A Design of Cloud-based Personal Machine Learning Workspace

Thanh Loi Hoang, Thai Nguyen Duc Thong, Young Han Kim*

Soongsil University

loiht2@dcn.ssu.ac.kr, thainguyen1309@dcn.ssu.ac.kr, *younghak@ssu.ac.kr

Abstract

In the field of Machine Learning research, the availability of a dedicated and isolated workspace is critical for maintaining focus, efficiency, and data integrity. Environments simplified to include only the required tools and resources help researchers significantly by eliminating possible distractions and lowering system complexity. Furthermore, isolating work areas is essential to accommodate datasets, workflows, and experimental needs of individual scientists, hence maintaining reproducibility and data security. This study presents a design of personalized, cloud-based workspaces specifically tailored for Machine Learning researchers, enhancing both productivity and the overall research process.

Ⅰ. Introduction

Cloud computing refers to the on-demand provision of computing services such as servers, storage, databases, software, and networking via the Internet, enabling users to remotely access, manage, and store data without reliance on local infrastructure [1]. Machine Learning (ML), a subset of artificial intelligence, enables algorithms that allow computers to analyze patterns and historical data to execute tasks typically requiring human intelligence, including decision-making and problem-solving, without explicit programming [2]. Modern Machine Learning (ML) operations [3] systems often demand significant computational resources, storage, and high-speed networking to handle massive data and train sophisticated models [4]. However, standard shared or locally deployed environments usually present challenges such as hardware requirements, resource allocation, and scalability. Dedicated workspaces tailored especially for ML tasks are important because they help to minimize distractions and streamline workflow by providing access only to the relevant tools and resources. Isolation work environments are also essential because each researcher generally operates with distinct data, tasks, and objectives. To address these issues, this study proposes a cloud-based infrastructure designed to deliver individualized and secure workspaces customized for the specific needs of ML researchers.

Ⅱ. System Architecture

This section will examine the architectural design of the system and its operational workflow in detail.

As shown in Figure 1, users initially gain access to the Central Dashboard via a specified URL, using their assigned authentication credentials, which include their account and password. After authenticating successfully, the system identifies user roles to provide the appropriate interface and functionality.

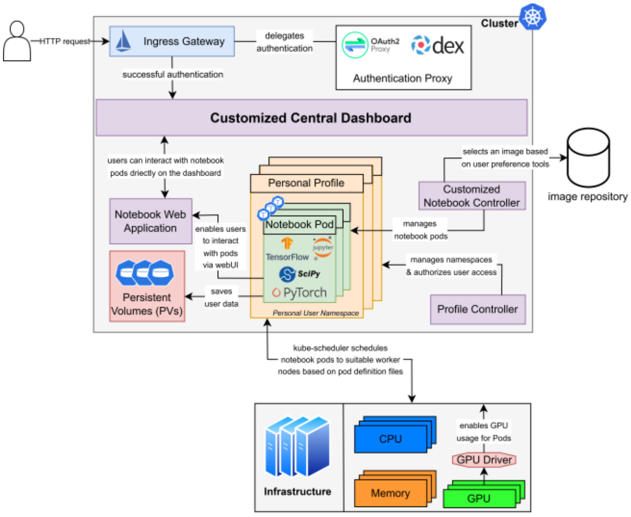
HTTP request
Ingress Gateway
delegates authentication
OAuth2 Proxy
dex
Authentication Proxy
Cluster
successful authentication
Customized Central Dashboard
users can interact with notebook pods driectly on the dashboard
Notebook Web Application
enables users to interact with pods via webUI
Persistent Volumes (PVs)
saves user data
Personal Profile
Notebook Pod
TensorFlow
jupyter
SciPy
PyTorch
Personal User Namespace
selects an image based on user preference tools
image repository
Customized Notebook Controller
manages notebook pods
manages namespaces & authorizes user access
Profile Controller
kube-scheduler schedules notebook pods to suitable worker nodes based on pod definition files
Infrastructure
CPU
Memory
enables GPU usage for Pods
GPU Driver
GPU

Figure 1: System Architecture

Specifically, users recognized as clients are prompted to activate their Personal Workspace; once accepted, they are redirected to their individualized environments. These personalized workspaces are equipped with selected tools and resources designed to help users perform their tasks efficiently. Otherwise, they are routed to their primary portal, where they may autonomously provision a dedicated workspace tailored to their professional requirements. By merely selecting the requisite tools and the tasks they intend to perform, the underlying system automatically instantiates a working environment optimized for their needs. Meanwhile, users identified as administrators are directed to an administrative control panel, provisioned with comprehensive management functionalities including, but not limited to, storage management, user permissions settings, and various system administrative tools tailored to support effective system maintenance. Upon activating a personal workspace, the system will allocate resources from the underlying infrastructure to the workspace. In detail, a dedicated GPU card and other resources like CPU and memory will immediately be assigned to their workspace to support their specific tasks. This allocation ensures that each user has access to the necessary computational power for efficient work. Similarly, this will happen for other users when they activate their workspace successfully. Ultimately, the resources will be released when the users log out of their sessions. Furthermore, users are able to upload their data to the individual workspace, while any workspace-generated data is retained during their work, providing consistent and continuous access.

Technically, we plan to use Kubernetes as an orchestrator. In the proposed architecture, the Istio Ingress Gateway [5] functions as the primary ingress point for all external HTTP traffic. Beyond this role, it also routes users to the appropriate endpoint that the system exposes through its web interface. More detailed, when a user initiates a session, the first HTTP request is delivered to the Istio Ingress Gateway, which then delegates the authentication procedure to an Authentication Proxy. Within this proxy, OAuth2-Proxy [6] is integrated with Dex [7] to deliver authentication services: Dex operates as an OpenID Connect (OIDC) provider, validating user identities against back-end identity sources, while OAuth2-Proxy acts as a reverse proxy that safeguards applications by leveraging Dex for user verification. Once authentication succeeds, the system redirects the user to the Central Dashboard according to their assigned role: administrators or normal users. To implement the personal workspaces, we will leverage the Customized Notebook Controller to deploy each notebook as a pod. Therefore, the admins can specify resource requests and limits, also determine the number of GPU cards to assign to each notebook pod, then the default scheduler in Kubernetes will schedule that pod to an appropriate node with adequate resources, as well as prevent any single notebook from consuming excessive resources that could affect others. Moreover, support from GPU drivers provided by several vendors is important to interface with GPUs, enabling notebook containers to access GPU resources, which can be requested in pod specifications. Additionally, this configuration enables the customization of the underlying images, which form the foundation of each pod. Furthermore, during notebook-pod instantiation, if the image initially designated for the pod fails to meet the required criteria, the Notebook Controller automatically retrieves an alternative image from the repository that aligns with the user's declared needs. The Notebook Web Application provides a browser-based interface that facilitates real-time, authenticated communication between end-users and their notebook pods. Through the Central Dashboard, users can launch, monitor, and execute code within these pods directly, all while leveraging Kubernetes-level isolation and resource governance. Each user is provisioned with a dedicated namespace. This namespace is linked to the appropriate Kubernetes RBAC roles and bindings - collectively referred to as the user's profile. The Profile Controller then automates namespace creation, role binding attachment, and resource-quota allocation in accordance with the declared profile specification. User data and session-generated data will be securely stored in persistent volumes, allowing data persistence and continuity, and protecting data from being lost when the notebook pods are terminated and restarted.

Ⅲ. Future work

In the future, the proposed architecture will initially be implemented within a single cluster to evaluate its performance and feasibility. Upon successful testing, the architecture will be deployed on a larger scale across multiple clusters to accommodate increased user demand. Future research will also investigate advanced GPU scheduling methodologies aimed at optimizing resource allocation and preventing wastage, ensuring system efficiency as user demand expands. Moreover, monitoring mechanisms will be enhanced to capture detailed metrics related to CPU and GPU utilization, resource tracking, and associated operational costs, thus facilitating comprehensive performance assessment and optimization.

ACKNOWLEDGMENT

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT)(RS-2024-00398379, Development of High Available and High Performance 6G Cross Cloud Infrastructure Technology)

REFERENCES

[1] R. Buyya, J. Broberg, and A. Gościński, *Cloud Computing: Principles and Paradigms*. Hoboken, N.J: Wiley, 2011.

[2] I. E. Naqa, R. Li, and M. J. Murphy, *Machine learning in radiation oncology: theory and applications*. Cham: Springer, 2015.

[3] D. A. Tamburri, "Sustainable MLOPs: Trends and challenges". *2020 22nd International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, 2020.

[4] C. Huyen, *Designing Machine Learning Systems: An Iterative Process for Production-Ready Applications*. O'Reilly Media, Inc., 2022.

[5] Istio, "Traffic Management", Available Online at https://istio.io/latest/docs/concepts/traffic-management/, Last Accessed on May 2025.

[6] OAuth2 Proxy, "OpenID Connect", Available Online at https://oauth2-proxy.github.io/oauth2-proxy/configuration/providers/openid_connect/, Last Accessed on May 2025.

[7] Dex, "Documentation", Available Online at https://dexidp.io/docs/, Last Accessed on May 2025.