CourseCompare Distributed

To transform the CourseCompare term project into a distributed computing system, we first containerize the program using Docker. This involves packaging each component of the application into separate containers, ensuring portability and consistency across different environments. To internally set up the environment for running the project, a requirements file will include the dependencies for the project, including Django, Flask, and cryptographic libraries. Additionally, we modify the program to integrate with another service responsible for validating login details and managing the user credential database.

After containerization, we adopt Kubernetes as our orchestration tool to manage the deployment and scaling of our containers. Leveraging Helm, a package manager for Kubernetes, simplifies the deployment process by using Kubernetes manifests. These manifests define the desired state of our application, including the number of replicas for the Python Django app and the configuration for exposing internal services.

Furthermore, we tailor our Dockerfiles to Kubernetes manifests format, preserving essential startup commands and exposed ports for the Python Django app. The Kubernetes manifests specify the deployment of two replicas of the Python website and the creation of a service to expose the ports on these replicas. Additionally, we define an ingress Kubernetes resource in a manifest to act as a load balancer, distributing traffic to the website deployment pods and providing users with a domain name to access the site.

For the backend, we implement a distributed MySQL database, leveraging Kubernetes volumes to ensure data persistence. This distributed database is partitioned based on different tables, such as Courses and CustomGroup, CourseTaking, and MyCustomUser, each residing on independent MySQL servers. This partitioning scheme enhances scalability and optimizes database access as traffic increases and the databases grow. Finally, we modify the program to utilize the DNS names of the MySQL server pods for database connectivity.

Kubernetes, Helm, and Docker were chosen as the app suite for containerizing and distributing this project because one of our team members has experience dealing with containerized applications using those apps. Additionally, Kubernetes is a powerful tool for docker orchestration, and large scale deployment and termination of large projects, with a more comprehensive yet easier networking ability than Docker Compose.