A Cloud-native performance monitoring system

Group 5

Background

The client for this project is RISE - The research institute of Sweden. They are an organisation that works for sustainable growth in Sweden by supporting innovations in the business community. RISE is currently migrating to the cloud, specifically AWS, and would like to have an easy way to monitor the performance of their AWS virtual machines.

The project is about monitoring the performance-related metrics of a virtual machine and visualizing them by making use of AWS CloudWatch (AWS's built-in monitoring system).

Product Functionality

Key Features

The final product contains all the features first laid out by the clients requirements, spare a few. Features included are the dashboard and its graphs, visualizing the metric data gathered from AWS (Amazon Web Services). The graphs are able to be viewed in multiple ways (line-, bar- and pie-graphs) and are able to be hidden from view. These options can be selected from a menu on each graph.

The actual collection of metrics from the AWS API can be toggled on or off by an admin, that can be signed in through a login-page. Turning off the collection of a metric means that the dashboard will make no attempt to fetch data about that metric, and thus will have no dataset to construct a graph with.

Missing Features

When a graph is hidden from the dashboard, the user is forced to update the webpage to make it viewable again. There is no log file to be downloaded from the website, either. Currently the product lacks the ability to log multiple machines at the same time. Omitting these features was done because we just didn't have time to implement them.

Possible improvements

There are a multitude of improvements that are necessary to the project to make it a truly useful product. We would've liked to see the ability to add multiple virtual machines to get metrics from instead of as now, only collecting from a sole machine.
Naturally such an improvement would also require a removal of virtual machines from the admin page. This would have made the website feel much more dynamic and filled with more functionality.

Adding the ability to create new users/admins on the admin/configuration page would have been an immense upgrade in product usability. Likewise, saving the configuration of metrics onto cookies so that the state can be reapplied would have been a good improvement.

Log file downloading was something that we would have wanted to implement from the very beginning, but circumstances forced us to abandon it. Minor improvements to the design, such as adding a white-design allowing for creation of night and day -mode would have been a quality-of-life improvement.

The design should've also included a much better function for time. As it works right now you're forced to choose 30 minutes, one hour or four hours back in time, this should've preferably been completely dynamic allowing the user to choose any time he or she prefers.

Another improvement would've been the ability to show data of the specific virtual machine. This data could've included where it is being hosted, it's ID et cetera.

A major improvement would've been to build an application that would sit inside the virtual machine and collect data on every single process currently running, that would have allowed the user to see specific information on every single process, their state, running time et cetera. That improvement would have made the website/application very useful as it would not only be a front to the Cloudwatch API but in the end much more powerful than the Cloudwatch API. Such an application was built in the early stages of the project as a proof-of-concept, and given enough time it would have been perfectly possible to implement this feature into our project.

Acceptance test

We performed an internal acceptance test by the end of the project. This test included six different test cases (see Appendix A), that three of our group members conducted separately. Unfortunately not all test cases passed (as seen in figure 1), due to features not having been implemented in time, but the result of the test cases has been sent to the client.

ID	**Result**
1	Passed
2	Passed
3	Passed
4	Passed
5	Failed
6	Passed

Figure 1. Table of acceptance test results.

Changes

Requirements

The client did not diverge from the initial requirements. In fact the client hadn't set their own requirements in stone. Rather their requirements were often something that could be one way or the other. For example we were early on given the ability to choose between AWS, OpenStack or even write it from scratch.

The database was something we focused on early in the project. We thought it was an important feature and that the client wanted different kinds of users. This was later on changed thanks to vital feedback from the client, telling us that the ability to have different users was not an important functionality. As a result the priority of the database and functionalities regarding multiple kinds of users was set to low priority.

However the result of the indecisive requirements delayed us as we had to rethink our own position, while at the same time requesting additional information from the client.

Organization and routines

In the project plan we settled on using the SCRUM-method with a weekly meeting discussing the current and upcoming sprint, as well as daily check-offs to see how the group was progressing individually. This routine however, quickly failed. Instead we'd only have one meeting every week to discuss a multitude of things, such as documentation et cetera). During which sprint backlog would be a part of said meeting.

In regards to our quality assurance we also failed to allow the client to see the current progress in a weekly manner which would have given us (in hindsight) vital feedback.

The degradation of the initial plan and organization likely reduced our ability to shape the website design to the wishes of the client.

Project

Total project effort

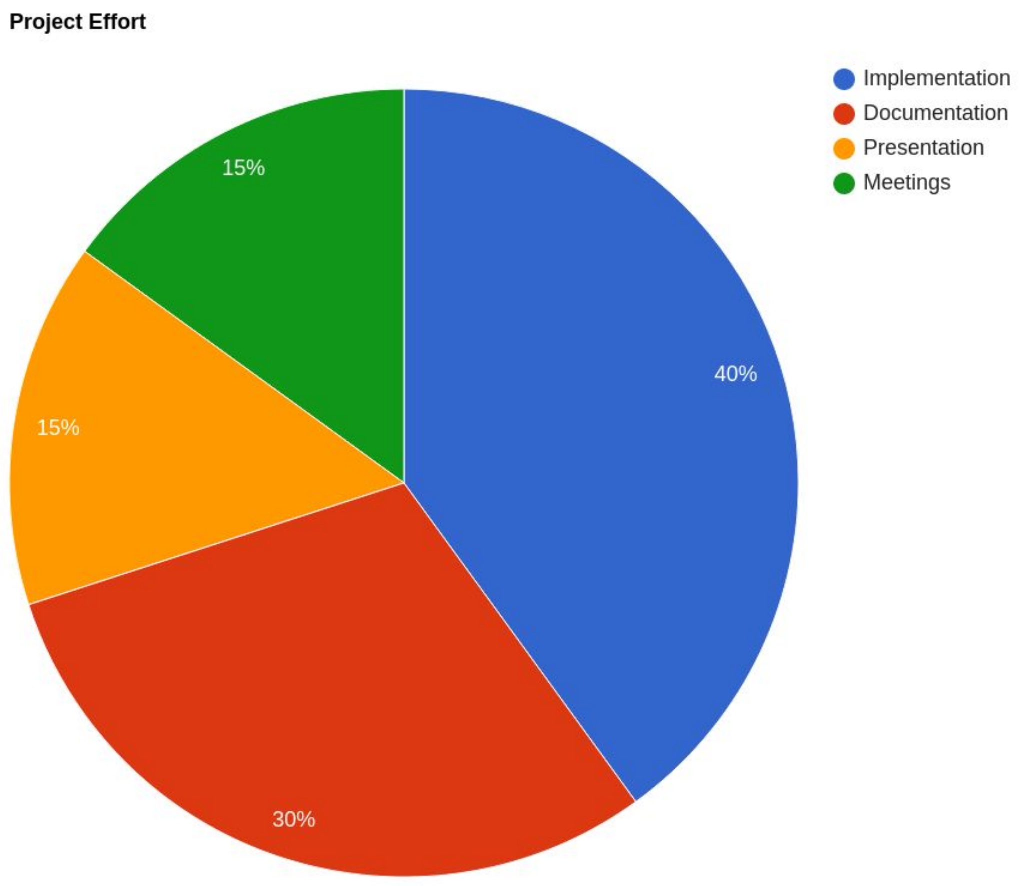

Figure 2. Pie chart showing total project effort for the whole group.

Distribution of work and responsibilities

Gunnar has been largely responsible for the construction of the front-end, and has created the dashboard, graphs, menu-bars, and did some work on integrating the front-end with the back-end. Gunnar is also largely responsible for the theming and visual design.

Emil was the project manager and the groups contact towards the client. It was through him that questions, meetings and suggestions were presented to the client. Regarding the implementation Emil contributed mainly to the front-end and the hide functionality of the charts. He was also present during the last two presentations.

Ville's contribution was mainly the backend. MetricCollectors (getMetricData), Login.php and dbconn. Part of the update-button on each chart. On the frontend side, involved in linking frontend to backend. At the end tried to create a bona fide installer that would automate the entire installation of the project, however it couldn't be finished in time for the project submission.

Sandra’s contribution was to the back-end. Admin page, create and delete users when we were still working with that. And also cookies for storing user information. She also presented in the two previous presentations.

Gezim’s contribution was mainly to the database, building the internal database, also the procedures required from the back end team and also he got involved in the admin page. He also presented during the first and the final presentation and also for the project plan and the final design report.

Roland’s contribution was mainly to the creation, management of the internal database , creating the stored procedures to provide the required data to the back end team. In addition to this he was involved in the admin page testing it’s functionality. He presented the first presentation and one presentation during the design phase of the project. His responsibilities started with a project plan and other documentations.

Johan’s contribution was mainly being a backend developer. He created the toggle metrics functionality both on the frontend and backend side of things. He also created a major part of the login page. He also presented during the first presentation and contributed to the initial project plan as well as the final design report.

Everyone was involved in the documentation/presentations, principally Emil Larsson was the point man for the main part of the documentation.

Working hours

Gunnar: 150-170 hours.
Emil: 140-160 hours.
Ville: ~150 hours
Sandra: 140-150 hours
Gezim: 150 hours
Johan: 157 hours
Roland: 150 hours

Experience and advice for new students

You should start from day one, have a structured plan throughout the whole project. If we would restart the project today with all the experience we have gained during this process we would first of all create well-defined tasks from day one.
Because in the beginning of the project we were not using a Trello board which resulted in that we moved forward very slowly.

Make sure to report work hours every week. This is something we didn't do properly during the middle of the project and so it was harder to get an exact value of the amount of work hours each project member contributed.

Our meetings were quite chaotic and unstructured in the beginning of the project. This was solved by establishing meeting protocols and by appointing a chairman and a secretary for each meeting. This made our meetings more time efficient and informative. Members were much more interested and focused. So this is one of the more important pieces of advice we can give.

A great challenge during the project was the holidays. We established beforehand that we were going to have meetings and still work on the project but this was not followed through. If we did the project again we would try to be much more efficient during the holidays or make sure to work more beforehand.

Appendix A

ID	1
Name	Log in as admin.
Pre-conditions	Have an admin account, on the mainpage.
Steps	1. Click on the cog icon on the left hand side menu. 2. Enter username and password. 3. Click on the “login” button.
Expectation	Settings page is displayed.

ID	2
Name	Stop collection of metric.
Pre-conditions	Logged in as admin, on settings page.
Steps	1. Click on any box in the table. 2. Make sure the box is unchecked. 3. Click on the speedometer icon on the left hand side menu.
Expectation	The chosen metric is not displayed on the dashboard.

ID	3
Name	Change timespan of metric collection.
Pre-conditions	On main page, haven't changed timespan previously.
Steps	1. Click the “time span” button at the top of the page. 2. Click on the “last hour” button in the dropdown menu.
Expectation	All graphs on the dashboard should update with metric data over a longer period of time.

ID	4
Name	Hide graphs.
Pre-conditions	On main page, have visible graphs.
Steps	1. Click on the hamburger menu in the top left corner of a graph. 2. Click on the “Hide” button.
Expectation	The chosen graph should disappear.

ID	5
Name	View hidden graphs.
Pre-conditions	On main page, there are hidden graphs.
Steps	1. Click on the “View hidden graphs” button at the top of the page.
Expectation	The previously hidden graphs should now be visible on the mainpage.

ID	6
Name	Change graph to be displayed as a pie or bar chart.
Pre-conditions	On main page, there is a graph displayed as a line chart.
Steps	1. Click on the hamburger menu in the top left corner of a graph. 2. Click on the “Pie” or “Bar” button.
Expectation	The graph changes to the selected chart type.