Monitoring Web Applications: An Automated Approach

A Technical Report submitted to the Department of Computer Science

Presented to the Faculty of the School of Engineering and Applied Science

University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree

Bachelor of Science, School of Engineering

Sierra Shuman

Spring, 2023

On my honor as a University Student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments

Briana Morrison, Department of Computer Science

Monitoring Web Applications: An Automated Approach

CS4991 Capstone Report, 2022

Sierra Shuman
Computer Science
The University of Virginia
School of Engineering and Applied Science
Charlottesville, Virginia USA
sms6ss@virginia.edu

ABSTRACT

Solution Street, a software engineering firm based in Herndon, Virginia, found that its manual system of monitoring hosted web-based applications for issues such as downtime was inefficient, and they wanted to replace this system with a more streamlined automated process. To increase the efficiency of Solution Street's website monitoring, a team of interns and I created an internally used web-based application to periodically test the firm's websites for downness and response time. We met with our internal client to determine the requirements of the application and show progress, and we utilized an Agile process during the software development cycle. We designed the application from the ground up, including the user interface and database system. The application used a Hypertext Transfer Protocol (HTTP) application programming interface, hosted as a cloud application, to make calls to websites and wait for a response. The application, now in use at Solution Street, was able to check websites successfully and alert users in the case of any problems. Future maintenance of the application will require a number of improvements, including increasing the accessibility and clarity of the user interface, allowing for more functionality and more ways to test websites, as well as better analysis on website statistics (for example, the percentage of uptime).

1. INTRODUCTION

Imagine you are the leader of a development team with the task of maintaining a web application. It is late in the day on a Friday, and you get an angry phone call from your client. He tells you the website is down and has been for hours, he has gotten numerous complaints from dissatisfied customers, and he is losing money. Now your team has to work through the weekend to get the website back up and running, and you are at risk of losing the client entirely. How could this have been avoided?

All software maintenance involves fixing problems with the program, such as testing to find failures, patching bugs, and adding new updates. However, with web applications it is important to also be sure the website is always up and running, its response time is fast, and users can access it easily. If issues affecting these goals are detected early, it takes less time and money to fix it, resulting in a more positive developer-customer relationship.

Solution Street had been having issues with this. With as many websites as they maintained, they could not manually monitor them all fast enough. One would go down, and they would not know about it until annoyed users complained and their client noticed the issue. An automated monitoring process was necessary for Solution Street so they could stay informed on the status of their websites.

2. RELATED WORKS

Uptime is defined by Uptrends (2022) as the ratio of time a website has spent available ("up") over a certain amount of time. The industry standard is to have an uptime of 99.999% [1]. Additionally, even when a website is up and running, there are other performance considerations. According to a consumer survey completed by Gehrke and Turban (1999), the speed of a website is the number one complaint made by e-commerce website users, who are likely to leave the website if it is not responding fast enough [2].

My team looked at available website monitoring services for ideas on what to include in our application. SolarWinds (2022) touted an existing tool, Pingdom, a paid service that offers page speed monitoring, uptime monitoring, and real-time text and email alerts as services [3]. Another tool, Uptime Robot, (n.d.) offers SSL certificate monitoring in addition to uptime monitoring [4]. These existing services helped inform us about features that would be useful to include in our monitoring service for Solution Street.

3. PROCESS DESIGN

Solution Street, after realizing their manual process for monitoring website performance was inefficient, needed a new web application to monitor the websites the company maintains. They tasked my intern team to design and build a new system for them to use internally.

3.1 Gathering Requirements

Our team's client, on behalf of the company, met with us many times to discuss and refine the requirements for our web application. First and most importantly, the application needed to automatically perform health checks on the site periodically, with a time between checks specified by the user. Health checks were required to be able to test whether the website is up, the response time of the website when loading, and if the website is accurately displaying the expected data. If the check detects any problems, the bounds of which are specified by the user, the application needed to alert the user managing the site by email or SMS message. Additionally, the client requested that the application keep record of previous health checks and have analytical tools such as graphs so the user can monitor the health of a website over time.

3.2 Application Stack and Development

For the framework and programming language used for our application, my team chose to use Ruby on Rails. This framework lends itself well to beginner web developers, as it makes it quicker to get a web app up and running compared to other frameworks we were considering, such as Django.

For the user-facing frontend of the app, my team decided to use HTML, CSS, and the JavaScript library JQuery as languages since they are compatible with Rails and are widely used for web apps. To begin development on the app, we utilized these technologies to create a mockup version of the application that displayed the user interface without any data. We met with our client many times to ensure he approved of the visuals and workflow of the application before we moved on to develop the logic behind it.

To persist the data entered into the app, we chose PostgreSQL as a relational database management system. We designed the database from scratch, including database tables for HealthChecks, Tests, Responses, and others. Once we had both the user interface and database set up for the app, we linked them together so user input would be stored in the database. Before we got the functionality of the app working, we tested this storage by using "dummy" data.

After linking the frontend and backend, my team began to work on the actual functionality of the application. We used a library for Rails called httparty in order to simplify sending HTTP requests to websites. We used this library to develop the way the application performs tests during health checks, as the library returns a response that is checked by the application to be correct and expected. We used a Rails email library to allow the application to alert the user if the response of the test is not as expected. All the responses the application receives back from websites are stored in the database, so this data is what contributes to the analytics portion of the application, where we had multiple ways to display the data for the user.

4. RESULTS

After my team finished our internship, the application was able to successfully monitor websites automatically and alert users when their websites were not performing well enough. Some Solution Street employees who wanted extra help monitoring their websites started using the application by just before my internship finished. I know that the application has been developed even further and is still in use.

Website outages that used to take hours to detect manually are now detectable in minutes by project managers, since most users scheduled their health checks to run every five to sixty minutes. This reliable method of monitoring websites has benefitted the relationships Solution Street has with its customers, since problems in websites are being found and fixed much more quickly than before.

5. CONCLUSION

My internship experience helped me expand my skills as a software developer while solving a problem for a real company. The program's emphasis on an enforced development cycle introduced me to the development process in a real-world setting, and I was able to explore all of the various aspects of development such as user interface design, working with the database, and gathering requirements from the client. At the end of this process, my team and I had produced an efficient website monitoring tool that has proven useful and effective through Solution Street's internal use.

6. FUTURE WORK

Although my team was able to create a fully functioning application during our internship, numerous improvements can be made to it in the future. Future maintenance should include updating the user interface to be clearer and more accessible, since no one on my team had any background in user interface design and it was not a main focus during development.

More functionality could be added to the application, including new and more complicated types of tests to run and more robust tools for analyzing website statistics. We saw many types of website performance tests in currently existing software that were too complicated for us to implement before our deadline, but these can be added to the application later.

Additionally, the tools we included to track and analyze the status of the website were simple, such as an uptime percentage and a chart that displayed this data over time. There are more useful statistics that a user might like to see about their websites, however, since my team did not have enough time to conduct research on what would be practical, these will have to be implemented in the future.

REFERENCES

[1] Uptrends. 2022. What is website uptime? Retrieved from https://www.uptrends.com/what-is/website-uptime.

[2] Gehrke, D. and Turban, E. 1999. Determinants of successful website design: Importance and recommendations for effectiveness. *Proceedings of the 32nd Annual Hawaii International Conference on Systems Sciences*. DOI: https://doi.org/10.1109/HICSS.1999.772943.

[3] SolarWinds. 2022. Pingdom: Uptime, Website, and Performance Monitoring. Retrieved from https://www.solarwinds.com/pingdom.

[4] UptimeRobot. n.d. UptimeRobot: Free Website Monitoring Service. Retrieved from https://uptimerobot.com/.