROID**. **Android** is a software bunch comprising not only operating system but also middleware and key applications. Android Inc was founded in Palo Alto of California, U.S. by Andy Rubin, Rich miner, Nick sears and Chris White in 2003. Later Android Inc. was acquired by Google in 2005. After original release there have been number of updates in the original version of Android. These are the basics of Android applications:

Android applications are composed of one or more application components (activities, services, content providers, and broadcast receivers). Each component performs a different role in the overall application behavior, and each one can be activated individually (even by other applications). The manifest file must declare all components in the application and should also declare all application requirements, such as the minimum version of Android required and any hardware configurations required. Non-code application resources (images, strings, layout files, etc.) should include alternatives for different device configurations (such as different strings for different languages)

 Google, for software development and application development, had launched two competitions ADC1 and ADC2 for the most innovative applications for Android. It offered prizes of USD 10 million combined in ADC1 and 2. ADC1 was launched in January 2008 and ADC 2 was launched in May 2009. These competitions helped Google a lot in making Android better, more user friendly, advanced and interactive. Android is the world's most popular operating system for mobile devices and tablets. It is an open source operating system, created by Google, and available to all kinds of developers with various expertise levels, ranging from rookie to professional.

From a developer's perspective, Android is a Linux-based operating system for smartphones and tablets. It includes a touch screen user interface, widgets, camera, network data monitoring and all the other features that enable a cell phone to be called a smartphone. Android is a platform that supports various applications, available through the Android Play Store. The Android platform also allows end users to develop, install and use their own applications on top of the Android framework. The Android framework is licensed under the Apache License, with Android application developers holding the right to distribute their applications under their customized license. Cloud computing has taken the IT world by storm. There are various layers to the Android programming model that easily fit in with the creation of secure applications specially made for the cloud environment. The open source Android operating system allows complex cloud computing applications to run wherever the user is.

Android developers can write applications to take advantage of the cloud and can leverage the faster time to market, the agility, cost benefits, etc. Most of the time, as users, we merely consider games and other apps that simplify daily life as the inspiration for Android apps. But make no mistake; enterprise apps are a good bet too. According to top research analysts, mobile-centric applications and interfaces are among the top 10 strategic technology trends in 2012 and 2013. Now, the question is, for Android app developers, how different is it to develop apps in the traditional environment and the cloud environment?

In the traditional environment, the complete infrastructure needs to be maintained at the back end. Hence, the focus is more on maintaining the environment and not on making applications that are robust and innovative.In the cloud environment, infrastructure is managed by service providers in public clouds. Hardware maintenance is the responsibility of the service provider and, in addition to this, service providers also maintain the software stack.

Cloud Computing is an innovative technology that is revolutionizing the way we do computing. The key concept of cloud computing is that you don't buy the hardware, or even the software, you need anymore, rather you rent some computational power, storage, databases, and any other resource you need by a provider according to a pay-as-you-go model, making your investment smaller and oriented to operations rather than to assets acquisition. Cloud computing is an internet based computing which is enables to provide convenient as well as on-demand network access. Cloud computing provided the SAAS platform. The number of smart phone users and mobile applications are growing rapidly. There are various mobile operating systems, such as android, windows etc. Developers construct many kinds of applications for these platforms and day to day they are increasing but smart phones are not providing pc like functionality because of their limitations like battery power, powerful processer, large size of memory. To overcome these limitations we proposed a conceptual architecture of android as a server platform. It enables user to access the cloud services via network. In this paper, we propose system that without installing software in mobile device accessing that software through cloud server. We can use android as a server platform. Android is an open source mobile OS and it enables many users to use resources on remote cloud server. Here we are using android as a server platform because android able to run not for smart phone but also for x86 platform. Cloud computing is the upcoming area in the real world. It provides the cloud services, but to utilize this cloud computing resources computer like hardware is required. Smart phone have less functionality than pc’s, because hardware resources are limited. Cloud computing is not easy to manage through mobiles. Managing the cloud computing through mobile is not easy job till now. Using mobile internet connection it is not easy to connect with remote network. Cloud integrative mobile applications are not in use. Because of all these things user faces many problems regarding the cloud services. So our problem statement is that to implements Cloud Computing Architecture for Mobile Devices. Android User can utilize software as a service Process from the cloud server, without installing the software in the user Android mobile.

Cloud computing means using web applications and/or server services that you pay to access rather than software or hardware that you buy and install.  Cloud computing is a model for delivering information technology services, whose resources are retrieved from the internet through web-based applications and tools. Cloud computing is so named because of the information being accessed is found in clusters that are available globally, by which not requiring the user to be in a specific place to gain access to it.  The data that are accessible are provided by data centres all over the world, which are collectively called as clouds.

Mobile Computing is using a computer (of one kind or another) while on the move. Mobile Computing is when a (work) process is moved from a normal fixed position to a more dynamic position. Mobile Computing is when a work process is carried out somewhere where it was not previously possible. Mobile Computing is an umbrella term used to describe technologies that enable people to access network services anyplace, anytime, and anywhere.

Mobile computing is taking a computer and all necessary files and software out into the field. It is being able to use a computing device even when being mobile and therefore changing location. Portability is one aspect of mobile computing.

* Mobile computing is human–computer interaction by which a computer is expected to be transported during normal usage. Mobile computing involves mobile communication, mobile hardware, and mobile software.
* Mobile computing is the ability to use computing capability without a pre-defined location and/or connection to a network to publish and/or subscribe to information.
* Communication issues include ad hoc and infrastructure networks as well as communication properties, communications protocols, data formats and concrete technologies.
* Hardware includes mobile device or device components. Mobile software deals with the characteristics and requirements of mobile applications.

Mobile cloud computing (MCC)at its simplest, refers to an infrastructure where both the data storage and data processing happen outside of the mobile device. Mobile cloud applications move the computing power and data storage away from the mobile devices and into powerful and centralized computing platforms located in clouds, which are then accessed over the wireless connection based on a thin native client.

Mobile devices face many resource challenges (battery life, storage, bandwidth etc.) Cloud computing offers advantages to users by allowing them to use infrastructure, platforms and software by cloud providers at low cost and elastically in an on-demand fashion.

* Mobile cloud computing provides mobile users with data storage and processing services in clouds, obviating the need to have a powerful device configuration (e.g. CPU speed, memory capacity etc), as all resource-intensive computing can be performed in the cloud.
* Mobile cloud computing is a highly promising trend for the future of mobile computing.
* Mobile devices are connected to the mobile networks via base stations that establish and control the connections and functional interfaces between the networks and mobile devices.
* Mobile users’ requests and information are transmitted to the central processors that are connected to servers providing mobile network services.
* The subscribers’ requests are delivered to a cloud through the Internet.
* In the cloud, cloud controllers process the requests to provide mobile users with the corresponding cloud services.
* MCC enables mobile users to store/access large data on the cloud.
* MCC helps reduce the running cost for computation intensive applications.
* Mobile applications are not constrained by storage capacity on the devices because their data now is stored on the cloud.

The mobile cloud is Internet-based data, applications and related services accessed through smart phones, laptop computers, tablets and other portable devices.

Mobile cloud computing is differentiated from mobile computing in general because the devices run cloud-based [Web apps](http://searchsoftwarequality.techtarget.com/definition/Web-application-Web-app) rather than [native apps](http://searchsoftwarequality.techtarget.com/definition/native-application-native-app). Users subscribe to cloud services and access remotely stored applications and their associated data over the Internet. Typically, mobile devices run a mix of Web-based and native apps. However, the trend is increasingly toward the mobile cloud. According to ABI Research, the number of mobile cloud computing subscribers is expected to reach 998 million by 2014.

A mobile cloud application is a software program that is designed to be accessed over the Internet by many types of portable computing devices. Mobile [cloud apps](http://searchcloudapplications.techtarget.com/definition/cloud-application) and mobile [Web apps](http://searchsoftwarequality.techtarget.com/definition/Web-application-Web-app) are similar. They both run on server external to the mobile device, they both store [data](http://searchdatamanagement.techtarget.com/definition/data) externally and they are both accessed over the Internet with a [browser](http://searchwindevelopment.techtarget.com/definition/browser). However, it is often said that while all cloud apps are Web apps -- not all Web apps are cloud apps. Simply put, not all mobile Web apps can run in a [virtual environment](http://searchservervirtualization.techtarget.com/definition/virtualization) without being re-engineered. This is because a Web app may have originally been written to run and store data on a dedicated physical server in a data centre. A cloud app, on the other hand, will always be written to live on [virtual servers](http://searchnetworking.techtarget.com/definition/virtual-server) in a distributed, [multi-tenant](http://whatis.techtarget.com/definition/multi-tenancy) architecture and store data in the cloud.

Mobile cloud and Web apps are both very different from native mobile apps. [Native apps](http://searchsoftwarequality.techtarget.com/definition/native-application-native-app) in mobile software development run on one particular mobile device or platform and are downloaded and installed on the mobile device. The challenge of writing native mobile apps is that developers must create three different versions of the same mobile app if they want it to be used by [iOS](http://searchconsumerization.techtarget.com/definition/iOS), [Android](http://searchenterpriselinux.techtarget.com/definition/Android) and [Windows](http://searchwindowsserver.techtarget.com/definition/Windows) devices.  Because mobile cloud apps are not downloaded, developers can just write one version of their mobile app and any device with a browser and Internet connection can use it. The challenge becomes writing and managing application programming interfaces ([APIs](http://searchexchange.techtarget.com/definition/application-program-interface)) that use [loosely coupled](http://searchnetworking.techtarget.com/definition/loose-coupling) cloud services in the most cost-effective manner.

The NIST defines cloud computing as a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction. It has three layers of services, namely,

* Software as a Service (SaaS)
* Platform as a Service (PaaS)
* Infrastructure as a Service (IaaS)

In SaaS, it provides software or applications for users to access the application that will be run in the cloud. So users simply use the applications without concerning system configuration problems. In the PaaS layer, users can choose preferable operating systems and develop personal software by using supported resources in the cloud. For instance, users can store their data in the cloud and send any queries to the cloud whenever they need to retrieve information.

In the IaaS layer, users can setup personal operating systems, configure computation environment, and develop software. The cloud provides a powerful processing core and a massive storage space with configurable computing resources for users to do computation on it. By this architecture model, from top to down, users can have more control on the available computing resources. Cloud service is characterized as on-demand, elastic, quality of service guaranteed, and pay-per-use. With the advancement of wireless infrastructure, mobile device can connect to the cloud any place and any time.

Cloud computing is especially an enabler for the bring your own device (BYOD) technology permitting employees to bring personally owned mobile devices to their workplace, and use those devices to access privileged enterprise content and applications stored on the cloud. Cloud computing goes hand-in-hand with mobile virtualization which enables multiple operating systems or virtual machines (VMs) to run simultaneously on a mobile device. That is, cloud computing can provide separate services (including applications, user profiles, contacts and data)to completely isolated VM containers running on the same mobile device or smartphone with mobile virtualization. Cloud computing thus advances mobile computing in three major areas: supporting ultrathin mobile devices with mobile virtualization, providing scalable mobile computation, and supporting big data mobile applications. Essentially, cloud computing propels a new class of applications which we call MCC applications extending traditional MC applications with unlimited storage and computation resources as well as task-oriented services. In the literature, there are two definitions for MCC based on the same line of observations discussed above extending MC to MCC. In the first definition, MCC is defined as a computing model combining mobile computing and the cloud, where the cloud can handle large storage and processing for mobile devices remotely. In the second definition, the cloud does not have to be a remote powerful server, but one that advances mobile devices cooperating for storage and processing.

* 1. **OBJECTIVE OF THE PROJECT**

The main objective of the project is to develop a Cloud Computing. Application Setup server and deploy the application on the cloud to test it across the range of learn mobile computing systems. We are implementing Software as a Service (SAAS) for Cloud Computing. Develop Android based application for students.

**1.2 OVERVIEW OF THE PROJECT**

The number of smart phones and mobile applications are growing rapidly. Though smart phones are expected to have PC-like functionality hardware resources such as CPUs, memory and batteries are still limited. To solve this resources problem, many researchers have proposed architectures to use server resources in the cloud for mobile devices. We propose a conceptual architecture of Android as a Server Platform, which enables multiple Android application on cloud server via network. Though Android is mainly designed for physical smart phone.

**1.3 EXISTING SYSTEM**

To compile and run Java program on our computer ,we need to have a working installation of the Java Development Kit (JDK) from Sun Microsystems. Even we compile and run C, C++ program on our computer ,we need working installation of Turbo C. Cloud computing is the upcoming area in the real Networks, but to utilize this cloud computing Resource computer like Hardware is required. Cloud Computing is not easy job to access in our mobile device.

**1.3.1 DRAWBACKS OF EXISTING SYSTEM**

* Managing the Cloud Computing through Mobile is not a easy job till now. They cannot access their Remote Network through GPRS connectivity using Mobile. The Cloud integrative Mobile Application is not in Use.
* One of the drawbacks of Java, C, C++ compiler in
* system is while the small program that many novice programmers code take, larger application suites can take significant amount of time to compile.

**1.4 PROPOSED SYSTEM**

Cloud Computing Application can be initiated using Android Smart Phones. We are implementing Software as a Service (SAAS) for Cloud Computing. SAAS is the Cloud Computing Resource, used for the service of without installing that Software in the Device. Here, we are compiling the code using Android Smart phones without installing Software in Mobile Phone. Implementing cloud computing architecture for mobile devices. Android can utilize software as a service (SAAS) Process from the cloud server, without installing the software in the Android mobile. This features allows students to do Java, C, C++and programming anywhere, anytime using just mobile interface.

**1.4.1 ADVANTAGES OF PROPOSED SYSTEM**

* We are implementing Software as a service (SAAS) for Cloud Computing. SAAS is the Cloud Computing Resource, used for the service of Software without installing that Software in the User Device.
* It allows compiling and executing Java, C, C++ programs directly through the Android mobile so that they can concentrate on the programming concepts rather than learning to operate new technologies (os).
* We will execute small Java, C, C ++ program through Android Mobile it reduce the time consistency.
* This allows students can to do Java, C, C++ programming anywhere, anytime using just mobile interface.

**1.5 ORGANIZATION OF THE PROJECT**

|  |
| --- |
| In this project the chapter 1 deals with the introduction about the outsourcing databases, data owners and various service providers. It discusses about the various location bases available. It also discusses the objective and overview of the project and literature survey. It describes the surveys about the existing system its advantages and disadvantages. In chapter 2 mainly concentrates the software requirements specification. It specifies the software and hardware requirements, external interface requirements and describes about the description of the algorithm. Chapter 3 contains information about the Design of the project, the system architecture its dataflow diagrams, sequence diagrams and module description. Chapter 4 is about the implementation part and the various coding and test plan and all the methodologies are tested using unit testing for accuracy, correctness and completeness. The 7th chapter includes conclusion and information for future enhancement of the project.  **LITERATURE SURVEY**  A Literature Survey is generally an evaluative report of the studies found in the literature related to the selected area. The survey should describe, summarize, evaluate and clarify the literature. It should give a theoretical basis for the research and the surveyors determine the nature of their own research. A limited number of works that are central to the topic of focus must be collected rather than collecting a large number of works that are not closely connected to the topic area which is chosen. The objective of this literature survey are to establish a theoretical framework in the domain of the surveillance, to establish the area in which the system is to be based as well as to identify studies, models, algorithms, techniques and technologies which support the proposed system or are similar to it.  Gayathri V, Priyadarsini K (2014) described about the Mobile Multimedia cloud computing on the web to develop and creasing array of online personal health record (PHR) systems by utilizing the web-based systems and cloud computing infrastructure. In this study, we present a web-based PHR visualization system, called the 3D medical graphical avatar(MGA), which was designed to explore web-based delivery of a wide array of medical data types including multi-dimensional medical images, videos, text-based data and spatial annotations. Mapping information was extracted from eeach of the data types and was used to embed spatial and textual annotations, such as regions of interest (ROIs). As such, further research is necessary to better visualize and present these records, in ways that make the complex medical data more intuitive. Although these system provide the technical capacity to store and retrieve medical data in various multimedia formats, including images, ideas, voice, and text, individual patient use remains limited by the lack of intuitive data representation and visualization techniques.  Mrs.MinalZaa, et al.,(2013)proposed a survey on Migration of Task between Cloud and Mobile Device which combined the working of mobile device and cloud computing so that the task migration can take place with less effort and applications can be utilized easily. This topic helped the purpose of application usage at the same time demonstrate the technique of integration between mobile device and cloud computing which are platform independent, means any mobile can use this application in any heterogeneous environment. New mechanism should be developed in order to reduce the access latency and the overhead during migration.  R. Ma and C.L. Wang (2012) proposed a Lightweight application level task migration for mobile cloud computing that allows mobile applications to use the enormous resources in the cloud. In order to seamlessly utilize the resources, it is common to migrate computation among mobile nodes and cloud nodes. Therefore, a highly portable and transparent migration approach is needed.In the existing system, this approach imposes significant runtime overhead, even when no migration takes place. In this paper, we propose a Java byte code transformation technique for realizing task migration without imposing significant overhead on normal execution. Asynchronous migration technique is used to allow migrations to take place virtually anywhere in the user codes, and the proposed Twin Method Hierarchy minimizes the overhead resulting from state-restoration codes in normal execution.  Bharat Bhargava et al., (2012) describes a survey of Computation Overloading for mobile systems to overcome the restrictions in mobile systems such as battery life, network bandwidth, storage capacity and processor performance. These are alleviated by computation overloading, which sends heavy computation to resourceful servers and receiving the results from these servers. It provides an overview of the background, techniques and research areas for offloading computation. They survey different types of algorithms used to partition and offload programs in order to improve performance or save energy. The techniques which are Virtualization and cloud computing and enabled multitasking: the ability to run multiple applications and processes at the same time. The drawbacks are Under utilization, security & operational costs.  Byung-Gon Chun et al., (2011) presented the design and implementation of CloneCloud, a system that automatically transforms mobile applications to benefit from the cloud. Mobile applications are becoming increasingly ubiquitous and provide eer richer functionality on mobile devices. At the same time, such devices often enjoy strong connectivity with more powerful machines ranging from laptops and desktops to commercial clouds. CloneCloud uses a combination of static analysis and dynamic profiling to partition applications automatically at a fine granularity while optimizing execution time and energy use for a target computation and communication environment. Their evaluation shows that CloneCloud can adapt application partitioning to different environments, and can help some applications achieve as much as a 20x execution speed-up and a 20-fold decrease of energy spent on the mobile device. The disadvantage of the paper is that the CloneClouddoesnot virtualize access to native resources that are not virtualized already and are not available on the clone.  Moo-RyongRa et al., (2011) describes the Odessa: Enabling Interactive Perception Applications on mobile devices to develop and evaluate interactive perceptual applications. Resource constrained mobile devices need to leverage computation on nearby servers to run responsive applications that recognize objects, people, or gestures from real-time video. The two key questions that impact performance are what computation to offload, and how to structure the parallelism across the mobile device and server. To answer these questions, we develop and evaluate three perceptual interactive applications. In this, they developed Odessa, a novel, lightweight, runtime that automatically and adaptively makes offloading and parallelism decisions for mobile interactive perception applications. Their evaluation shows that the incremental greedy strategy of Odessa converges to an operating point that is close to an ideal offline partitioning. It provides more than a 3x improvement in application performance compared to application configurations by domain experts. Odessa can adapt quickly to changes and compute resource availability and network bandwidth and exploring easy deployability on mobile devices by leveraging modern browser architectures.  Damianos Gavalas and Daphne Economou (2011) describes a comparison of four popular runtime environments clarifies the options available for developing applications that run on resource-constrained mobile devices. Mobile devices have steadily gained acceptance as a multimedia platform. Current tools offer application developers options to use various technologies-for example, Java, Open C, Python, Flash Lite, XHTML/CSS, JavaScript, and Mobile Ajax- to implement highly functional mobile applications. They described the general results of this comparison as well as details from the game application’s development. We summarize the results from all sources in a table and conclude with our assessment of how appropriate the different platforms are with respect to critical application-development requirements. As mobile applications become more computationally intense and require faster runtime speeds and storage, performance also becomes a critical issue. Even though market and application requirements largely determine the platform for mobile development, our review offers some specific and general guidance into the assets and deficiencies of available tools as developers face the increasing demand for applications on resource-constrained devices.  Mahadev Satyanarayanan et al.,(2009) proposed the case for VM-Based Cloudlets in Mobile computing. A new vision of mobile computing liberates mobile devices from severe resource constraints by enabling resource-intensive applications to leverage cloud computing free of WAN delays, jitter, congestion, and failures. In this paper, proposed an architecture in which a mobile user exploits virtual machine(VM) technology to rapidly instantiate customized service software on a nearby cloudlet and then uses that service over a wireless LAN; the mobile device typically functions as a thin client with respect to the service. A cloudlet is a trusted, resource-rich computer or cluster of computers that’s well-connected to the Internet and available for use by nearby mobile devices. Although much remains to be done, the concepts and ideas introduced here open the door to a new world of mobile computing in which seamless cognitive assistance for users occurs in diverse ways at any time and place.  Ajay surie et al., proposed Rapid Trust Establishment for Pervasive Personal Computing. Trust-sniffer’s staged approach to establishing confidence in untrusted machines balances security and ease-of-use, facilitating rapid use of transient hardware. In this model, establishing trust in unmanaged hardware for transient use becomes a major issue. Today, when a user sits down at a computer in the office or home, he or she implicitly assumes that the machine hasn’t been tampered with and that it does not contain malware, such as a keystroke logger. This assumption is reasonable because unauthorized physical access to the machine is restricted. The same assumption applies to a portable computer that the user physically safeguards at all times. To address this problem, we have created Trust sniffer, a tool that helps users incrementally gain confidence in an initially untrusted machine. Trust-Sniffer’s staged approach to establishing confidence in an untrusted machine balances the needs for security, usability, and speed. Trust-Sniffer provides greater overall security for users and increases their awareness about security risks. In future versions, we plan to explore ways to provide increased security while keeping the design focussed on nonexpert users.  Prasad Kulkarni et al., proposed a Dynamic Compilation: The Benefits of Early Investing and described the shortcomings of current approaches and demonstrate their potential to perform poorly under certain conditions. In this paper, they describes the importance of enforcing a minimum level of utilization for the compilation thread, and evaluate the performance implications of varying the utilization that is enforced. They observed large speedups by increasing the priority of the compilation thread, averaging 18.2% improvement over a large benchmark suite.They also discussed the options for implementing these techniques in a VM and address relevant issues when moving from a single-processor to a multiprocessor machine. It has demonstrated that significant performance gains can be achieved on a production virtual machine without adding any optimizations to the compiler. This illustrates that in a dynamic compilation setting, making good decisions about when and what to compile can be as important as how the code is compiled.  Naveen Neelakantam et al., described the Hardware Atomicity for Reliable Software Speculation to eliminate the complexity as well as increase the effectiveness of optimizations. These primitives provide to the compiler the ability to optimize the program’s hot path in isolation, allowing the use of non-speculative formulations of optimization passes to perform speculative optimizations. In this paper, they demonstrate the benefit of hardware atomicity in the context of a Java virtual machine. They find incorporating the notion of atomic regions into an existing compiler intermediate representation to be natural, requiring roughly 3,000 lines of code, most of which were for region formation. These optimizations reduce dynamic instruction count by 11% on average speedup, relative to a baseline compiler with a similar degree of inlining. As a result, speculative optimizations can be performed without compensation code, enabling a great reduction in compiler complexity to achieve a given code quality.  Mathew Arnold et al., proposed a framework for Reducing the cost of Instrumented code. Instrumenting code to collect profiling information can cause substantial execution overhead. This overhead makes instrumentation difficult to peform at runtime, often preventing being used in online systems. This paper presents a general framework for performing instrumentation sampling to reduce the overhead of previously expensive intrumentation. The framework is simple and effective, using code-duplication and counter-based sampling to allow switching between instrumented and non-instrumented code. The framework doesnot rely on any hardware or operating system support, yet provides a high frequency sample rate that is tunable, allowing the tradeoff between overhead and accuracy to be adjusted easily at runtime. Experimental results are presented to validate that our technique can collect accurate profiles. A Jalapeno-specific implementation is also presented as an example of how hardware- or compiler-specific optimisations can be used to further reduce overhead, resulting in an average total overhead of 3%.  In this chapter several related works are studied in detail and relevant methods, technologies, implementations are analysed. |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
| * 1. **SOFTWARE REQUIREMENTS SPECIFICATION**   **2.1 SOFTWARE REQUIREMENTS**  Operating System : Windows XP  Technology Used : Android 2.2  IDE : Eclipse 3.4(min)  Emulators : Micro emulator 5055  Plug-in : ADT plug-in  Tools Used : Android SDK1.2,  GoogleAPIv8 or minv7  **2.2 HARDWARE REQUIREMENTS**  Processor : Pentium P4  Motherboard : Genuine Intel  RAM : Min 1 GB  Hard Disk : 80 GB | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |
|  | |

**2.3 EXTERNAL INTERFACE REQUIREMENTS**

**2.3.1 USER INTERFACE**

The system includes smart’s phone. Hence the UI depends on their own device. Here we will us only Android phones. The device is expected to use internet connection through Wi-Fi/GPRS.

UI-1: They can sign up with the developed application.

UI-2: Can login into the application.

UI-3: Is allowed to provide the IP address of the cloud server.

UI-4: Now has to register as a software service for the java compiler.

UI-5: Is now allowed to log into the cloud software service.

UI-6: Here is provided with a text editing field where the program is to be typed and saved.

UI-7: The next UI lets the user to provide the java file name to compile.

UI-8: Here the error or successful report of the program is provided.

UI-9: Now has to run the program by providing the class file name which is created after compilation.

UI-10: The output program is displayed.

**2.3.2 HARDWARE INTERFACES**

This will be an Android phone application and as such will be designed to interface with the hardware present on the Android phone. In theory the application will be able to run by other devices that can emulate the Android, but this will not be a consideration during design. There are 4 physical buttons on the phone. The options button will be used specifically in multiple instances to bring up menus, such as in bringing up the ability to add an idea during the idea generation phase.

As this is a mobile device, it will be using the Android network to connect to the interest, which will allow it to communicate with the database servers. This means that it will be using infrastructure, be it wireless communication points or physical lines, of the network in order to perform properly. There will have to be some sort of error checking for if the network is down or inaccessible. This system does not require any specific hardware interface.

**2.3.3 SOFTWARE INTERFACES**

The operating system the software runs on will be the operating system the Android phone runs on, which comes with a software framework that will be utilized, including many prepackaged components to do things like create menus, hookup buttons, and other common functions excepted of a mobile device.

In order to execute and test this and android application Emulator tool is being used. This tool enables the developer to test the android application without installing it on the android smart phone. It virtually provides a android operating system on the windows frame and allows to run the application for testing and implement it virtually.

**2.3.4 COMMUNICATION INTERFACES**

HTTP is the used communication protocol. Load balancing algorithm is used in order to achieve better performance parameters such as response time and Data processing time. A proper communication interface to access internet is required.

**2.4 SYSTEM FEATURES**

This program will allow being able to use the Java compiler from their mobile phone. Any phone that supports that Android operating system will be able to install the applications and run the tools from their phone. The features of the system can be illustrated by the modules and functions involved in the system.

Some of the important features are listed below

* Software as a Service
* Mobility
* Working Environment In Application
* Debugging Environment

**2.4.1 SOFTWARE AS A SERVICE**

Software as a Service is a feature cloud computing and this feature is being implemented in this project. It is nothing but providing a software to the without installing it on the client device or machine.

**2.4.1.1 Description and Priority**

The software as a Service (SAAS) is that the software are uploaded in the cloud server, whenever the client request the software to the cloud server, the cloud server will provide the software.

**2.4.1.2 Stimulus/Response Sequences**

The is provided with the java compiler as a service. The client side android application connects with the server. This service can provided software as the product to the server.

**2.4.1.3 Functional Requirements**

Internet connection is expected to have in order to have client server architecture between the android client and server. The primary requirement is speed of the network.

**2.4.2 MOBILITY**

Mobile applications are excepted to have this feature to use and access any service from anywhere. Therefore these applications also have the feature of mobility with the help of internet connection available in it.

**2.4.2.1 Description and Priority**

The user can use this application from anywhere he wants. The user can connect to internet and start his services from anywhere he/she wants.

**2.4.2.2 Stimulus/Response Sequences**

Connecting the client android application with server to start the service and achieve SAAS.

**2.4.2.3 Functional Requirements**

Here the feature mobility is achieved by the help of internet connection.

**2.4.3 WORKING ENVIRONMENT IN APPLICATION**

An Android mobile client is an application that access a service made available by the server. The server is on another computer, in which the client accesses the service by way of a network.

**2.4.3.1 Description and Priority**

This application is developed to be used by all androids. The working environment is the android operating system on the mobile device.

**2.4.3.2 Stimulus/Response Sequences**

Interface for the android is specially concentrated in order to connect to server and work on the software available on it.

**2.4.3.3 Functional Requirements**

This function requires an android operating system mobile with internet connection enable in it.

**2.4.4 DEBUGGING ENVIRONMENT**

Debugging allows you told to run a program interactively and to watch the source code and the variables during this execution.

**2.4.4.1Description and priority**

The debugging environment refers to the compiler that is being providing as a service. This service mainly works based on watch point and break point.

**2.4.4.2 Stimulus/Response Sequences**

This provides a compiler to start the execution and stop the execution of the program with the help of watch point and break point. The user can easy to execute the program in the debugging environment.

**2.4.4.3 Functional Requirements**

This function is the main core to the project here the compiler itself is the main object that is being used.

**2.5 NON FUNCTIONAL REQUIREMENTS**

**2.5.1 PERFORMANCE REQUIREMENTS**

The primary performance requirement is speed of the network. While there should not be that much information flowing across during a SAAS utility on android phone session, if the session is short, the cannot continue with the next process like compiling and running the program. The application itself will only have minimal logic and so there should be little to no issues with the computation required by the phone itself. In order to achieve better performance parameters such as response time and Data processing time load balancing algorithm is used.

**2.5.2 SAFETY REQUIREMENTS**

There are no safety requirements with this application, other than any normal hazards of a mobile device. The only hazards is a using the device when they should not be, such as while driving.

**2.5.3 SECURITY REQUIREMENTS**

* There will be proper security regarding to the accessing the Java compiler.
* The external security can be provided by given the login authentication.
* The data that are stored in the database must be private.
* There is also required authentication.
* There is also the facility that the admin can lock his private data that will not be accessed by anyone.
* The whole software is secure from the outside accessing.
* The user cannot compile and execute the program without the proper authentication.

**2.5.4 SOFTWARE QUALITY ATTRIBUTES**

Our software has many quality attribute that are given below:

**Usability:** Th**e** primary attribute of this application will be usability given the large amounts of data and information that will be presented on such a small screen, as well as the ability to input data into the device in a reasonable manner that should not be that much more difficult than if they were at an actual computer.

**Cloud Computing:** We are implementing Software as a Service (SAAS) for Cloud Computing. SAAS is the Cloud Computing Resource, used for the service of Software without installing that software in the device.

**Mobility:** It allows compiling and executing Java programs directly through android mobile the Android mobile from anywhere by using this application.

**Reduced Time:** We will execute small java program through android mobile it reduce the time consistency.

**Open Source:** The Android device is being used because both of its popularity and the ability for the code to be open-source.

**Productivity:** It allows the programmer to develop his program using his android device.

**2.6 ALGORITHM DESCRIPTION**

**2.6.1 LOAD BALANCING ALGORITHM**

In this algorithm firstly analysis of different Virtual Machine (VM) load balancing algorithms is done. Secondly, a new VM load balancing algorithm has been proposed and implemented for an SAAS framework in Simulated cloud computing environment, i.e. ‘Weighted Active Monitoring Load Balancing Algorithm’ using tools, for the Data center to effectively load balance requests between the available virtual machines assigning a weight, in order to achieve better performance parameters such as response time and Data processing time.

* 1. **SYSTEM ANALYSIS AND DESIGN**

Systems development can generally be though of as having two major components: System analysis and Systems design. System design is the process of planning a new business system or one to replace or complement an existing system. But before this planning can be done, we must thoroughly understand the old system and determine how computers can best be used to make its operation more effective. System analysis, then, is the process of gathering and interpreting facts, diagnosing problems, and using the information to recommend improvements to the system.

**3.1 SYSTEM DESIGN ARCHITECTURE**

A system architecture or systems architecture is the conceptual model that defines the structure, behavior and more views of the system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structure of the system which comprises system components, the externally visible properties of those components, the relationships (e.g. the behavior) between them and provides a plan from which Products can be procured, and system developed, that will work together to implement the overall system.

**FIG 3.1 System Architecture**

Figure 3.1 indicates the architecture diagram of online java compiler using cloud computing in android mobiles. First of all, the user requests the program to compile in a server. The server compiles the program in the server using cloud. The server will be connected to SAAS. SAAS is the Cloud Computing Resource, used for the service of without installing that Software in the Device. Here, we are compiling the code using Android Smart phones without installing Software in Mobile Phone. Finally after the compilation, the result/execution will be generated to the Android user.

**3.2 MODULE DESCRIPTION**

**3.2.1 MOBILE CLIENT**

An Android mobile client is an application that access a service made available by a server. The server is often (but not always) on another computer, in which case the client accesses the service by way of a network. The term was first applied to devices that were not capable of running their own stand-alone programs, but could interact with remote computers via a network.

User Information

Login

Client

Application

Connection established

Java program

**FIG 3.2 Mobile Client**

**3.2.2 CLOUD SERVER**

**Fig 3.4 Sequence Diagram**

Cloud Computing, as the name suggests is a style of computing where dynamically scalable and often visualized resources are provided as a service over the internet. These services can be consumed by any over a standard HTTP medium. The user does not need to have the knowledge, expertise or control over the technology infrastructure in the "cloud" that supports them. Cloud Servers offer increased flexibility and higher quality than dedicated server solutions. Cloud servers are highly available. A cloud server is considered to be logical when it is delivered through server virtualization. Servers are highly available. A cloud server is considered to be logical when it is delivered through server virtualization.

Cloud

Compiler

Login

Cloud IP address Cloud Interface

**FIG 3.3 Cloud Server**

**3.2.3 SAAS**

The SAAS implementation is achieved using Java, C, C**++** software. The traditional model of software is purchased for and installed on personal computers, is sometimes referred as software as a product .We all understand that without these four software we cannot compile our program. The Software as a Service (SAAS) is that the software are uploaded in the cloud server, whenever the client request the software to the cloud server, the cloud server will provide the software, which is chargeable in rental manner. This process will be of use to reduce the client system load. The Service Level Agreement is an important agreement document because it clearly defines what a (SaaS) service provider is offering and also what consequences they will face if they fail to deliver these services, to the agreed standard.

Login

Cloud

Compiler

Java Program Program output

**FIG 3.4 Software as a Service**

**3.3 DATA DESIGN**

**3.3.1 DATA DESCRIPTION**

Data structure is representation of login relationship among individual element of data. Because the structure of information will invariably affect the final procedural design, data structure is very important as the program structure to the representation of the software architecture. Data structure dictates the organization, method of access, degree of associability and processing alternative for information. The organization complexity of data structure is limited only by the ingenuity of the designer. Scalar item array and linked list are some of the representation of the data structure. Data structure dictates the organization, method of access, degree of associability and processing alternative for information.

**3.3.2 DATA DICTIONARY**

A database is a collection of interrelated data store with minimum redundancy to serve many s quickly and efficiently. The general objective of the database is to make the data access easy, inexpensive and flexible .The tables need to be created in order to store the details of the . The login table store the information needed to be checked when a tries to login.

**3.4 COMPONENT DESIGN**

* + 1. **AUTHENTICATION**

Authentication is server that provides authentication services to users or other systems via networking. Authentication is used as the basis or authorization (determining whether a privilege will be granted to be particular user or process).Privacy is keeping information from becoming known to non-participants.

* Client side validation on android device
* Server IP address validated at client
* Server URL is validated to establish connection
* If invalid input of user exception message display
* Server side user validation to start service
  + 1. **IMPLEMENTATION OF SOFTWARE AS A SERVICE**

We are implementing Software as a Service (SaaS) for Cloud Computing. SAAS is the Cloud Computing Resource, used for the service of Software without installing that Software in the Device.

* Client provide the IP address of cloud service
* Establish connection
* Map the corresponding servlet application
* Client stores his java program on server
* Enable java compiler service to client
  + 1. **WORKING ENVIRONMENT IN APPLICATION**

When the user can easily access and compile the program through just using mobile interface. Cloud computing gives you access to an environment that you can customize or build out to suit your needs.

* Create and connect to the sqlite data base
* Connect to server by valid IP address of server
* Access a service made available by a server
* Client accesses the service by way of a network.
  + 1. **DEBUGGING ENVIRONMENT**

The main tab defines the class to be launched enter the name of the project containing the class launch to the project field and fully qualified name of the main class in the main class field. Check the stop main checkers if you want the program to stop in the main method whenever the program launched in debug mode.

* Watch the source code and the variables during this execution
* Use breakpoints to stop execution of the program
* Use watch point to start the execution of the program
* Compile the user program
* Generate class file
* Run program by executing class file.

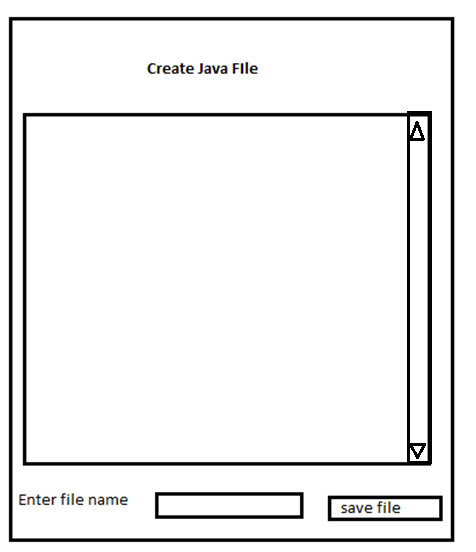
**3.5 HUMAN INTERFACE DESIGN**

**3.5.1 OVERVIEW OF INTERFACE**

The main interface for the user is extremely friendly. The user just needs to authenticate his profile to store information in the database. The can create his program, compile his program, debug easily with the help of error reports and run the program.

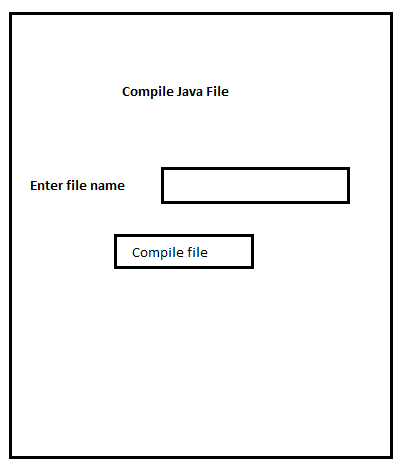
**3.5.2 SCREEN OBJECTS AND ITS ACTION**

Main form provides the fields of creating a java file using multiline text control. After clicking the save file button, corresponding java file saved in the serve and made ready to compile and run class file.



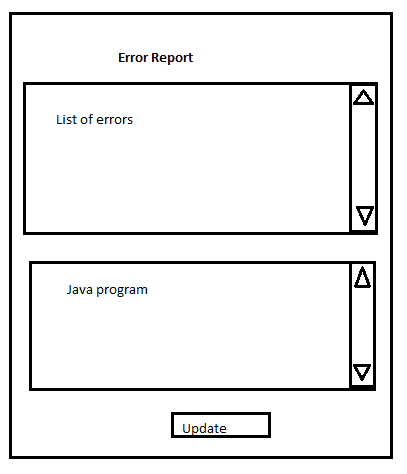
**FIG 3.5 Create Java Program**

The above Fig 3.5 allows the user to type the java program and save the java program on the server side. The created java file is saved at the server and then made ready to compile. If extension then the created file is not in the form of java extension message is displayed and the user has to create a new java program on the same user interface.



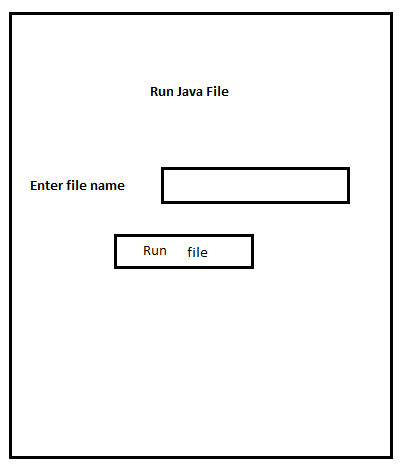
**FIG 3.6 Compile Java Program**

The Fig 3.6 shows the process of saving the java program file. Here the java file is saved by the name of class name. Now the file is made ready for compiling. When the compilation of the file is done the class file is generated. The class file is created at the same directory where the source code or java program is saved.



**FIG 3.7 Debugging Interface**

This Fig 3.7 illustrated the way by which the user created java program is debugged. While compiling if any error is found the error report is being generated on first half of the user interface layer. The user is enabled to update the error in the program and save the program with the changes made on the program.



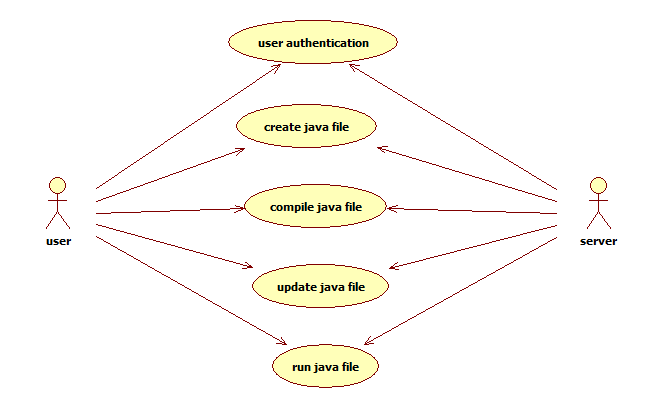
**FIG 3.8 Running Program**

The above Fig 3.8 shows the process of execution of the java program saved on the server and the corresponding output generated from it. The execution of the java program is done based on the class file which is given as input by the user. Class file is stored in the directory where the java program files are present.

* 1. **DFD DIAGRAM**

**3.6 USE-CASE DIAGRAM**

Use case diagram the relationship and dependencies between a group of use cases and actors participating in the process. Use case diagram are meant to facilitate the communication with future of the system and with the customer. Use case diagram explains about the process to be carried between various use cases.



**FIG 3.9 Use-Case Diagram**

**Fig 3.2 Use –Case Diagram**

**3.7 CLASS DIAGRAM**

The class diagram models class structure and contains using design elements such as classes, packages and object .It illustrates the relationship and source code dependencies among the classes. It also displays relationship such as containment, inheritance, associations and others. Lines, which may have two arrows at one end or both ends, connect the boxes. These lines define the relationships, also called associations, between the classes.

**Client**



**cid: int**



**pwd: string**



**createfile ()**



**compilefile ()**



**runfile ()**

**server**



**username: string**



**password: string**



**filename: string**



**validate ()**



**compile ()**



**sendresult ()**

**1..\***

**1..\***

**User**



**username: string**



**password: string**



**filename: string**



**classname: string**



**validation ()**



**createfile()**



**compile()**



**run ()**

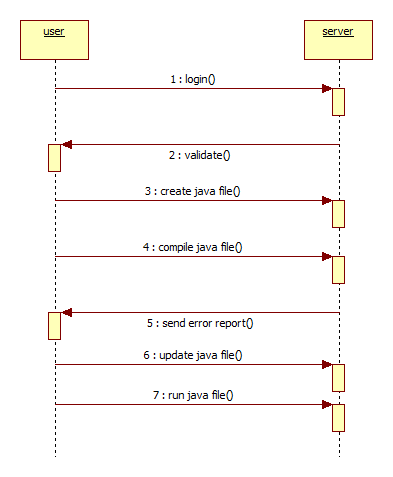
**1**

**1**

**FIG 3.10 Class Diagram**

**3.8 SEQUENCE DIAGRAM**

Sequence diagram model the flow of logic within the system in a visual manner, enabling both to document and validate the logic and are commons. It shows the object interaction arranged in time sequence. When these diagram are typically associated with the use case realizations in the logical view of the system under development.

****

**3.9 ACTIVITY DIAGRAM**

An activity diagram is represented by shapes that are connected by arrows. Arrows run from activity start to completion and represent order of performed activities. Black circles represent an initial workflow state.

**FIG 3.12 Activity Diagram**

login

register

enter the source

code

save the class file

compile the class

file

runfile

output generated

create the

class file

yes

no

**4. IMPLEMENTATON SAND TESTING**

Implementation is the most crucial stage in achieving a successful system and giving the confidence that the new system is workable and affective. This type of conversation is relatively easy to handle, provide there are no major changes in the system.

Each program is tested individually at time of development using the data

And has verified that this program linked together in the way specified in the programs specification, the computer system and its environment is tested to the Satisfaction. The system that has been developed is accepted and proved to be satisfactory for them. The system is going to be implemented very soon. A simple operating procedure is included so that they can understand the different functions clearly and quickly. Initially as a first step the executable from of the application is to be created and loaded in the common server machine which is accessible to the entire and the server is too connected to a network. The final stage is to document the entire system which provides components and the operating procedures of the system.

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be most critical stage in achieving a successful new system and in giving the, confidence that the new system will work and be effective.

The implementation stage involves careful planning, investigation of the system and its constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

Implementation is the process of converting a new system design into operation. It is the phase that focuses on training, site preparation and file conversion for installing a candidate system. The important factor that should be considered here is that the conversion should not disrupt the functioning of the organization.

**4.1 CODING**

**AUTHENTICATION IN CLIENT DEVICE**

public class Welcome extends Activity

{

Button submit, reset, exit signup, signup\_back\_button, signup\_submit;

Edittext\_txt, text\_pass, signup\_name, signup\_password, signup\_reenter\_password, email\_ edit, phone\_ edit;

ViewFlipperviewFlipper;

SQLiteDatabasedb;

Boolean = false, pass = false, check = false;@Override

public void onCreate (Bundle savedInstanceState) {

super.onCreate (savedInstanceState);

if(((!txt\_.getText().toString().isEmpty()))&&((!txt\_pass.getText().toString().isEmpty()))){

SetContentView (R.layout.main);

create Database ();

createWidgetId ();

submit.setOnClickListener (new OnClickListener () {

public void onClick (View v) {

super.onCreate (savedInstanceState);

SetContentView (R.layout.main);

create Database ();

createWidgetId ();

submit.setOnClickListener (new OnClickListener () {

public void onClick(View v) {

if(((!txt\_.getText().toString().isEmpty()))&&((!txt\_pass.getText().toString().isEmpty()))){

Cursor cursor = db.query ("Registration", new String [] {"name"}, null, null, null, null, null);

while(cursor.moveToNext ()) {

if(cursor.getString(0).equals(txt\_getText().toString())) {= true;

Log.d ("Pass", cursor.getString (0));

Cursor cursor1 = db.query ("Registration", new String [] {"Password"}, "name like \'" + cursor.getString (0) + "\'", null, null, null, null); while (cursor1.moveToNext ())

{

if (cursor1.getString(0).equals(txt\_pass.getText().toString()))pass = true;

}

}

}

if ( && pass) {

Intent intent = new Intent (Welcome.this, MainMenu.class);

startActivity (intent);

}

else {

Toast.makeText (getApplicationContext (), "name or Password is incorrect", Toast.LENGTH\_LONG).show ();

}

}

else {

Toast.makeText (getApplicationContext (), "name or Password fields can't be empty", Toast.LENGTH\_LONG).show();

}}

});

reset.setOnClickListener (new OnClickListener () {

public void onClick(View v){

txt\_setText ("");

txt\_pass.setText ("");}

});

**CLIENT CONNECTION SERVER**

Public class ConnectToServer extends Activity

{

privateHttpResponse response;

privateHttpResponse response;

private String responseText, message=””;

private void onCreate (BundlesavedInstanceState) {

super.onCreate (savedInstanceState);

try

{

Bundle bundle = this.getIntent ().get Extras ();

message = bundle.getString ("sts");

Log.v ("ConnectToServer", message);

httpclient = new DefaultHttpClient ();

HttpGethttpget=newHttpGet("http://"+ServerIPAddress.getIpaddress ()+":8000/"+message);

response = httpclient.execute(httpget);

HttpEntity entity = response.getEntity();

responseText = EntityUtils.toString (entity);

Log.v ("ConnecttoServer", responseText);

if ((responseText.trim ()).startsWith("CLI")){

Intent i = new Intent (ConnectToServer.this, ClientList.class);

Bundle = new Bundle ();

bundle.putString ("CLI", responseText.trim ());

i.putExtras (bundle);

startActivity (i);

}

} catch (Exception e)

{

e.printStackTrace ();

Log.v ("Connect To Server”, “Exception..."+e);

}

}

**CREATING A JAVA FILE ON SERVER**

public class Save File extends GenericServlet

{

FileOutputStreamfos;

Public void service (ServletRequest req, ServletResponse res) throws ServletException, IOException

{

res.setContentType ("text/html");

String s=req.getParameter ("textarea");

String ff=req.getParameter ("filename");

String dir=req.getRemoteHost ();

File kk=new File ("c: /"+dir);

booleanbb=kk.mkdir ();

Print Writer pw=res.getWriter ();

fos=new FileOutputStream (kk+"/"+ff);

int end=s.length ();

int start=0;

while (start<end)

{

fos.write (s.charAt (start));

start++;

}

fos.close ();

//RequestDispatcherrd = req

pw.println ("<html>");

pw.println ("<head>");

pw.println ("<title>successfully stored</title>");

pw.println ("</head>");

pw.println ("<body>");

pw.println ("<table width=250 border=1 align=center bgcolor=#993366>");

pw.println ("<tr><td><h3><center>FileActionReport</center></h3></td></tr>");

pw.println("<tr><td><h4><br/><br/><center>"+ff+"File is successfully saved");

pw.println ("</h4></center>");

pw.println ("<a href='CompileJavaFile.html'>");

pw.println ("<center> Click here to Compile</center></a></td>");

pw.println ("</tr>");

pw.println ("</table>");

pw.println ("</body>");

pw.println ("</html>");

pw.close () ;

}

}

**COMPILING THE JAVA FILE AT THE SERVER**

Public class CompileFile extends GenericServlet

{

ServletException,IOException

{

res.setContentType("text/html");

PrintWriter pw=res.getWriter();

String s=req.getParameter("file");

System.out.println("File name:- "+s);

// s="Master.java";

String dir=req.getRemoteHost();

System.out.println("File dir:- "+dir);

InputStream is;

int c;

int i=0;

StringExtract se=new StringExtract(s);

if(se.extract())

{

FileInputStreamfs=new FileInputStream("c:/"+dir+"/"+s);

Runtime r=Runtime.getRuntime();

System.out.println("c:/"+dir+"/"+s);

Process p=r.exec("javac "+"c:/"+dir+"/"+s);

is=p.getErrorStream();

c=is.read();

if( c!=-1)

{

pw.println("<html>");

pw.println ("<head>");

pw.println ("<title>Successfully Compiled</title>");

pw.println ("</head>");

pw.println ("<body>");

pw.println ("<tr>");

pw.println ("<td width=272 height=57 align=center valign=middle >ERROR REPORT");

pw.println ("</td>");

pw.println("</tr>");

//pw.println ("<marquee><h2><b>ERRORREPORT</marquee>");

pw.println ("<tr>");

pw.println("<td>");

pw.println ("<form method='post' action='SaveAgain'>");

pw.println ("<pre>");

pw.close ();

}

}

**4.2. TEST PLAN**

Testing is a set of activities that can be planned in advance and conducted systematically. For this reason a template for software testing, a set of steps into which we can place specific test case design techniques and testing methods should be defined for software process. Testing often accounts for more effort than any other software engineering activity. If it is conducted haphazardly, time is wasted, unnecessary effort is expanded, and even worse, errors sneak through undetected. Some of the test plans are

1. Conventional Testing
2. Unconventional Testing

**4.2.1 UNIT TESTING**

The primary goal of unit testing is to take the smallest piece of testable software in the application, isolate it from the remainder of the code, and determine whether it behaves exactly as you expect. Each unit is tested separately before integrating them into modules to test the interfaces between modules. Unit testing has proven its value in that a large percentage of effects are identified during its use. In the company as well as seeker registration from, the zero length name and password are given and duplicate name is given and checked. Also the duplicate name is given and checked. In the job and question entry, the button will send data to the server only if the client side validations are made. The dates are entered in wrong manner and checked. Wrong email-id and web site URL (Universal Resource Locator) is given and checked.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SI. No.** | **Function/ Method** | **Sample Input** | **Expected Output** | **Original Output** | **Status** | **Error** |
| 1 | Validation | Get name and password | Login to client application | Login to client application | Success | NIL |
| 2 | Start the Service | Provide cloud server IP address | Connect to server | Connect to server | Success | NIL |
| 3 | Server side validation | Get name and password | Connect to server | Choose another file | Success | NIL |
| 4 | Service menu | Select the options | Enable the service to client | Enable few service to client | Failure | Interrupt |

**Table 5.1: Unit Testing**

**4.2.2 INTEGRATION TESTING**

Testing is done for each module. After testing all the modules, the modules are integrated and testing of the final system is done with the test data, specially designed to show that the system will operate successfully in all its aspects conditions. Thus the system testing is a confirmation that all is correct and an opportunity to show that the system works.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SI. NO.** | **Function/ Method** | **Sample Input** | **Excepted output** | **Original Output** | **Status** | **Error** |
| 1 | Creating java file at server | Java code | Creation of java file | Creation of text file | Failure | Interrupt |
| 2 | Compiling java file at server | Java file name to compile | Generating class file | Generating class file | Success | NIL |
| 3 | Running file  at server | Class file name | Output of program | Output of program | Success | NIL |

**Table 5.2: Integration Testing**

**4.2.3 VALIDATION TESTING**

The final step involves Validation testing, which determines whether the software function as the excepted. The end-rather than the system developer conduct this test most software developer as a process called “Alpha and Beta Testing” uncover that only the end seems able to find. The compilation of the entire project is based on the full satisfaction of the ends. In the project, validation testing is made in various forms. In question entry form, the correct answer only will be accepted in the answer box. The answers other than the four given choices will not be accepted.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SI. No.** | **Function/ Method** | **Sample Input** | **Expected Output** | **Original Output** | **Status** | **Error** |
| 1 | Login | Inappropriate password | Login failure | Invalid password | Success | NIL |
| 2 | Registration | Null input | Invalid | Enter valid | Failure | Interrupt |
| 3 | Verify id | id | id valid | Message | Success | NIL |
| 4 | File name | Name of java file | Invalid | Invalid | Success | NIL |

**Table 5.3: Validation Testing**

**4.3 TESTING STRATEGIES**

Anumber of software testing strategies have been proposed in the literature. All provide the software developer with a template for testing and all have the following generic characteristics:

* Testing begins at the component level and works “outward” toward the integration of the entire computer-based system.
* Different testing techniques are appropriate at different points in time.
* The developer of the s/w conducts testing and for large projects, independent test group.
* Testing and debugging are different activities, but debugging must be accommodated in any testing strategy

**4.3.1 INTEGRATION TESTING**

The strategies for integrating software components into a functioning product include the bottom-up strategy, the top-down strategy and to ensure that modules will be available for integration into the evolving software product when needed. The integration strategy dictates the order in which modules must be available and thus exerts a strong influences on the order in which modules are written, debugged and unit tested.

**4.3.2 WHITE BOX TESTING**

It is just the vice versa of the Black Box testing. There we do not watch the internal variables during testing. This gives clear idea what is going on during execution of the system. The point at which the bug occurs were all clear and were removed.

**4.6.3 INTERFACE TESTING**

The interface Testing is performed to verify the interfaces between sub modules while performing integration of sub modules aiding master module recursively.

**4.3.4 MODULE TESTING**

Module Testing is a process of testing the system, module by module. It includes the various inputs given, outputs produced and their correctness. By testing in this method we would be very clear of all the bugs that have occurred.

**4.3.5 SMOKE TESTING**

Smoke testing refers to physically tests made to closed systems of pipes to test for leaks. By metaphorical extension, the term is also used for the first test made after assembly or repairs to a system, to provide some assurance that the system under test will not catastrophically fail. The term smoke testing is used in several fields, including electronics, computer software development, plumbing, woodwind repair, infectious diseases control and the entertainment industry.

**4.4 MAINTENANCE**

The objectives of this maintenance work are to make sure that the system gets into work all time without any bug. Provision must be for environmental changes which may affect the computer or software system. This is called the maintenance of the system. Nowadays there is the rapid change in the software world. Due to this rapid change, the system should be capable of adapting these changes. In our project the process can be added without affecting other parts of the system. Maintenance plays a vital role. The system will be able to accept any modification after its implementation. This system has been designed to flavor all new changes. Doing this will not affect the system’s performance or its accuracy. This is the final step in system life cycle. Here we implement the tested error-free system into real-life environment and make necessary changes, which runs in an outline fashion.

As a rule, system testing takes, as its input, all of the “integrated” software components that have successfully passed integration testing and also the software system itself integrated with any applicable hardware system(s). The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called assemblages) or between any of the assemblages and the hardware.

**5. CONCLUSION AND FUTURE WORK**

**5.1 CONCLUSION**

This project helps to compile and execute Java programs directly the Android mobile so that they can concentrate on the programming concepts rather than learning to operate operating system. This feature allows students to do Java programming anywhere, anytime using just mobile interface.

This project developed a prototype for android architecture for a multiple system. This proposed system gives more efficient gain when compared with the existing system. By providing an open development platform, Android offers developers the ability to build extremely rich and innovative applications.

Developers are free to take advantage of the device hardware, access location information, run background services, set alarms, add notifications to the status bar, and much, much more.

**5.2 FUTURE WORK**

The application architecture is designed to simplify the reuse of components, any application can publish its capabilities and any other application may then make use of those capabilities (subject to security constraints enforced by the underlying all applications is a set of services and systems, including: A rich and extensible set of views that can be used to build an application, including lists, grids, text boxes, buttons and even an embeddable web browser.

**LOGIN LAYOUT**

****

**A1.Register Page**

The screenshot shows the register form. It is the first stage of the project. This page provides registration for new. Already registered can sign in from the home page. All the new records are stored on the mysql database. The data adapter is initialized here.

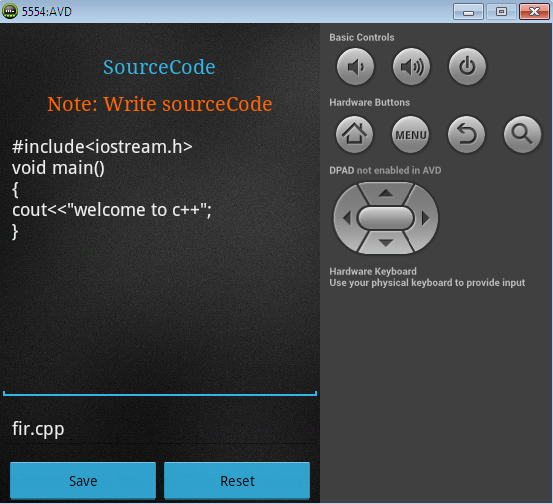
**SERVICE OPTIONS**

****

**A2. Service Options**

The above screenshot shows how the user to choose the option he wants to perform and use the cloud service that is provided as online compiler using Android smart phone. Without installing the software the program is compiling in our Android smart phone.

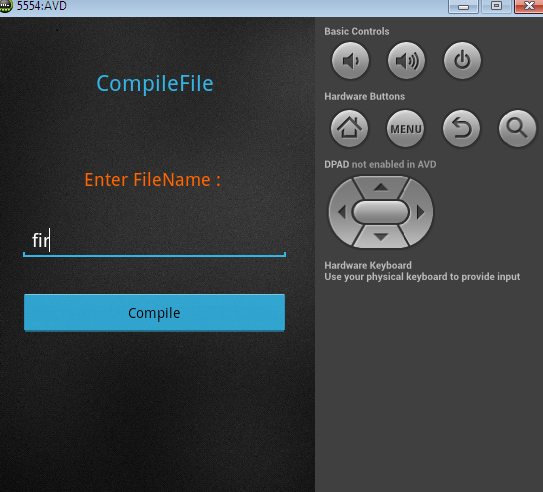
**CREATING A FILE**

****

**A3.Creating a file**

The above screenshot allow the user to type the program and save the program on the server side. The created file is saved at the server and the made ready to compile. If the created file is not form of programming extension the the exception message is displayed and the user has to create a new program on the same user interface.

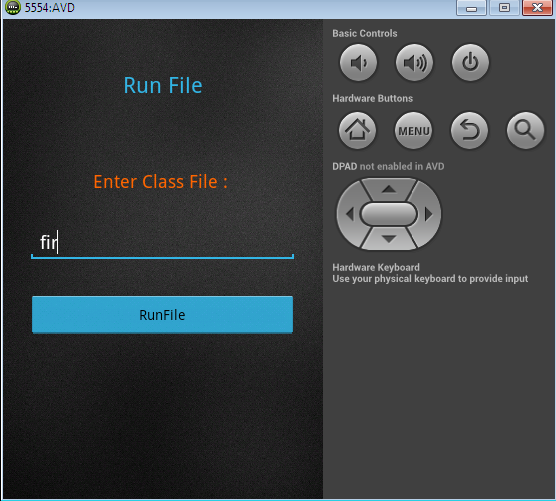
**COMPILING THE PROGRAM FILE**

****

**A4.Compiling Program File**

Above screen shot shows the process of saving the program file. Here the program is saved by the name of class name. Now the file is made ready for compiling. When the compilation of the file is done the class file is created at same directory where the source code or programming is saved.

**EXECUTION AND OUTPUT**

****

**A5.Execution and Output**

This screenshot shows the process of execution of the program saved on the server and the corresponding output generated from it. The execution of the program is done based on the class file which is given as input by the user. Class file is generated immediately when the compile process is done. The class file is stored in the directory where the compile is done. The class file is stored in the directory where the program file is present.

**DEBUGGING ENVIRONMENT**

**REFERENCES**

[1] Markets and Markets, “World Mobile Applications Market -

Advanced Technologies, Global Forecast(2010 - 2015),” 2010.

[Online]. Available: <http://www.marketsandmarkets.com/>

[2] D. Gavalas and D. Economou, “Development platforms for

mobile applications: Status and trends,” IEEE Software, vol. 28,

pp. 77–86, 2011.

[3] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti,

“CloneCloud: elastic execution between mobile device and

cloud,” in Proc. ACM EuroSys, 2011.

[4] R. Ma and C.-L. Wang, “Lightweight application-level task

migration for mobile cloud computing,” in Proc. IEEE AINA,

2012.

[5] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, “The

case for VM-based Cloudlets in mobile computing,” IEEE

Pervasive Computing, vol. 8, no. 4, pp. 14 –23, 2009.

[6] M.-R. Ra, A. Sheth, L. Mummert, P. Pillai, D. Wetherall,

and R. Govindan, “Odessa: enabling interactive perception

applications on mobile devices,” in Proc. ACM MobiSys, 2011.