amazon

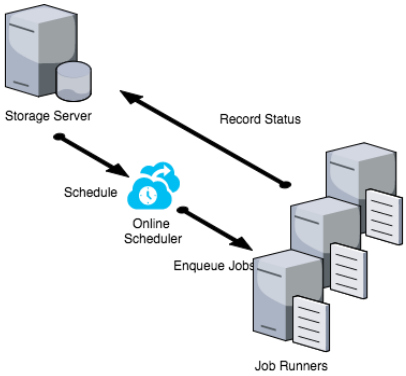
Job Scheduling System

Jane is an engineer on the data quality monitoring team. Her team processes large volumes of data periodically. During the problem review, Jane proposed creating a generic job scheduler that can satisfy the team's needs. After quick research, she came up with the following requirements that the system needed to meet.

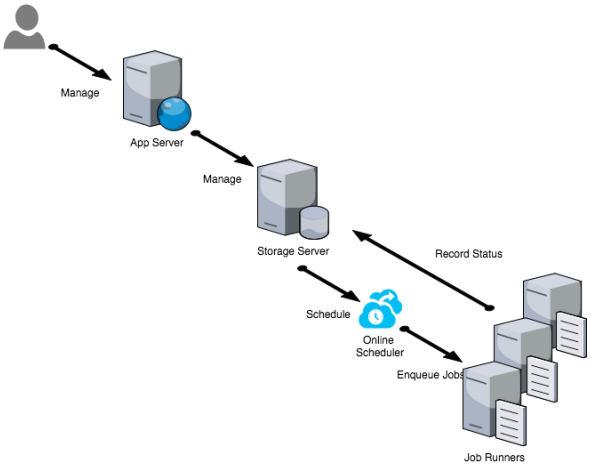
a. Must run a weekly batch job that needs to finish running by midnight of Tuesday each week. The job processes a week worth of data Sunday – Saturday and usually takes about 8 hours to run.
b. Must run a small job (3-4 minutes run time), that pulls data from a queue and processes it.
c. Stretch goal: Build a way to create and manage the jobs. We want to see which jobs are currently running and the status and execution history previously run jobs.
d. Stretch goal: Provide a mechanism to alert the team in case a job may not complete with the allotted time (or if the small job queue is too large).

For the weekly jobs, we will start with 1 job, but if we are successful, it may need to run up to 10k jobs each week. For the small jobs, we are looking at 500-1000 jobs / day for now, but will scale to 10k jobs / day

Initial Design: After a few days of research, here was Jane's initial design. A simple storage server that stores the job, with a scheduler that pulls the jobs and runs them on job runners. The job runners will report back status of the job execution back to the storage server. This high-level design meets the requirements (a) and (b). The DB may be a normal relational database or perhaps noSQL.



The design went through a few iterations based on feedback from peers and senior engineers. This design is flexible and can easily be scaled by adding more job runners, which are basic EC2 instances. Jane has not yet figured out how to manage the jobs. This is something she takes up as a follow up task.



Few iterations later: After spending more time on the design, she decides to add a UI layer in the form of an application server that integrates with a user management system and lets authorized users log-in, and based on the permissions of the user, enable creation of new jobs, managing the job schedules and view the status as reported in the database.

One of the key advantages of this design is that the UI layer is isolated from the job schedulers and runners, and each can be independently scaled based on needs. Jane's design is still not clear about how she plans to add user management and authorization. Ideally, the app server should integrate with the company's login servers to support single sign on (SSO) capability to make the application more user friendly.

---- Pause Here! ---

Before you proceed further, reflect on how the design has evolved. How would you approach this design? What feedback would you give Jane if you were reviewing it?

On the next page, we have included some questions that were asked by engineers who reviewed this design. Read them after you have spent a bit of time with the design above.

© 2017 Amazon.com, Inc. or its affiliates. All rights reserved.

Think about the following:

As you reviewed the design, were you able to spot the following:

a. Do you think the design meets functional requirement? Maybe, clearly, stretch goal (d) is not met. How do you think you can extend the design to support it?
b. What are the points of failure and how can we address them?
c. Do you think we need more than 1 app server? 1 db? 1 job runner? With 1000 jobs / day @ 4 mins each, it is 60+ hours and won't finish in a day. You will need at least 3 runners.
d. What would be your choices for various components?
e. What would it take to build this system?

In a typical design interview, you will be evaluated on one or more of the following aspects

- ❑ Did you ask clarifying questions to identify system requirements, limits and constraints?
- ❑ Did you come up with a high-level design, covering high level needs of the system?
- ❑ Did you connect the design with something you have delivered in the past?
- ❑ Did you identify examples of components that can be used, like which app server or which scheduler?
- ❑ Did you identify bottlenecks and challenges?
- ❑ Did you define the data model for jobs and the reporting output?
- ❑ Did you articulate your thoughts clearly without needing too much guidance?
- ❑ Was your approach structured, logical and easy to follow?

© 2017 Amazon.com, Inc. or its affiliates. All rights reserved.