A Subscription-Based Online Bookstore

Group Members:

1. Chinmayee Dharmastalam Sreedhar (dharm052@umn.edu)
2. Matthew Folefac (folef001@umn.edu)
3. Lavanya Radhakrishnan (radha057@umn.edu)

Abstract:

This project focuses on developing a digital textbook platform that provides users with seamless access to a variety of educational resources. The system leverages cloud storage concepts for efficient and secure storage of textbook content, ensuring scalability and reliability. Users can create personalized accounts, enabling them to access their designated textbooks and related materials. The storage system used is a key-value persistent database system developed in Python. The frontend users will interact with the backend using RESTful API's.

Motivation and Problem Statement:

This project aims to provide a cost-effective option for college students to access textbooks. Over the past decade, the cost of college textbooks has seen a significant increase. The average cost of college textbooks during the 2020-2021 academic year was over $1200 **[1]**, with some students stating they spent an average of $300 to $400 per semester on textbooks **[3]**. With the rise of cloud computing services, some institutions have seen positive effects with the use of cloud computing to store online textbooks and could use computing services to teach **[4] [5]**. During the COVID era, there was a higher engagement with online textbooks and online resources, and this trend has persisted in the post-COVID era **[4]**. Open-source platforms such as Openstax are trying to assist students with free access to high school and college textbooks **[2]**. Similar to such platforms, our project will utilize the benefits of cloud computing to provide students with access to a wide variety of college textbooks at a low cost.

Related Work:

There are a number of key-value databases available in the market which are open-source and have user-friendly API's to interact with. A simple key-value in memory database can be implemented in Python using dictionaries. More complex systems use Tries for faster querying **[6]**. Indexing, load balancing, duplication, sharding and fault tolerance are a few features that are important and have to be addressed when developing such a system **[7]**. A few systems have also implemented ElasticSearch to find the necessary key faster and reduce latency. For the data to be persistent, the database has to write the files to hard drives. The application that is being developed is more read-heavy. The need for availability and partitioning takes priority over consistency **[8]**.

Redis will be used to reduce the latency of our application, which would increase the performance of our application. It allows for the caching of frequently used books through a key-value store system that can quickly retrieve high frequency books **[9]**. In addition, Redis can also be used as a database management system. This all in one approach makes Redis a much more suitable system to use than other caching methods such as Riak or Memcached **[10].** However, Riak is better at fault tolerance and has a lower risk of data loss. Those advantages are traded for higher latency and performance when compared to Redis **[10]**. Overall, with the introduction of fault tolerance, load balancing and reducing the risk of data loss, Redis will be more suitable for our use case. The high read rate and low latency fits the needs of our project.

Kubernetes is a popular container orchestration tool among practitioners for automatically managing and deploying containers. They assemble multiple computers, whether virtual machines or bare metal, into a cluster to efficiently run containerized workloads. Its ability to manage workloads of all sizes has made it widely adopted in both cloud and data centers **[11]**. Kubernetes optimizes hardware resources, reducing costs while ensuring high availability without downtime. It offers robust security features and efficient resource management, keeping infrastructure secure and well-utilized. However, developers face challenges with Kubernetes due to its complex, unfamiliar concepts and multi-step deployment process, which requires containerization, CI/CD pipelines, manifest creation, and network configuration **[12]**. While topics like efficient resource utilization are thoroughly researched by the academic community, other areas where practitioners encounter challenges, such as the learning curve, maintenance, and testing, remain less explored. **[13]**.

Cloud computing enhances library services by providing scalable infrastructure, remote access, and cost efficiency. It enables collaboration, data sharing, and robust disaster recovery. Libraries can integrate innovative technologies like AI and offer personalized services using cloud-based analytics **[14]**. In a typical online library system**,** the cloud service structure consists of three layers: user, cloud service, and resource layers. The resource layer manages both physical and electronic resources, while the IaaS layer handles resource storage, operation, and allocation. The PaaS layer organizes, stores, and delivers library resources, offering customized services. The SaaS layer provides software services over the Internet, allowing users to rent software, with the service provider handling maintenance and management, while library administrators focus on application security **[15]**.

Description of the solution:

1. The application is going to be built using an object storage database to store the textbooks. Design the databases. We aim to store the pdf files and the metadata in a key-value storage. This will be developed in Python. Performance mechanisms like load balancing, fault tolerance, availability, reduced latency and caching will be implemented.
2. Caching using the Redis database will be integrated into the application. This will help in faster retrieval of data, therefore reducing latency.
3. To interact with the database, RESTful API's will be used and will be coded in Python. Rate-limiting and other security mechanisms will be implemented to ensure security of the application.
4. The application will then be containerized and deployed on Kubernetes for orchestration and scaling.
5. The frontend of the application will then be designed and integrated with all of the other modules.
6. Benchmark tests for different performance measures like latency, throughput, high availability and fault tolerance will be done to measure the performance of the application. Any optimizations to improve the performance can be done permitting we have enough time.

Timeline:

Building the user and textbook database	Sep 27, 2024 - Oct 15, 2024
Build the caching module	Oct 16, 2024 - Oct 22, 2024
Setting up API's to interact with the storage system	Oct 23, 2024 - Oct 30, 2024
Implementation of Docker and Kubernetes	Oct 30, 2024 - Nov 9, 2024
Frontend user interaction module	Nov 10, 2024 - Nov 15, 2024
Benchmark tests and optimization	Nov 16, 2024 - Nov 28, 2024
Report and Presentation	Nov 29, 2024 - Dec 3, 2024

References:

[1] Education Data Initiative, "Average cost of college textbooks," *Educationdata.org*, Sep. 20, 2023. [Online]. Available: https://educationdata.org/average-cost-of-college-textbooks. [Accessed: Sep. 23, 2024].
[2] OpenStax, "OpenStax," *OpenStax*, [Online]. Available: https://openstax.org/. [Accessed: Sep. 23, 2024].
[3] L. R. Wittkower and L. S. Lo, "Undergraduate student perspectives on textbook costs and implications for academic success," *Open Praxis*, vol. 12, no. 1, pp. 115–130, 2020. [Online]. Available: https://search.informit.org/doi/10.3316/informit.219347356277982. [Accessed: Sep. 23, 2024].
[4] E. Castro-Rodríguez, A. Mali, and V. Mesa, "University students' engagement with digital mathematics textbooks: a case of linear algebra," *International Journal of Mathematical Education in Science and Technology*, vol. 55, no. 9, pp. 2293–2315, 2022. [Online]. Available: https://doi.org/10.1080/0020739X.2022.2147104. [Accessed: Sep. 23, 2024].
[5] P. K. Dunn, E. A. Brunton, and M. B. Farrar, "Your online textbook is ready: a shareable, interactive online textbook in response to COVID-19 lockdowns," *International Journal of Mathematical Education in Science and Technology*, vol. 53, no. 3, pp. 582–593, 2021. [Online]. Available: https://doi.org/10.1080/0020739X.2021.1983051. [Accessed: Sep. 23, 2024].
[6]
https://amsayed.medium.com/nosql-databases-a-guide-with-python-code-snippets-key-value-databases-26d77e9c49b
[7] https://www.influxdata.com/key-value-database/
[8] https://nikasakana.medium.com/how-to-design-a-distributed-key-value-store-cfd83248541b
[9] L. Zhou, B. Lu, S. Zhang, and L. Qi, "Data Cache Optimization Model Based on HBase and Redis," in *Proceedings of the 3rd International Conference on Data Science and Information Technology (DSIT)*, Xiamen, China, Jul. 24-26, 2020, pp. 31-35, ACM, 2020. [Online]. Available:
https://doi.org/10.1145/3414274.3414279&#8203;:contentReference[oaicite:0]{index=0}
[10] R. Gupta, P. Chhillar, and N. Agarwal," Supply of a Key Value Database Redis In-Memory by Data from a Relational Database" *Journal of Computers*, vol. 14, no. 3, pp. 137–144, 2019. [Online]. Available: http://www.jcomputers.us/vol14/jcp1403-03.pdf. [Accessed: Sep. 23, 2024].
[11]
https://medium.com/@rphilogene/what-is-kubernetes-what-you-need-to-know-as-a-developer-674af25e3947.
[12] https://en.wikipedia.org/wiki/Kubernetes.
[13] S. Shamim, J. Gibson, P. Morrison, and A. Rahman, "Benefits, Challenges, and Research Topics: A Multi-vocal Literature Review of Kubernetes," 2022. [Online]. Available: https://arxiv.org/pdf/2211.07032. [Accessed: Sep. 26, 2024].
[14] https://billingplatform.com/blog/cloud-based-subscription-management-benefits.

[15] S. Panda, S. Hasan, and N. Kaur, “Enhancing Library 5.0: Leveraging Cloud and Fog Computing for Intelligent Services and Resource Management,” in *Proceedings of the International Conference on Next-Generation Digital Technologies (ICNGDT 2024) on Global Digital Horizon: Innovations and Insights for next Generation Technologies,* 2024.[Online]. Available: Enhancing Library 5.0. [Accessed: Sep. 26, 2024].