**Reservation Management System**

**Cloud Computing Systems Project**

**2020/2021**



**Authors:**

António Beirão

Henrique Realinho

Nuno Pereira

**Introduction**

The proposed project was challenging and very interesting to us, allowing us to explore some important cloud concepts and models, while developing a system that is pertinent and useful.

We implemented all the services that we set out to implement. The project was not implemented for the full grade (20), given that we did not implement support for advanced search or computation with Cognitive Search or Spark, and also did not test for geo-replication, though we tried to implement it. We tried to begin using the advanced search feature for our Forum messages but ended up not being able to fully use it.

**Design**

Our project was designed using 4 main resources: Entities, Forums, Calendars and Reservations. Each of these resources has its own container in Cosmos DB. We also have a blob storage container for the Media images. Each Entity can hold multiple media files, but only one forum and one calendar. Each Calendar has the information on the respective Reservations, only allowing one Reservation per day.

**Implementation**

We used Java with JAX-RS for the REST endpoints.

We separated each resource into its own Interface, Controller and Service. A resource’s interface is the resource’s API which defines the resource’s endpoints, the controller is the class that send data to and from the services, with the service being the main logic of the resource. It is the service that accesses the databases and the cache, retrieving and processing the data.

We used Redis caching for all resources except Media, meaning that in each GET for every resource we first try to retrieve the data from the cache, and only if the data is not there the database is accessed. In each create and update operations we put (or update) the data from the cache.

The server is deployed in the West-Europe Azure availability zone.

We implemented two azure functions. One with a Timer Trigger, which runs every 24 hours and updates the available days on each calendar, subtracting one day (the day that just passed) from the available days list that each calendar holds. The other Azure Function implemented is Blob Store function that runs whenever a new blob is inserted into the system. As we only use the blob storage for the Media resource, this Azure Function runs whenever a new media file is uploaded into the system and it takes that media file and replicates it into another availability zone (US central).

The Cosmos DB resource is in the West-Europe region, geo-replicating data to the East-Europe and US Central Azure regions, with multi-master.

We set the consistency level to SESSION in all our Cosmos DB containers. We decided to use the SESSION consistency level as it is appropriate for our use case. The trade-off between the SESSION consistency level and a stronger one are just not worth it for this use case, though obviously a STRONG consistency level would be ideal for the reservations container at least, but given that it could, and in most cases would, lead to a big decrease in performance, we opted for the SEASON level. As for a weaker level, one could be used in the entities or forums containers possibly (CONSISTENT PREFIX), but we thought the SESSION level was still the most appropriate and reasonable one, given the performance trade-off.

The database is using Last Write Wins conflict resolution, as we think it is appropriate for our use case, with no need to write a custom merge procedure.

**Evaluation**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Delete** | **Delete** | **Get** | **Get** |
|  | **Entity** | **Forum** | **All entity** | **Entity** |
| **With Caching** | **332** | **272** | **218** | **343** |
| **No Caching** | **269** | **250** | **321** | **295** |
| **With Cache USA** |  |  |  |  |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Media** | **Entity** | **Media and Entity** | **Calendar** | **Forum** | **Message** | **Message Reply** |
| **With Caching** | **332** | **119** | **218** | **343** | **254** | **135** | **181** |
| **No Caching** | **347** | **65** | **174** | **295** | **258** | **144** | **209** |
| **With Cache USA** |  |  |  |  |  |  |  |

**Caching**

Get entity: 221 ms

Delete entity: 233 ms

Post calendar: 136 ms

Get calendar: 147 ms

Add nessage

Delete calendar: 107 ms

DELETE forum

Add forum

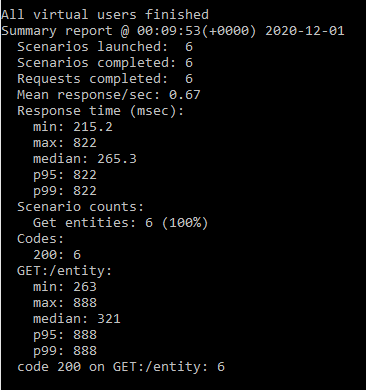
**No caching**

**20 users add entity and add media**

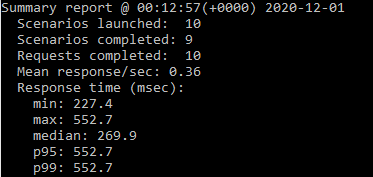
Add\_message

Add message\_reply

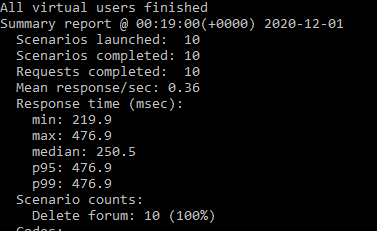
Get-all Entity



Delete entity



Delete Forum



**Conclusions**