**Propose a simple architecture for handling high traffic and CMS sync. Could be a markdown diagram or short doc.**

**Solution 1**

* Client sends a request to the endpoint. The Endpoint could be a API Gateway or Web App which is knows as API Layer.
* API Layer which has the controller and other Cache based reads lets say we have Redis cache implemented.

If key is available, served cache page  
 if key is not available query CMS and then refresh cache

* Redis handles chunk of the reads so high traffic resilience is maintained.
* We can scale out API Layer horizontally for elasticity and high availability
* If CMS is unavailable Redis Cache can still serve the cached contents and if both fails them service is unavailable presented to the user.

**Solution specific to Sitecore**

This solution, I have implemented in the past using GraphQL queries using the following steps.

* Install the Sitecore Headless Services
* Create API key
* And by sending the GraphQL request you can directly query the Sitecore indexes which is great for performance. See the following query which returns all the events

**List All Events**

**query {**

**search(**

**index: "events\_master\_index"**

**)**

**{**

**results {**

**items {**

**path**

**name**

**id**

**fields{**

**name**

**value**

**}**

**}**

**}**

**}**

**}**

* Index strategy helps in syncing the indexes
* Even if CMS isn’t available we can still serve the requests from API