ONLINE DATABASE PROJECT

CSS 436 – Program 4

Eric Feldman

November 17th, 2019

Project Overview

This project uses multiple cloud services to enable a user to access a hosted website to upload, clear, query a database. The three services used for this project are: Azure App Services, Azure Blob Storage, and Azure Cosmos DB. The website technology used is Node.js.

Website: https://eric-feldman-program4.azurewebsites.net/
Object storage: https://ericeldmanstorage.blob.core.windows.net/program4file/data.txt

- The object storage is also accessible through the website's main page through the button labeled *Data Object Storage.*

Utilization

- **Load Data Input Box –** This input box enables a user to type in a URL link to load data from object stored at a certain URL.
- **Load Button –** This button kicks off a process which fetches the data object at the URL and parses the file, stores a copy of the file, and populates a Cosmos DB table with the data.
- **Clear button –** This button kicks off a process which clears the copy of the file stored in Azure blob and clear the Cosmos DB table of all data.
- **First Name Input Box –** This case-sensitive text input box enables a user to input a first name to query the database.
- **Last Name Input Box –** This case-sensitive text input box enables a user to input a last name to query the database.
- **Query Button –** This button kicks off a process which will query the database for either all matches of the first name, all matches of the last name or all matches of both first and last names given.
- **Data Object Storage Button –** This button will go to the data object file storage and download the file being stored there.

Design

This programming assignment follows a 3-tier architecture. The presentation tier consists of the HTML. The application tier includes the server-side logic running using JavaScript/Node.js. Lastly, the database consists of Azure's Cosmos DB table and blob storage.

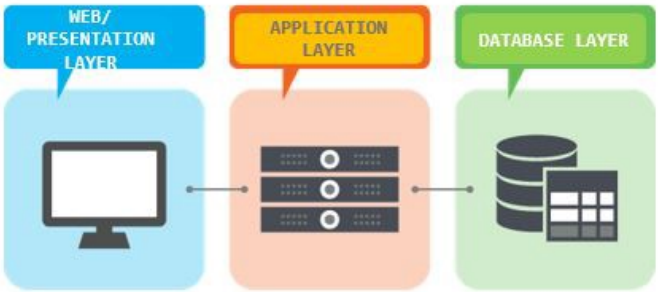


Figure 1

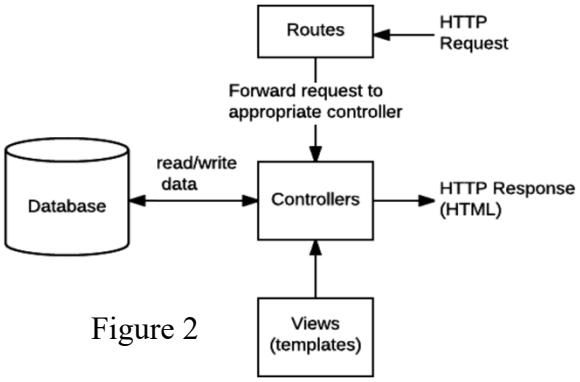
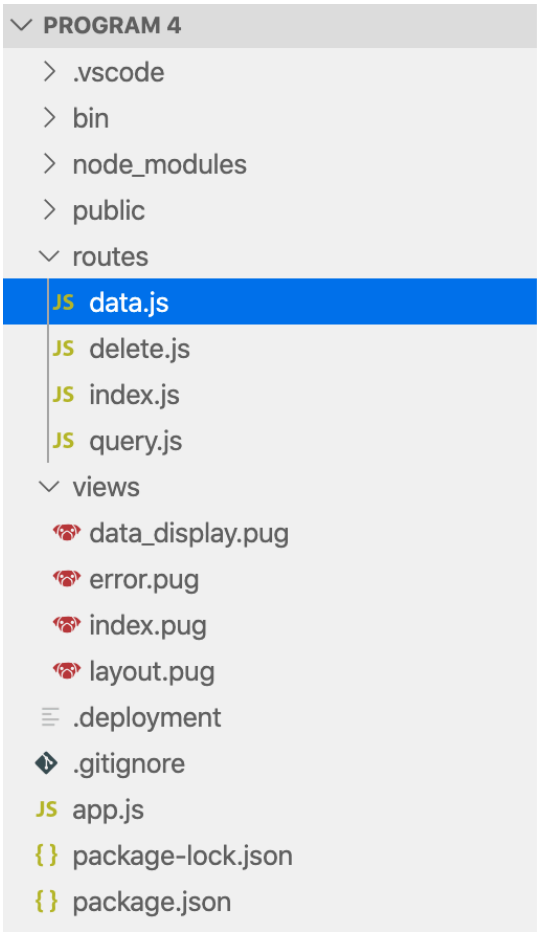


Figure 2

Figure 1 is a visual diagram of a 3-tier architure, which this program takes advantage of as descirbed previosuly. Figure 2 is a diagram which helps show how Node.js is used to build out the codebase. The controller is the center brains of the operation. HTTP requests are funneled through routes which are then forwarded to the logic controller. When the controller is ready to render page to the user it grabes the view templates and displays the HTTP response using HTML to the user in a broswer. Lastly, if there is a HTTP requests which needs to access the database, the route specifies that information to the controller and the data is pulled from the database to controller to use it.

Below in Figure 3 is the file strucuture overview of the project



The three routes the program utilizes is data.js, delete.js, index.js, and query.js.

The four views the program utilizes are data_display.pug, error.pug, index.pug, and layout.pug. Pug is a file format that makes writing HTML very user friendly.

There is a CSS file for formatting the HTML. It is located under public/stylesheets.

App.js is the main/root file of Node.js which powers the application.

Scaling with Load

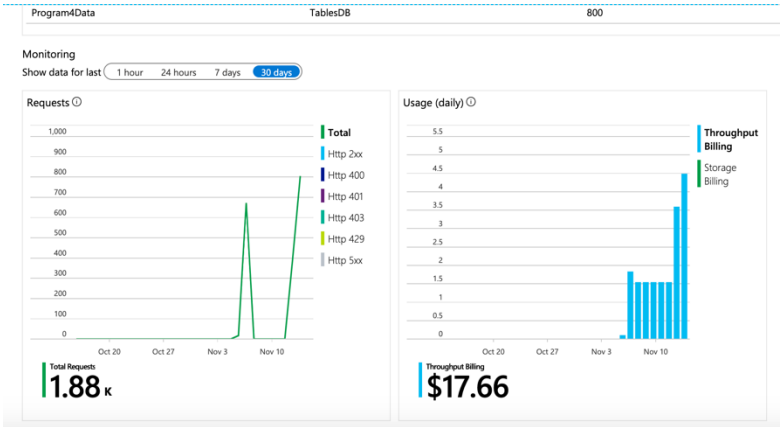
Azure web services enables you to scale your web service in two different ways. You can scale your web service either up and/or out. Scaling out enables you to get more CPU, memory, and disk space. Scaling out enables you to increase the number of VM instances that run your app service.

Azure Cosmos DB can be scaled in serval ways as well. The provisioned throughput (requests/min) can be increased to handle more requests. You can also globally scale the throughput by adding more regions. Cosmos DB tables can grow as big as you need them too, so there is no need to be concerned with how many items you are storing.

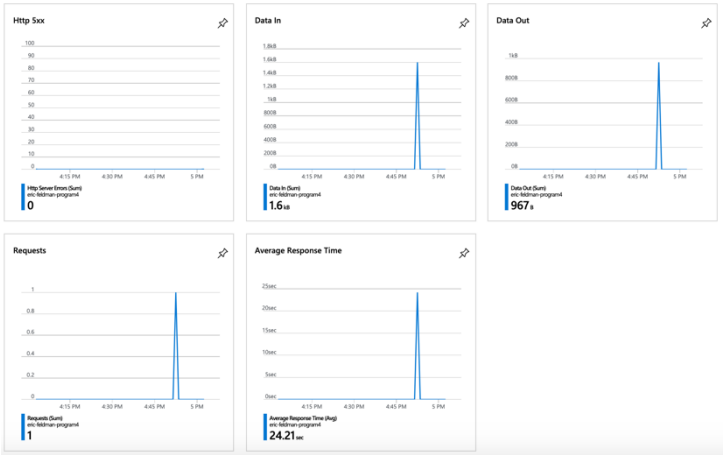
Azure blob storage is only being used to hold one single file. The basic account type can handle 20,000 requests/second, and if needed the account type can be upgraded to a Premium performance storage account for example. This account type can handle 100,000 requests/second.

Monitoring

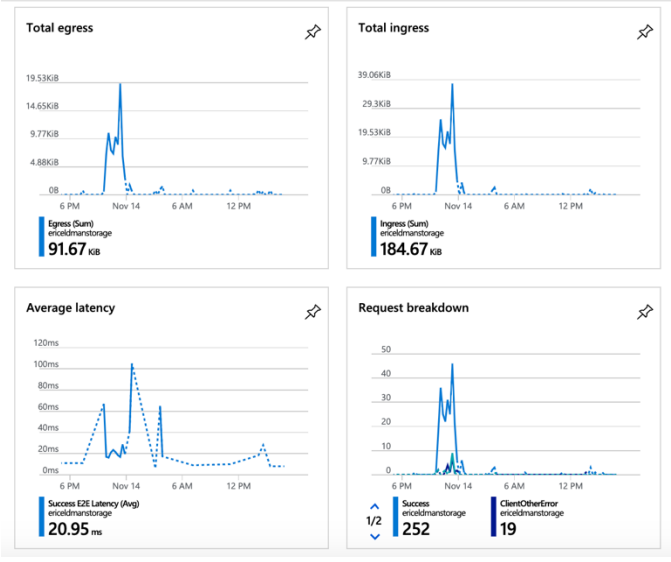
Below are screenshots of how each of the services being used can be monitored.



Azure Cosmos DB Table



Azure App Services



Azure Blob Storage

SLA Estimate

Azure App Services

- No SLA is provided for the free-tier, but for this calculation I am assuming I will assume I am running the standard account type which provided 99.95%

Azure Cosmos DB

- 99.99% for single Azure region regardless of other configurations.

Azure Blob Storage

- 99.9% for standard Locally Redundant Storage account types.

Overall SLA:

- 0.9995 * 0.9999 * 0.999 = 0.99840065
- Overall SLA for this program is **99.840065%**