WebMon

A simple website monitoring service

CS263 - Runtime Systems
Winter 2016, UCSB

Darshan Maiya (dmaiya@umail.ucsb.edu)
Suraj Rajesh (suraj_rajesh@umail.ucsb.edu)

Agenda

- Overview
- Motivation
- WebMon Architecture
- Google App Engine Services
- WebMon APIs
- Performance Optimizations
- Testing and Documentation
- Novelty
- Key Takeaways
- **DEMO!!!**

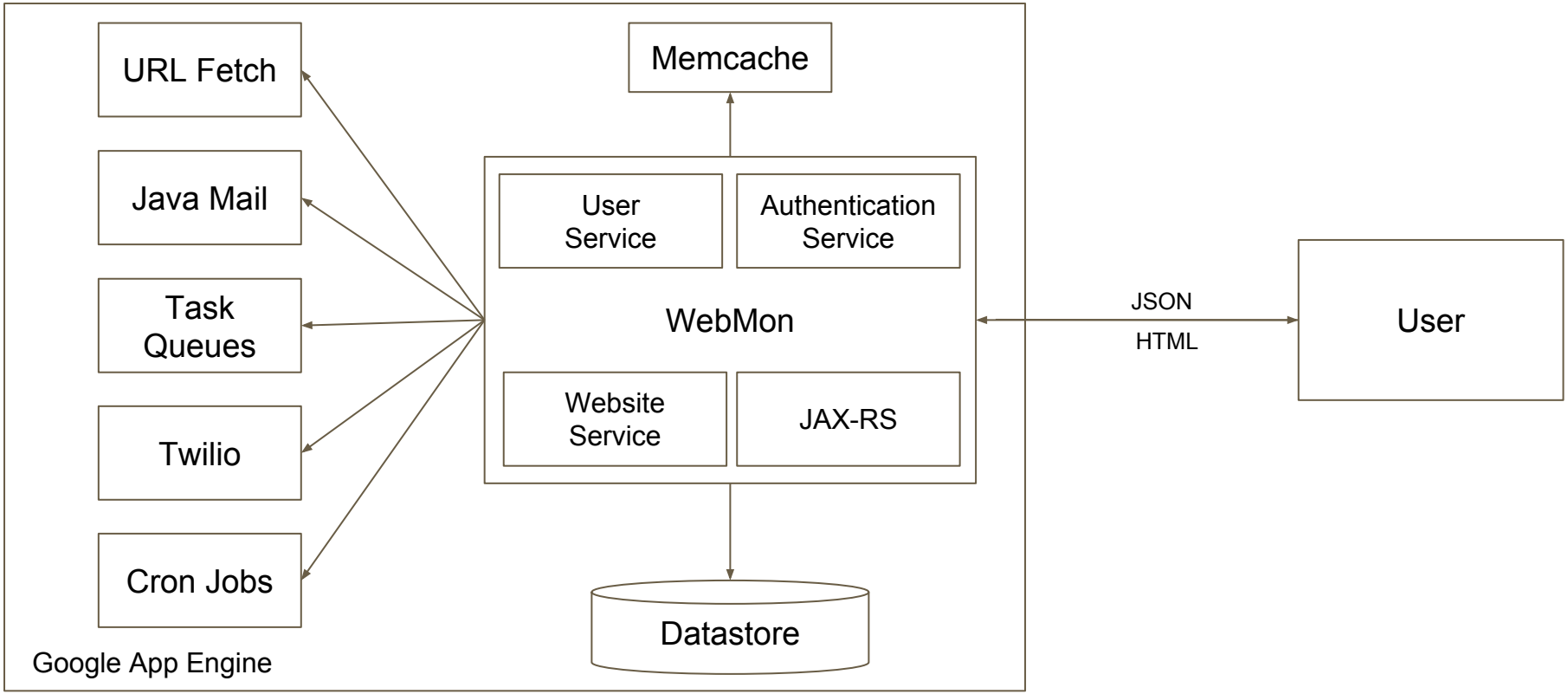
Overview

- WebMon continuously monitors a user chosen website
- Notifies the user if one of the websites being monitored experiences downtime or high response time
- Real time notifications via SMS and email
- Keeps track of response times and displays them in graph format

Motivation

- Websites may occasionally experience downtimes or high response times due to various reason
- Downtimes and high response times are bad for revenue and may turn users away from a website
- Keeping track of website downtimes and high response times is a difficult and tedious process
- Monitoring response times from a client perspective is important

WebMon Architecture



Google App Engine Services

- Datastore - low level API for storing user and website details
- Memcache - for temporary storage of user and website details
- Task Queues - for getting response time of a website
- URL Fetch - for sending request and receiving data from websites
- Java Mail - for notifying user via email
- Twilio (3rd Party Integration) - for notifying user via text message
- Cron Jobs - for invoking the task queue on a fixed interval basis

WebMon APIs

Users (REST): (GET/PUT/DELETE) https://cs263-webmon.appspot.com/webmon/users***

Websites (REST): (GET/PUT/POST/DELETE) https://cs263-webmon.appspot.com/webmon/websites

Non-REST Endpoints

(GET/POST) https://cs263-webmon.appspot.com/{login|logout|signup|error}

Task Queues

(GET) https://cs263-webmon.appspot.com/response - Called by Google cron service

(POST) https://cs263-webmon.appspot.com/pinger - Pings individual websites and records response time

** All requests to the REST interface deployed at /webmon go through the Authentication and Authorization Filter.
*** Since /webmon/users is protected by Authentication, POST to /webmon/users is handled by POST to /signup

Performance Optimizations

- All requests for data from datastore are routed through memcache
- Common website added by multiple users is monitored only once and is not connected to a particular user.
- URL Fetch API is used for efficient network fetch operations
- Centralized authentication service for performance and security

Testing and Documentation

JUnit

Sanity tests - website up/down, jersey deployed url (/webmon) down/up

Selenium

Firefox based testing for end to end scenario

Javadoc

~3500 lines of fully documented code. Javadoc is available in github repo

Novelty

- Simple web based website monitoring
- Real time notifications via email and text
- Historic data of response times for a website

Key Takeaways

- Experience of developing for the cloud on Google App Engine
- NoSQL is easy but dealing with eventual consistency is not
- App Engine APIs are fairly easy to use and well documented
- REST based web application development
- Web request handling via POJOs using JAX-RS

DEMO