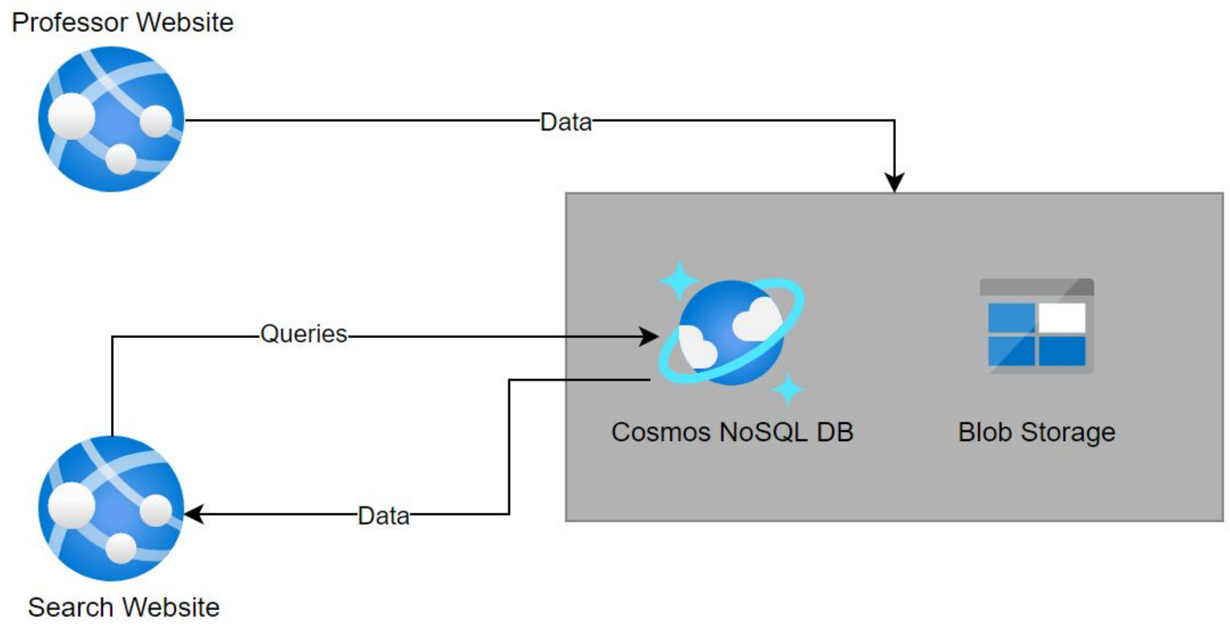
Simon Aronsky
02 / 27 / 2021

Program 4 Report

Location of the URL for you site:
https://program3websitestorage20210228004359.azurewebsites.net

Location of where the blog/object is stored (make this public). We'll check to make sure it is loaded there:
https://simoncss432.blob.core.windows.net/program4container/peopleblob

Clear design diagrams of final project



Discussion of how the site will scale with load

As the project is currently constructed, it is not built to automatically scale with load. This is because I have chosen to use the free tier of the App Service. If I was expecting a large increase in traffic to my page, Azure provides service plans that allow for automatic scaling. Production plans include Auto scale, which allows up to 10 instances to be created as needed to which the load is evenly balanced. I could also couple this with a scale up, improving the individual computing hardware.

Unlike the front end system, Cosmos DB tables and Blob Storage don't scale quite as easily. The tables can access up to 20,000 entities per second and hold up to 500 TiB, but at the current setting can only handle up to 400. The blobs can also process up to 20,000 requests per second but can store up to 5 PiB. Should I need to access more than that, according to the documentation I should create multiple storage accounts and partition my data across these multiple accounts. If this became a necessity I would design my application to automatically create these accounts as necessary. I am also able to replicate the data globally, which would ease load on each individual database instance should my page become popular worldwide.

Discussion on how monitoring is done on the site

Similar to the load scaling, due to the simplicity and frugal nature of this project, there is no monitoring built in. Should I choose to change this I would need to configure the four components of Azure Web Application Monitoring.

The first is Azure App Service portal. Here I can view resource usage, app metrics, and logging. This component is managed for me, and comes with the establishment of the App Service.

Next are the Application Insights. According to the documentation, this service “monitors the application, detects application anomalies such as poor performance and failures, and sends telemetry to the Azure portal”. Using this anomaly detection, alerts can be set up to notify me of unusual activity or specified metrics.

The Azure Monitor inspects the metrics and logging for most services in Azure. This information can be quirried via REST API calls or using the command line library.

Finally, Log Analytics which combines data from the Application Insights with performance data from connected Azure services. Similarly to the Azure Monitor this data can be queried via a REST API or using the command line library. Alerts can also be created using a preprogrammed query that is run at regular intervals.

Estimate of your SLA

Free Tier App Service: No Guarantee 99.95% Subscription
Cosmos DB: 99.99% for a single region
Storage Accounts RA-GRS Hot storage: 99.99% Read 99.9% Write

To calculate the SLA of my service I referenced the Azure documentation for the SLA of the individual services I used and calculated accordingly. The free tier of the App service has no guarantee, so in this example I will assume I am using the paid subscription. With the subscription, my website has 99.95% availability. I will also assume that the page from which we collect the data, hosted by the professor, has the same availability. Regarding the data storage, the Cosmos DB has 99.99% availability and the Blob storage 99.99% for reading and 99.9% for writing, with my RA-GRS hot storage. Given that the success of the blob storage shouldn't have any effect on the user experience, because the data loaded to and queried from the NoSQL DB, I will not include it in my overall calculation.

This puts my SLA at 99.95% x 99.95% x 99.99% = 99.89%.