CS 443 Cloud Computing & (Mobile) Applications

Final Report

Spring 2020

Team Number: 5

Burak Kırımlı - 21400690
Ali Can Zeybek - 21401162
Mert Saraç - 21401480

1.Evaluation of the Process

1.1. Tech. Stack Decisions Updates

1.1.1 Infra/Platform Decision: Google Cloud Kubernetes Engine [updated]

Google Cloud Kubernetes Engine

- First reason for choosing Google Cloud is our team is familiar with it and it's features. Deployment of microservices is easy in Google Cloud. The most important reason for choosing Google Cloud is it's scalability. Also Google Cloud has support for code written in any language. [10]

1.1.2 Front-End Design Decision: Flutter, Dart, HTML, CSS, JavaScript

Flutter, Dart (programming language that is used in Flutter)

- First reason for choosing Flutter framework and Dart language for implementing the project's front-end of mobile application is Flutter framework is around 90% compatible with Android and IOS. With the help of this compatibility, it isn't necessary to write different codes for Android and IOS. [1]
- Second reason for choosing Flutter framework is Flutter has hot reload feature. With the help of that feature, developer can view the changes that he/she made in the code on emulators. [1]

HTML, CSS, JavaScript

- The reason for choosing HTML, CSS and JavaScript for implementing the project's front-end of webpage is HTML is a simple language that everyone can easily learn and build a basic webpage on browsers. With the help of CSS, the webpage that is built with using HTML can become more decorative and user friendly. Lastly, with the help of JavaScript, developers can make this webpage dynamic such as adding events to buttons. [2]

1.1.3 Development and Cloud Decisions: Spring Boot, Spring Cloud

Spring Boot

- The reason for choosing Spring Boot in development stage of the project is Spring Boot reduces huge amount of time for development stage and it increases productivity. Spring Boot also provides a lot of plugins to develop and test Spring Boot Applications and again it has a lot of plugins to work with databases. It is easy to develop with Java. [3]

Spring Cloud

- The reason for choosing Spring Cloud in implementing the cloud stage of the project is Spring Cloud provides developers to log correlation and tracing features. Also Spring Cloud has "Spring Cloud Gateway" which is programmable router based on Project Reactor. The most important feature of Spring Cloud is it provides client side load balancer that makes load balancing easy. Spring Cloud provides an implementation for circuit breaker

pattern. With the help of this circuit breaker pattern, microservice can continue operating when a related service fails. [4]

1.1.4 Database Decision: MariaDB [updated]

MariaDB

- The reason for choosing MariaDB in implementing the database is MariaDB has cloud-native support for data storage. Any project that uses MariaDB can lower data storage costs by using object storage services. Another advantage of MariaDB is it is designed to work with huge amount of columns and rows and produce results in seconds. This advantage can be used in testing phase. [5]

1.1.5 Project Management Tools Decisions: Skype, Google Drive

Skype

- The reason for choosing Skype for communicating during the implementation of the project is ease of use. Skype is the most common tool for online video chat and voice calls.

Google Drive

- The reason for choosing Google Drive for sharing the project's milestones (front-end codes, reports etc.) is ease of use.

1.1.6 Testing Tools Decision: Locust [updated]

Locust

- The reason for choosing Locust for testing purpose in the project is Locust is highly scalable due-to its event-based implementation. Another reason for choosing Locust is it allows to simulate thousands of concurrent users on a single machine easily. The last reason is Locust provides built-in functionalities to monitor performance scripts and analyze our test results. [9]

Postman[Updated]

- We also use Postman that enables us to test calls to APIs. We test whether GET and POST methods works correctly with corresponding data. We prefer Postman since every team member is familiar with it and it is easy to use.

1.2 Tasks of Each Team Member

1.2.1 Burak Kırımlı

Tasks:

- Implemented the home page of the website with login and signup features.
- Implemented the user page which users can shorten their link, users can see their active links and users can see their analytics.
- Implemented the admin page which admins can see system-wide analytics.

- HTTP requests (POST and GET requests) for building communication between services and pages that are listed above (home page, user page, admin page).
- Wrote the algorithm that can shorten the given link.
- Implemented the mobile application of the project which can deploy our website.

% of all workload: 35

1.2.2 Ali Can Zeybek

Tasks:

- Implemented microservices
- Implemented authentication logic
- Built Docker images for microservices
- Created Kubernetes config files for microservices
- Done networking for Kubernetes
- Deployed microservices on Google Kubernetes engine

% of all workload: 55

1.2.3 Mert Saraç

Tasks:

- Research about cloud deployment methods
- Help other team members in particular topics
- Testing on Postman
- Testing on locust.io

% of all workload: 10

1.3 Blocker Events

- Since most of our courses in the department are theoretical, we had hard times to learn using and making implementation with new tools such as Flutter, Docker etc.
- Because of Covid-19 pandemic we made our meetings with using Skype and Zoom. It is obvious that communicating with using online tools can cause some problems such as misunderstanding some points.
- Since everybody has different house environment, individually some of our team members had problems due to their house environment such as noise, lack of own room, bad internet connection etc.
- Some of our team members had concerns because of having doctor in their family.
- Learning completely new tools and techniques (Kubernetes, Docker, HTTP requests, Bootstrap, Locust etc.) and using them in a real project.

1.4 Solutions

- To prevent the bad effects of having a lack of practical experience, we worked very hard to learn these new tools and techniques which we are going to use in our project.

- To not to affect from house environment issues, generally, we worked in late hours.
- In order not to be affected by completely new learned techniques and tools for the project, we worked 8-9 hours per day on average.
- To prevent the possible misunderstandings due to online meetings, every day we made at least 3-4 meetings to understand every point in the project.

1.5 What we learned?

- Flutter framework and Dart language
- HTML, CSS, JavaScript and Bootstrap
- HTTP requests (POST and GET)
- MariaDB
- Docker
- Kubernetes
- Locust.io
- Implementing microservices
- Authentication logic
- How cloud system work
- Load balancing
- Working as a team

2. Final Outcomes

At the beginning of the project, we decided to use AWS cloud service platform. However, due to technical issues, we changed it with GKE(Google Kubernetes Engine). Also, our architecture has changed during the development process. The numbers that were specified at the design report also changed due to our test performance.

Final outcomes is decided by performance test, acceptability test and load test.

Load test is performed by locust.io. We performed this test with 100 number of users to simulate and hatch rate (users spawned/second) is 10. In this test, each users login the system and create 2 links. One is customize and one of them is randomized.

Acceptance test is performed by Postman to check whether we get OK message from server when GET or POST.

Performance test is also performed by locust.io. We plan to use HttpRider to generate code for running performance tests with locust.io which shorten the process of performance test.

We used AWS Total Cost of Ownership to enlighten us about the pricing for the system and cost for to keep it running for previous report. However, we didn't use AWS due to some technical difficulties so servers' cost should be changed. Since we use GKE(Google Kubernetes Engine), we used GKE Cost calculator. According to it, total estimated cost for GKE is 2921.17 USD per month and 7.90 USD for persistent disk. These are 3 years cost breakdown.

****Note:** Testing results are missing due to some personal problems that is discussed with our instructor. There was a serious fire with deaths happened in my building. Since I was responsible for all tests, results are missing seriously. It also affected my performance.

3. API Documentation

To write API Documentation, we used Postman API Documentation tool. The link of our API Documentation is :
https://documenter.getpostman.com/view/11191315/SzfAxR1h

4. Details of Codebase

Our codebase is located at: https://github.com/ali-z-can/cs443_project/

4.1 Client-side Details

In client-side, we implemented front-end of the project for both mobile and web. With using HTTP POST and GET requests we build a communication between services and front-end. Link shortener algorithm and JWT token decoder algorithm are also implemented in client-side.

index.html

"index.html" file is for our project's home page. Codes for "Home Page", "About Us", "Contact", "Login" and "Signup" are placed in this file. If user presses the related button, this screen is displayed. Display mechanism is handled with JavaScript code that is placed in "script" tag in index.html file. HTTP requests for login and signup page is handled in "script" tag, too. When user fills the "Signup" form and presses the submit button, a POST request is sent to registration service. When user fills the "Login" form and presses the submit button, a POST request is sent to authentication service, if he/she enters the correct username and password combination, system redirects user to "user.html" page, otherwise, system displays an error message.

user.html

"user.html" file is for our project's main page for users. Codes for "Shortener", "My Links" and "Analytics" are placed in this file. If user presses the related button, this screen is displayed. Display mechanism is handled with JavaScript code that is

placed in “script” tag in user.html file. Link shortener algorithm and token decoder algorithm are placed in “script” tag, too. In “script” tag there are also, GET and POST requests for building “Shortener”, “My Links” and “Analytics” tabs.

admin.html

“admin.html” file is for our project’s system-wide analytics page for admins. HTML, JavaScript and Bootstrap is used for implementing. Codes for “Analytics” tab are placed in this file. GET request for getting analytics are placed in “script” tag.

4.2 Back-end Details

Our codebase started as a Spring Cloud project utilizing Spring Cloud Netflix. After we managed to get a working project and decided to deploy to Google Cloud using Google Kubernetes Engines we discovered that we either don’t need some projects, can’t use some projects or can’t handle some projects. The complete list of deprecated services given below with explanation. Currently instead of using Spring Cloud Netflix we pivoted to Spring Cloud Kubernetes. Currently there are 12 Spring Boot projects working in a microservice fashion.

Deprecated Services:

innergateway: This project started as an inner gateway routing communication between microservices. We decided that this was unnecessary because of kubernetes’ cluster architecture.

commonservice: We decided to start a commons service to avoid code duplication but because of time constraints this project did not get any attention.

netflixeurekanamingserver: This project is no longer in use since kubernetes already provides service discovery.

springcloudconfigserver: This project started to enhance 12-factor app principles but later replaced with kubernetes secrets.

userlinkanalyticsretrievalservice: This project never used since link retrieval service was already doing a very similar job and after our discovery of reflections we decided to use linkretrieval service instead.

In-Use Services:

adminstatisticsservice: This service is responsible from getting application wide link analytics from linkretrievalservice, getting application wide user analytics from useranalyticsretrievalservice and providing system wide statistics for monitoring.

authenticationservice: This service is responsible for fetching user information from the database, validating them and assigning jwt tokens to responses of authentication requests.

linkanalyticsservice: This service is responsible for updating currently active links statistics such as number of clicks.

linkdeletionservice: This service is responsible for deleting out of dated links and calling userpastlinkanalyticsrearrangementservice to update the owner of the links statistics.

linkretrievalservice: This service is responsible for fetching all information about links from the database.

linkstoreservice: This service is responsible for storing new links into the database.

netflixzullapigatewayserve: This service is currently our out facing service. It redirects all requests and validates jwt tokens for authentication.

redirectionservice: This service is responsible for getting short url information from linkretrievalservice and redirecting the users to long urls.

registrationservice: This service is responsible for registering new users into out system.

useranaylticsretrievalservice: This service is responsible for fetching information for users' statistics as live link information from linkretrievalservice and as outdated link statistics from userpastlinkanalyticsretrievalservice.

userpastlinkanalyticsarrangementservice: This service is responsible for updating users past link analytics from outdated links.

userpastlinkanalyticsretrievalservice: This service is responsible for fetching outdated link analytics from the database.

5. Access Details

link: http://34.65.210.158:8080

5.1 Admin Account

username = admin
password = cs443

5.2 User Account

username = usertest
password = test443

References

[1] “What is Flutter? Benefits and limitations”. https://blog.codemagic.io/what-is-flutter-benefits-and-limitations/. [Accessed: March 24, 2020]

[2] “What are HTML, HTML5, CSS, and JavaScript? What are their advantages?”. https://www.quora.com/What-are-HTML-HTML5-CSS-and-JavaScript-What-are-their-advantages . [Accessed: March 24, 2020]

[3] “Spring Boot Tutorial”. https://www.journaldev.com/7969/spring-boot-tutorial . [Accessed: March 25, 2020]

[4] “Spring Cloud and its advantages”. https://www.javacodemonk.com/spring-cloud-and-its-advantages-3ac60b2c . [Accessed: March 25, 2020]

[5] “Key Benefits of MariaDB Platform”. https://www.datavail.com/blog/the-advantages-of-mariadb-platform-for-analytics/ . [Accessed: 5 April, 2020]

[6] “Getting Started With Cloud Testing,” https://www.softwaretestinghelp.com/getting-started-with-cloud-testing/. [Accessed: 02-Apr-2020].

[7] “How the Cost of Cloud Computing is Calculated,” *Expedient*, 01-May-2015. [Online]. Available: https://www.expedient.com/knowledgebase/blog/2015-05-01-how-the-cost-of-cloud-computing-is-calculated/. [Accessed: 02-Apr-2020].

[8] “8 Factors to Consider in Cloud Computing Capacity Planning” https://www.sungardas.com/globalassets/_multimedia/document-file/sungardas-8-factors-to-consider-in-cloud-computing-capacity-planning-white-paper.pdf [Accessed: 02-Apr-2020].

[9] “JMeter vs. Locust: What to Use When”. https://dzone.com/articles/jmeter-vs-locust-what-to-use-when. [Accessed: 15 April, 2020]

[10] “Google Kubernetes Engine or Cloud Run: which should you use?”. https://cloud.google.com/blog/products/containers-kubernetes/when-to-use-google-kubernetes-engine-vs-cloud-run-for-containers. [Accessed: 16 April, 2020]