Nebula - Cloud Based Platform for Educational Institutions
Shruti Menon¹, Bhamidipati Prathyush², Sri Vidya B. Iyer³
Department of Computer Science and Engineering
National Institute of Engineering, Mysuru, India

Abstract:
Recent trends in the IT industry have shown that Cloud Computing is a fast developing sector. Most of the organizations in the industry are looking to make a transition towards a Cloud based platform for acquiring and providing their services. One of the main areas where Cloud computing can have a huge impact due to its features of simpler management and scalability is in the field of Education and learning. Any platform for an educational institution requires a good amount of hardware and software resources. This platform provides a way for centralized administration and automated provisioning of various cloud based services including Storage (Storage as a Service-StaaS), various software (Software as a Service-SaaS) and multiple development platforms (Platform as a Service-PaaS). The students can provision the resources and use them for academic purposes, whereas the professors can use the same in order to monitor the performance of various students.

Keywords: Cloud Computing, Containerization, Automation, Educational platform.

I. INTRODUCTION
Cloud computing is one of the most trending topics in today's world of technology. It is basically a model for enabling on-demand access to a pool of resources that can be accessed over a network platform. The options available in the form of multiple deployment models make this an efficient way of automated provisioning of resources. The advancement of cloud computing in the field of Education can result in more effective ways for a student to manage his academic life and at the same time provide the professors with a method to gauge the performance of the students over remote access of the submissions. Any platform that is developed for an Educational institution must be able to provide multiple students simultaneous access to the remote resources made available by the administrator. The main focus of such a platform must be the ability to blend in the various cloud service models on a single platform. This means that the students should be able to request and be allocated with storage, run high-end software required for academics without having to install the entire software at the client system and also get the required development platforms with all development frameworks available. This enables the end-users to perform the required coursework while at the same time saving time and space used up for installing the required software. To help the students and the teaching faculty to make use of all these features, this paper proposes a private cloud service for schools or colleges, providing services in the form of StaaS, SaaS and PaaS. A private cloud is envisioned for this purpose, as only the people of interest from a particular institution must be able to access these resources over the platform. StaaS, SaaS and PaaS service models are used in order to provide the required services to the users in the form of virtualized resources. Multi-tenancy i.e. multiple instance of the same applications running simultaneously, is provided by the system with the help of provisioning of the virtualized resources to each of the clients. This is made possible in the system with the help of containerization and automation, which can help in rapid provisioning and easy management of the resources. Containerization is the process of combining an application or a software with all its requisite configurations and dependencies, thus enabling it to run efficiently across multiple platforms without the hassle of configurations. This enables the students and the faculty to run the required software and development frameworks on their system, without the need of installing the heavy software on their systems. Most of the tasks that are being run in the platform are repetitive in nature, meaning that these tasks are performed for every client over and over again. For the admin to perform these tasks every single time, is a tedious thing to do. For reducing this efforts, the tasks are automated. Automation is the process of programming the system to automatically schedule some tasks to be run when certain conditions are met. This can be used in order to automate the flow of the system, thus reducing the efforts and also simplify the management of the provisioned resources. The clients are required to authenticate themselves by logging in to the platform by using their unique college credentials. Upon successful authentication, the client will then have the choice of choosing from one of the available services. On selection of the required service, further details are requested from the user, if any. An automated script runs in the background which successfully provisions the requested service for the particular client.

II. LITERATURE REVIEW
Cloud Computing is one of the most disruptive technologies that has come up over the last few years, with various models of deployment.

- In [1], authors elaborate on what exactly Cloud computing is, the different service models that can be deployed and the pros and cons of them. This research discusses the various benefits of Cloud computing. The major benefits include cost, speed, global scale and reliability. It defines various service models inclusive of Storage-as-a-Service, Platform-as-a-service and Software-as-
a) Presentation layer: This is the layer which acts as the frontline for the project, which is visible and interacts with the client.

i) HTML: It is a text markup language which is used in order to structure and present content on the World Wide Web.

ii) CSS: It is a language used for describing the presentation of a project by styling the content written by a markup language.

• In [2], the authors provide a brief overview on Cloud technology, how it is transforming industrial automation, enhancing productivity and cost optimization. The research also examines strategies for effective deployment and maintenance so that resources can be managed effectively and provisioned rapidly. There are various essential layers that work together to make cloud automation and platform orchestration successful. It provides a detailed strategy for containerization and automation in cloud computing. The implementation of Continuous Integration/ Continuous Deployment using Dockers is explored. With Docker, implementation of the private cloud becomes much easier and feasible as authentication and container management is handled.

III. PROPOSED METHODOLOGY

• System Design

The proposed system design divides the entire system functionality into 4 different layers. These include Code layer, Service layer, Business or Logical layer and Presentation layer. The components and functionality of each of them are given as follows:

• a) Presentation layer:
This is the layer which acts as the frontline for the project, which is visible and interacts with the client.

i) HTML: It is a text markup language which is used in order to structure and present content on the World Wide Web.

ii) CSS: It is a language used for describing the presentation of a project by styling the content written by a markup language.

iii) JavaScript: It is a client scripting language which is used in order to create dynamic web pages.

• b) Business or Logical layer: This layer specifies how exactly the multiple services and resources are provided to the user via the cloud platform.

i) Storage-as-a-Service: Logical volume partition is created according to the requested size and mounted on the client system using Network-File-System (NFS).

ii) Platform-as-a-Service: Platforms are configured inside Docker containers and saved as separate Docker images which are launched on client system according to the user choice or can be accessed on the browser through published ports.

iii) Software-as-a-Service: Software is installed and configured inside Docker containers and saved as Docker images, which are launched on the client system based on the client requests.

• c) Service layer: The service layer provides services to interconnect the front end and back end code. This layer provides authentication and security to the system as well.

i) NFS: Network File System is used for sharing files and folders across remote systems. It allows you to mount your local file systems over a network and remote hosts to interact with them as they are mounted locally on the same system. With the help of NFS, file sharing can be set up between UNIX to Linux system and Linux to UNIX system. NFS in the proposed system is used in case of Storage-as-a-Service (StaaS) for the client to use the space created on server.

ii) Docker: It is a containerization technology i.e. a Docker container is a one-click Operating System that packages up all code and dependencies of any software and shared amongst different systems. In a basic image of an OS, the required application software is installed along with pre-requisite system libraries and other user dependencies. This container is then saved as a Docker image. This Docker image can be used in order to deploy the container with the pre-installed application on any system, thus providing interoperability.

iii) MySQL: It is an open-source relational database management system. It stores the data in the forms of tables, where each row identifies a distinct object and each column signifies a specific attribute of the object. It is used for keeping a database of all the students and the teaching faculty who have an access to platform and their credentials.

• d) Code layer:
This is the most important layer, containing all the backend code that is responsible for running all the tasks which provide the cloud services to the clients. This layer connects to the database and also runs the required services based on the requests.

i) Python-CGI: Python-CGI stands for Common Gateway Interface. A HTTP script can call a Python-CGI script, such that any input given by the user can be processed at the back-end. The CGI scripts are stored in the cgi-bin directory on a UNIX or Linux system. When the client system performs an action on the HTML page which causes the CGI script to be called, a query
string containing all the parameters and also other required script variables are passed on to the server. The server is responsible for taking the values in the query string and executing the called-upon script and pass the results back on to the client system.

The server is responsible for taking the values in the query string and executing the called-upon script and pass the results back on to the client system.

**Ansible**: Ansible is an automation tool that can be used for orchestrating workflows in a software system. Ansible is mainly used for the deployment of applications, provisioning of resources on a cloud platform and for configuration management. In the proposed system, Ansible is used for provisioning of virtualized cloud resources, by automating the repetitive tasks that are done every time the client or the end-user requests for a particular service.

The main services provided to the end-users are:

i) **Storage-as-a-Service**: The client is given an option to use the storage systems available on the server either as a cloud storage or local storage. The storage systems are created using ansible-playbook. For storage creation, constant disk partition is pre-configured in the server and the block device is initialized to create the physical volume which is combined to form a volume group i.e. pool of disk-space from which logical volumes are allocated. From this volume group, the requested storage is created and mounted onto a folder in the Cloud folder with a unique name identifying the user. Formatting and mounting is performed following which Network File System (NFS) service is used to mount the storage on the client system with the help of the IP address of the server. The below two services are implemented using Docker containers which are saved as .tar files and transported on to the client as and when requested from which the particular Docker image is extracted. All the software and platforms are installed and pre-configured inside the Docker containers. Every machine has to be recognized with a universal unique id called as machine-id which is generated using dbus-uuidgen inside the container.

ii) **Software-as-a-Service**: The client is given an option between two software, Firefox and VLC media player. Firefox requires sharing graphical socket between server and host system with initialization of DISPLAY variable which provides the graphical interface for the client. This is done on.
the client system with the help of Ansible modules and the Docker container is launched, thus providing Firefox to the user. For VLC to be launched, in addition to graphical socket sharing and initializing DISPLAY variable, additional audio device has to be shared with the container in order to allow the container to access the physical audio device of the client system. Privileged access needs to be given to the container in order to allow VLC to be run on the containers. Also, the main home folder is mounted onto the container so that all media files to be played are available inside the Docker container without taking additional space in the container. All these environment variable configurations and sharing of audio devices is done with the help of Ansible modules. Once this is done, the VLC media player is launched.

iii) Platform-as-a-Service: The client is given an option between 2 platforms, Jupyter notebook and Codeblocks which serve as the
Package installer for python is loaded on the client system and configuration file for Jupyter notebook is generated. A Docker container is launched by exposing Jupyter notebook port number 8888 to the external network by binding it to a random port number. This process is called publishing the port which makes the Jupyter-notebook accessible on the browser by remote client system. For Codeblocks to run, it is installed and configuration file is generated. All the compilers which are necessarily required for codeblocks to run have to be pre-installed. This includes the GCC and G++ compilers which are required for the development work. This development platform is accessed on your system itself once the Docker container is deployed. The faculty have additional functionalities. Apart from being able to provision the above mentioned services, once a faculty member logs in to the platform, an option to view the student directories is given. A directory listing of all the student directories is shown. The faculty can choose which one of the student folders needs to be accessed and view the particular submissions made by the student. This enables them to keep a track of the performance of the students.

V. RESULTS
i) Every client is provided with one storage space which is checked against their unique USN or username of client. The user needs to input the size of the storage required by them. If storage is present, directory listing is shown. If not, then a storage is created on the server and is shared to the client.

vi) For Jupyter notebook, once the client requests for the platform to be deployed for their assignment work, or any other development work, it is launched and then the client can access the Jupyter notebook by opening a connection to a specified port number which is shown to the client.

V. REFERENCES