****

****

毕业设计（论文）外文资料翻译

**学 院： 计算机科学与工程**

**专 业 班 级： 软件工程**

**指 导 教 师： 谷 钰**

**姓 名 学 号： 刘育超 20141450**

**外 文 出 处：**[https://www.sciencedirect.com/science/article/pii/](https://www.sciencedirect.com/science/article/pii/S1877050917329800)

[S1877050917329800](https://www.sciencedirect.com/science/article/pii/S1877050917329800)

**2020年3月**

指导教师签字：

年 月 日

评阅人签字：

年 月 日

**作者：**Suryotrisongko, Hatma, Dedy Puji Jayanto, and Aris Tjahyanto.

**英文原文：**

**Design and Development of Backend Application for Public**

**Complaint Systems Using Microservice Spring Boot**

Design and Development of Backend Application for Public Complaint Systems Using Microservice Spring Boot

Hatma Suryotrisongko\*, Dedy Puji Jayanto, Aris Tjahyanto

Institut Teknologi Sepuluh Nopember, Kampus ITS Sukolilo-Surabaya 60111, Indonesia

Abstract

E-government is an implementation of ICT (information and communication technologies) in the field of governance in improving services to the public by a government or public sector. For example, Smart City, online licensing services, community complaint services, etc. The purpose of this research is to develop public complaint service application based on web application which uses springboot microservice architecture. Microservice architecture was used to divide the application functionality into many parts, or many micro-services based on business process and the services are interconnected, becoming a single application with a complete business process. One of the advantages of this architecture is that more microservice can be added without affecting others. The application was deployed in a cloud environment that can be accessed through a browser.

Keywords: e-Government; Microservices; Complaint Services; Spring Boot

1.Introduction

E-government consists of the use of electronic communication technologies such as internet, for improving citizens access to public services [1]. Implementation of E-Government provides the efficiency and speed of management on the reporting administration system, as well as the transparency of processes that occur in government administration. Through it, an aspect emerged which is called good governance. However, the implementation of E-government in Indonesia faces many issues such as expense issues for developing and

operating e-government applications, technical issues such as security issues, privacy, and system update and human resource issues in which there is lack of capability to manage it. Cloud technology becomes one of the alternative answers to tackle those issues. This model allows consumers to use applications that exist in the cloud online through providers that can be accessed in various kinds of devices without worrying about those issues.

The use of microservice technology can give some advantages to the e-government system in the cloud. The concept of modularity in microservice allows the management of services that exist separately in an application. The impact to a development of a particular service is that it will not interfere with other services. Capacity building of a service can be distinguished among other services so that the resource is used appropriately. Also, the development of services can be developed with different programming languages [2].

2.Related works

Sam Newman [3] developing a microservice application, first to conduct is determining the boundaries of the environment or bounded context. Simply, to specify the bounded context of the application that can be seen from the application business process which can then be grouped according to functional groups from the user, e.g., the finance department is in charge of the payment, and the warehouse department is engaged in customer orders. Then each function will be put together into one module. The module then becomes the bounded context to create microservices that match the purpose of the module made. The microservices that have been made will apply the concept of loose coupling between other modules and high cohesion microservice that are interconnected with the module made. Finally, the part which will do data writing to the database and read data from the database can be determined.

Purnama & Yatini [4] developed a thesis management application using Node.js which aims to avoid any similarity of topic or thesis title which is often the case of plagiarism. Node.js was built using microservice architecture purposing the ease of development of the application. When there is a case of new functional addition, re-creating the application is unnecessary, and the function can be added independently. It takes less time for further development.

Janssen & Joha [5] explained that the use of the Software as a Service (SaaS) model in the public sector is still infrequent. Although SaaS in the public/e-government sector promises many advantages, such as cost-saving and effectiveness, the challenges are severe, e.g., quality, security, privacy, and also the need to customize different systems in the region One with other areas.

3.Methodology

3.1.Functional requirement analysis

Table 1. Functional requirements.

ID Actor Functional Requirements

FR1 Administrator Customer registration

FR2 Administrator Citizen ID management

FR3 Administrator Category management

FR4 Citizen ID verification

FR5 Citizen Sent a complaint

FR6 Citizen Check complaint

FR7 Government Work Unit Show complaint recap

FR8 Government Work Unit Answer a complaint

FR9 Government Work Unit Delete a complaint

FR10 Vendor View all customer

Functional requirement analysis was conducted by looking at some similar applications. For example, the public complaint web application in Kediri city [6]. The analysis was attained through looking at design document of the application. Also, the analysis was made from literature studies to find some information related to the functional requirement of e-government public reporting applications.

Functional requirement consists of actor requirement and functional requirement. The actor is a user who will use the application. Few actors who use the application include an administrator, vendor, citizen, and the government work unit. The functional requirement itself is the function of the application. A list of some functional requirements of the application is in Table 1.

3.2.Model the microservice

After a functional requirement was made, the next step is to model the microservice. This step splits the functional requirement which is also known as bounded context into some microservices that match the purpose of the bounded context. In other words, the microservice is a group consisting of one or more microservices, and they are interconnected to perform a business process or a function. The microservices in this application is in Table 2.

Table 2. Microservice requirements.

ID Microservice Requirements

FR1 Create a new customer data

FR2 Create citizen ID data

FR2 Show list of citizen ID data

FR2 Delete citizen ID data

FR3 Create a category

FR3 Delete a category

FR3 Show list of category

FR4 Get a citizen ID data

FR5 Create a new complaint

FR5 Get a ticket number

FR6 Show complaint status by ticket

FR7 Show all complaint with particular status FR8 Update answer in a complaint data

FR9 Delete a complaint

FR10 Show all registered customer

3.3.Designing use case

The designing use case was generated from the analysis of functional requirement. Use case shows the interaction between the actor with the application like the Fig. 1.

3.4.Application development

This phase is the process of coding the application based on functional requirement, microservices requirement, and use case. This research focuses on developing the back-end application using java programming language and springboot framework. Springboot framework was favored for it has several advantages.

Spring has supported MVC and provides a RESTful Web Service feature. A database connection has also been provided in the spring package. Spring framework also supports a dependency injection. Dependency injection is the ease of configuration dependency in the application so that in the application development process becomes more

convenient. Each spring framework also supports Aspect Object Programming (AOP) [7]. The additional advantage using spring boot is that a tomcat server could easily be included and can be run directly [8].

Developing the front-end application is also needed for communicating to the back-end for making business process. The front-end development is made using typescript programming language and Angular2 framework.

Fig. 1. Application use case.

3.5.Microservice blackbox testing

Blackbox testing is a test conducted to check whether the functional application is running correctly without knowing the processes occur within the app [9]. Blackbox testing was done by creating a test case in the form of test input and output expected from functional applications. Testing can be done on applications that do not use algorithms or low granularity levels [10], so it does not take much time [11]. Although for testing microservice two testings are needed, which are white box and black box testing, this research just performs the black box test because the application does not use any sophisticated algorithm and has low granularity details then the black box test is enough.

4.Application business process

The public complaint application is operated and managed by a specific unit or team. In some cases, in areas such as Bandung city [12], this kind of application is operated and managed in command center so that its work is optimal.

This public complaint application is a cloud-based application. Local governments can rent the application services by paying some money to the vendor. An administrator can be assigned by the government for operating the applications. The administrator can upload citizen ID data.

Citizens create complaint via the application in web. To begin with, citizens must verify their ID based on the uploaded data ID by the administrator. After that, their complaint will be saved on the database and ready to be managed by the government work unit. First, there is a selection stage by the system to filter the content of the complaint. The selection process is managed by the government work unit in the command center. Valid complaint content will be classified by category that has been created by the administrator and then sent to the right government unit outside the command center so that that complaint can be answered with the right answer and right follow-up in the field. If the complainant report is addressed or handled in the field, then the citizen who sent the complaint may change the complaint status to completion to inform the government that the complaint has been adequately addressed.

5.Application architecture

Fig. 2. Application architecture.

The public complaint application that uses cloud technology has a cloud architecture like Fig. 2. Front-end applications will be stored in the cloud as well as back-end applications along with application databases in the same cloud service to save resource usage. The application is accessible to the public on the web through a browser. Complaints submitted through the web application will be sent into the database for further management by the government which in this case is the command center and other government work unit. Vendors access the applications either back-end or front-end to perform maintenance on the application.

6.Microservice architecture

There is a tool that can make it easier for developers to see all the microservice available on the application. The tool is called swagger. Swagger is a standard framework that allows developers to quickly find and understand all the services on the application without accessing the program code, application development documentation, and without requiring inspection of the application service network [13]. Swagger maps the microservice based on the controller class that has been created using springboot framework. Class controller (or can be called a controller) is a class file in springboot which consists of methods to perform an input-output process according to business needs made. In the controller class, the methods can be made into a REST API. Swagger shows all the controllers that have been created in the application. In each swagger controller the REST API can be accessed with the description of the method used (the REST method used is usually GET, POST, DELETE, PUT), then there is the REST API

URL, and also the method name in the controller class. REST API is what will be used by the front-end application as access to a microservice with a predefined communication method that has been made.

6.1.Class controller

Table 3 shows class controllers that exist in the application with the associated microservices.

Table 3. Controller class.

Controller Class Microservice Requirements Customer Controller Create a new customer data Citizen Controller Create citizen ID data Citizen Controller Show list of citizen ID data Citizen Controller Delete citizen ID data Category Controller Create a category

Category Controller Delete a category Category Controller Show list of category Citizen Controller Get a citizen ID data Complaint Controller Create a new complaint Complaint Controller Get a ticket number

Complaint Controller Show complaint status by ticket Complaint Controller Show all complaint with particular status Complaint Controller Update answer in a complaint data Complaint Controller Delete a complaint

Customer Controller Show all registered customer

6.2.Rest method

Table 4 is a table containing REST method information that exists in the application with the associated microservices.

Table 4. REST method.

REST Method Microservice Requirements

POST Create a new customer data

POST Create citizen ID data

GET Show list of citizen ID data

DELETE Delete citizen ID data

POST Create a category

DELETE Delete a category

GET Show list of category

POST Get a citizen ID data

GET Create a new complaint

GET Get a ticket number

GET Show complaint status by ticket

GET Show all complaint with particular status

PUT Update answer in a complaint data

DELETE Delete a complaint

GET Show all registered customer

6.3.REST URL

Table 5 contains the URL of the REST. The URL is used to communicate between front end and backend.

Table 5. REST URL.

Rest URL Microservices Requirements

/customer/new Create a new customer data

/citizen/upload Create citizen ID data

/rest/citizen/allby/{idkokab} Show list of citizen ID data

/rest/citizen/delete/by/{idkokab} Delete citizen ID data

/rest/admin/category/new Create a category

/rest/admin/category/delete/{categoryid} Delete a category

/rest/admin/category/all/{idkokab} Show list of category

/citizen/checknik Get a citizen ID data

/complaint/new Create a new complaint

/complaint/new Get a ticket number

/complaint/find/ticket/{ticketcode} Show complaint status by ticket rest/complaint/kokabandstatus Show all complaint with particular status rest/complaint/update/answer/{id} Update answer in a complaint data complaint/delete/{id} Delete a complaint

/rest/user-dev/customer/all Show all registered customer

6.4.Architecture diagram

The front-end can be divided into four user interfaces (UI), namely admin dashboard (Admin UI) for administrator, Operator UI for government work unit, UI complaint that is used by citizen to create and check compliant, and Vendor UI that is used by the vendor. Each UI will be associated with some microservice as shown in Fig. 3.

Fig. 3. Microservices achitecture.

6.5.Securing microservice

Fig. 1 depicts that administrator, government work unit, and the vendor have login functionality. The login itself is to be used to secure microservices for those actors. Those actors have to fill their username and password, and then the system will compare them to saved data in the database. If username and password match, the system will

send a Jason web token (JWT). JSON Web Token (JWT) is a compact, URL-safe means of representing claims to be transferred between two parties which are the back-end and the front-end [14]. JWT contains some claims such as user scope that refers to the user actor and their dashboard, expired token time. So, when the time comes it will require a login process again and some additional information which is not of sensitive information that was required. The password itself need to be saved in database safely. Hashing function needs to be done so the password in the database is not a just plain text so it can be stolen easily. This research is using a BCrypt algorithm. BCrypt is an irreversible algorithm with low hash collision, using which our password is protected with proper protection [15]. The Java language provides bcrypt library, so this is a convenient way.

7.Conclusion

Based on the processes that have been done in this research, conclusions are as follows: 1) The public complaint is an application as a service developed with microservice architecture using springboot framework. Therefore, this application was deployed in the cloud. In deploying the application, a jar file was made from the back-end. The front-end was also deployed into the same cloud system as well as database applications used. To access the application, it takes a browser that will create communication to applications that are in the cloud. 2) In developing a microservice application, the functional needs should be broken down into several microservices that are interconnected to form a unified application by business processes that have been determined. 3) The advantages of using microservice architecture are that more functional needs can be added and also microservices within it without affecting other microservices. Thus, the developer can add more functionality independently, and that creates a business value which saves time to further development.

Acknowledgement

This research was supported/funded by “Penelitian Laboratorium PUPT ITS 2017” research grant, under contract

number: 566/PKS/ITS/2017.

References

[1]D. A. Putra and N. Aminuddin, "Tactical Steps E-Government Development For Regional Government in Pringsewu Regency; Langkah – Langkah Taktis Pengembangan E-Government Untuk Pemerintahan Daerah (Pemda) Kabupaten Pringsewu," Technology Acceptance Model, p. 67, 2014.

[2]S. Sharma, "Mastering Microservices with Java," in Mastering Microservices with Java, Birmingham, PACKT publishing, 2016.

[3]S. Newman, "How to Model Services," in Building Microservices: Designing Fine-Grained Systems, O’Reilly Media, 2014.

[4]H. Purnama and I. Yatini, "Thesis Managemen Application in STMIK AKAKOM Yogyakarta Using Microservice Architecture With Node.Js," in Seminar Riset Teknologi Informasi (SRITI), 2016.

[5]Marijn Janssen and Anton Joha, "CHALLENGES FOR ADOPTING CLOUD-BASED SOFTWARE AS A SERVICE (SAAS) IN THE PUBLIC SECTOR," in European Conference on Information Systems, 2011.

[6]Electronic Data Management Department of Kediri City Government, "Application Development Document of E-Complaint Kediri City".

[7]Mr.Anil Kumar, Dr. Arvind Kumar and Mr. M. Iyyappana, "Applying Separation of Concern for Developing Softwares Using Aspect Programming Concepts," International Conference on Computation, pp. 906-9014, 2016.

[8]I. A. Pradana, "Mengenal Spring Boot," 3 February 2017. [Online]. Available: https://www.codepolitan.com/spring-boot-pengenalan- 588da0c4bedd1. [Accessed 16 February 2017].

[9]M. Kumar, S. K. Singh and Dwivedi, "A Comparative Study of Black Box Testing and White Box Testing Techniques," International Journal of Advance Research in Computer Science and Management Studies, 2015.

[10]S. R. Jan, S. T. U. Shah, Z. U. Johar, Y. Shah and F. Khan, "An Innovative Approach to Investigate Various Software Testing Techniques and Strategies," IJSRSET, 2016.

[11]A. Orso and G. Rothermel, "Software testing: a research travelogue (2000–2014)," in Proceedings of the on Future of Software Engineering, New York, 2014.

[12]A. Nurmatari, "Aplikasi Seluruh SKPD akan Terintegrasi di Bandung Command Center," 7 December 2016. [Online]. Available: https://news.detik.com/berita-jawa-barat/d-3365344/aplikasi-seluruh-skpd-akan-terintegrasi-di-bandung-command-center. [Accessed 1 Maret 2017].

[13]Swagger, "Getting Started With Swagger," [Online]. Available: https://swagger.io/getting-started/. [Accessed 10 July 2017].

[14]M. B. Jones, "draft-ietf-oauth-json-web-token-32," [Online]. Available: https://tools.ietf.org/html/draft-ietf-oauth-json-web-token-32. [Accessed 26 Juli 2017].

[15]Yung-Feng Lu, Chin-Fu Kuo and Yi-Yen Fang, "Efficient Storage Encryption for Android Mobile Devices," in Proceedings of the International Conference on Research in Adaptive and Convergent System, Odense, 2016.Examples of distributed database systems can be found in the engineering world. Sun’s Network Filing System (NFS), for example, is used in computer-aided engineering applications to distribute data among the hard disks in a network of Sun workstation.

**英文翻译：**

使用microServices SpringBoot设计公众投诉系统

摘要：

电子政务是ICT（信息和通信技术）的治理领域中由政府或公共部门提高对公共服务的实现。例如，智能城市，在线许可服务，社区投诉服务等。本研究的目的是在此基础上采用springboot微服务架构的Web应用程序开发公共投诉服务应用程序。微服务架构是将应用程序的功能划分成许多部分，或基于业务流程的许多微服务和服务相互连接，成为一个完整的业务流程的单个应用程序。其中一个这种结构的优点是，更多的微服务可以在不影响其他人加入。

关键词： 电子政务; 微服务; 投诉服务;

介绍

电子政务由使用电子通信技术，如互联网的，对于提高公民获得公共服务的[1]。电子政务的实施提供了效率和报告管理系统管理的速度，以及对发生在政府行政过程中的透明度。通过它，一个方面出现了被称为良好治理。但是，电子政务在印尼的实施面临着很多问题，如开发费用问题，

经营电子政务应用，技术问题，如安全问题，隐私和系统更新，并在其中有没有能力来管理它的人力资源问题。云技术成为备选答案来解决这些问题中的一个。这种模式使消费者，网上通过，可以在各种设备中，而不必担心这些问题访问供应商存在于云中使用的应用程序。

使用微服务技术可以给予一定的优势，在云计算的电子政务系统。在微服务模块化的概念允许在应用程序中单独存在服务的管理。特定服务的发展的影响是，它不会干预其他服务。服务的能力建设可以在其他服务中区分开来，使资源使用得当。此外，服务的发展可以用不同的编程语言[2]开发。

山姆·纽曼[3]开发的microService应用，第一到行为是确定环境或界上下文的边界。简单地说，指定可以从应用程序的业务流程，然后可以根据来自用户的官能团进行分组可以看出应用程序的有限范围内，例如，财务部门负责支付和仓库部门从事客户订单。然后，每个功能将被拼凑成一个模块。该模块便成为界上下文创建匹配该模块的目的是微服务。已经进行了该微服务将应用互连与该模块的其它模块和高凝聚力的microService之间的松耦合的概念。最后，

普尔纳马＆Yatini [4]开发了使用Node.js的其目的是避免主题或论题标题任何相似性这往往是剽窃的情况下的论文管理应用程序。Node.js的用微服务架构考虑对放心的应用程序的开发建造。当有新的官能加成的情况下，重新创建应用程序是不必要的，该函数可以独立地加入。它需要较少的时间进一步发展。

扬森和JOHA [5]解释说，使用本软件为在公共部门服务（SaaS）模式仍是罕见的。虽然在公共/电子政务领域的SaaS承诺的许多优点，如节约成本和效益，所面临的挑战是严峻的，如质量，安全性，私密性，同时也需要一个与其他地区定制不同的系统，在该地区。

方法

功能需求分析

表1.功能要求。

|  |  |  |
| --- | --- | --- |
| ID | 演员 | 功能要求 |
| FR1 | 管理员 | 客户登记 |
| FR2 | 管理员 | 公民身份管理 |
| FR3 | 管理员 | 分类管理 |
| FR4 | 公民 | ID验证 |
| FR5 | 公民 | 发出了投诉 |
| FR6 | 公民 | 检查投诉 |
| FR7 | 政府工作单位 | 展会投诉回顾 |
| FR8 | 政府工作单位 | 接听投诉 |
| FR9 | 政府工作单位 | 删除投诉 |
| FR10 | 供应商 | 查看所有客户 |

功能需求分析，通过看一些类似的应用程序进行。例如，在谏义里市[6]市民投诉的Web应用程序。通过查看应用程序的设计文档达到的分析。此外，分析从文学研究以发现相关的电子政务公共报表应用功能要求的一些信息。

功能要求包括演员的要求和功能要求。演员是谁使用该应用程序的用户。谁使用该应用程序几个演员包括管理员，供应商，市民，政府的工作单位。功能需求本身是应用程序的功能。的应用程序的某些功能的要求列表在表1中。

模型中的微服务

一个功能性需求作出之后，下一个步骤是将微服务建模。此步骤拆分其也已知为界上下文到匹配有界内容的目的，一些微服务功能需求。换句话说，该微服务是由一个或多个微服务的一组，并且它们相互连接以执行业务流程或函数。在本申请中的微服务是在表2中。

表2.微服务的要求。

ID 微服务需求

FR1 创建一个新的客户数据

FR2 建立公民ID数据

FR2 显示市民ID数据的列表

FR2 删除公民ID数据

FR3 创建类别

FR3 删除类别

FR3 显示分类列表

FR4 获取公民ID数据

FR5 创建一个新的投诉

FR5 拿到票号

FR6 显示凭票投诉状态

FR7 显示与特定状态FR8所有投诉 在投诉数据更新答案

FR9 删除投诉

FR10 显示所有注册用户

设计用例

设计用例从功能需求的分析生成的。用例1示出了使用类似图1中的应用程序中的演员之间的交互。

应用程序开发

这个阶段是基于功能性需求，微服务要求，并用例编码的应用程序的过程。这项研究的重点是开发使用Java编程语言和springboot框架的后端应用程序。Springboot框架赞成它有几个优点。

春天已经支持MVC，并提供一个RESTful Web服务功能。数据库连接也已在弹簧组件提供。Spring框架还支持依赖注入。依赖注入是在应用程序的容易性配置依赖的，从而在应用程序开发过程变得更

方便。每个弹簧框架还支持属性对象编程（AOP）[7]。使用弹簧引导的附加优点是，Tomcat服务器可以很容易地被包括，并且可以直接[8]中运行。

开发前端应用程序，也需要向后端制造业务流程进行通信。前端开发利用打字稿编程语言和Angular2框架制成。

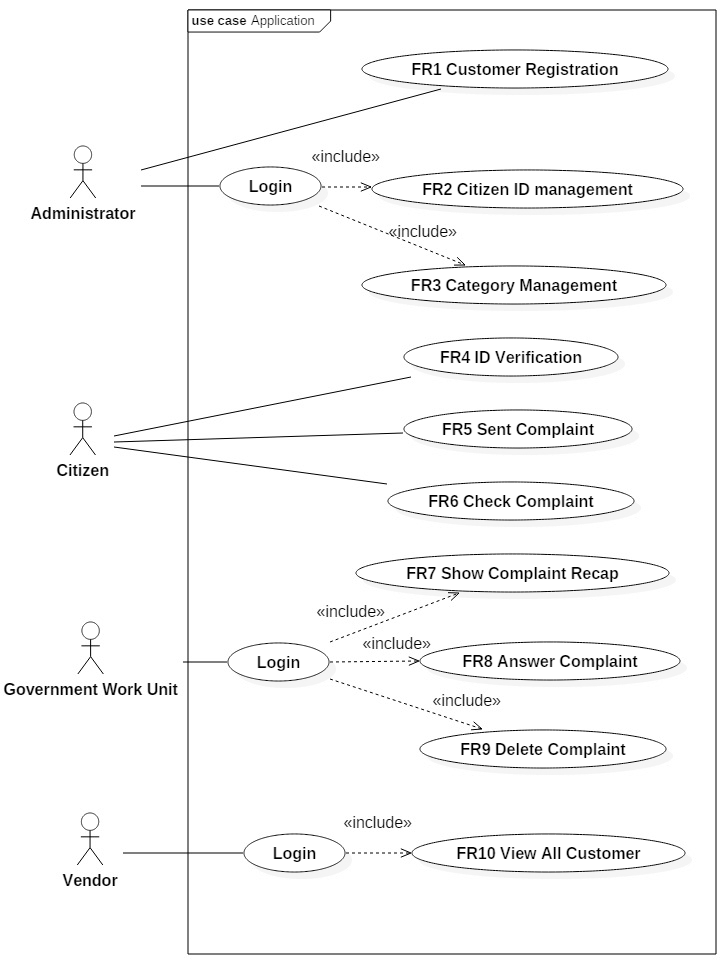


图1的应用使用情况。

微服务黑盒测试

黑箱测试是进行以检查是否该功能应用是在不知道应用程序[9]内发生的过程正确地运行测试。黑箱测试是通过创建在测试输入和输出的从功能应用预期形式的测试用例来完成。测试可以在不使用算法或低粒度水平[10]的应用程序来完成，因此它并不需要太多的时间[11]。虽然测试的microService需要两个testings，这是白盒和黑盒测试，这项研究只进行黑盒测试，因为应用程序不使用任何复杂的算法和具有低粒度的详细信息，然后黑盒测试就足够了。

应用业务流程

公众投诉申请的，由特定单位或团队管理。在某些情况下，如万隆[12]的区域，这类应用的操作，并在指挥中心管理，使得它的工作是最佳的。

本次公开抱怨应用程序是一个基于云的应用。地方政府可以通过支付一些钱给供应商租用的应用服务。管理员可以由政府操作的应用来分配。管理员可以上传公民ID数据。

公民通过网络中的应用程序创建的投诉。首先，公民必须基于管理员上传的数据ID验证其ID。在那之后，他们的投诉将被保存在数据库中，并准备通过政府工作单元进行管理。首先，存在由系统中的选择阶段过滤投诉的内容。选择过程是通过在指挥中心的政府工作单元进行管理。有效投诉内容将通过已经由管理员创建，然后指挥中心外发送到正确的政府单位，使得投诉可以用正确的答案和正确的跟进在现场回答类别进行分类。如果投诉报告在现场解决或处理，

应用程序架构

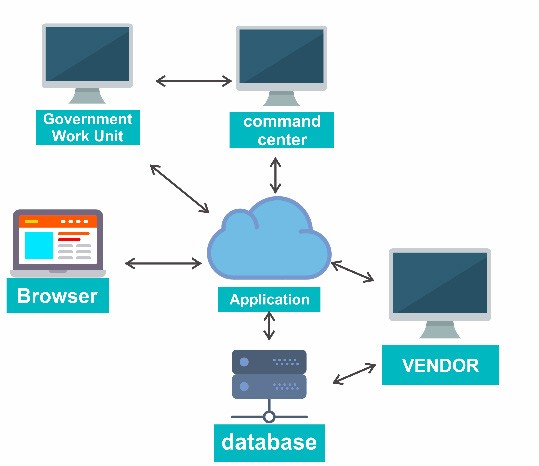


图2的应用架构。

市民投诉的应用程序，使用云技术有一个云架构类似于图2，前端应用程序将被存储在云中，以及在同一个云服务应用程序数据库，以节省资源的使用以及后端应用程序。该应用程序是通过浏览器在网络上对公众开放。通过Web应用程序提交投诉书将由政府在这种情况下是指挥中心和其他政府工作单元发送到数据库进行进一步的管理。供应商访问的应用程序任一后端或前端上的应用程序执行维护。

微服务架构

有一个工具，可以让开发者更容易地看到应用所有可用的微服务。该工具被称为招摇。扬鞭是一个标准的框架，使开发人员能够快速找到并理解所有的应用程序的服务，而无需访问程序代码，应用程序开发文档，并且无需应用服务网络[13]的检查。扬鞭映射基于已使用springboot框架创建控制器类的微服务。类控制器（或可被称为控制器）是在springboot它由方法根据由业务需求进行输入输出处理的类文件。在控制器类中，所述方法可以制成一个REST API。扬鞭显示已在应用程序中创建的所有控制器。

URL，并在控制器类中的方法名。REST API是什么将由前端应用程序被用作访问一个微服务与已经作出了预定的通信方法。

类控制器

表3示出了类的控制器存在于与相关联的微服务应用程序。

表3. Controller类。

控制器类 微服务要求客户​​控制器 创建一个新的客户数据公民控制器 建立公民ID数据公民控制器 显示市民ID数据公民控制器列表 删除公民ID数据分类控制器 创建类别

类别控制器 删除类别类别控制器 显示类公民控制器列表 获取公民ID数据投诉控制器 创建一个新的投诉投诉控制器 拿到票号

投诉控制器 显示凭票投诉控制器投诉情况 显示与特定的身份投诉控制器的所有投诉 在投诉数据投诉控制器更新答案 删除投诉

客户控制器 显示所有注册用户

休息方法

表4是存在于与相关联的微服务应用程序的含表REST方法的信息。

表4. REST方法。

REST方法 微服务需求

POST 创建一个新的客户数据

POST 建立公民ID数据

得到 显示市民ID数据的列表

删除 删除公民ID数据

POST 创建类别

删除 删除类别

得到 显示分类列表

POST 获取公民ID数据

得到 创建一个新的投诉

得到 拿到票号

得到 显示凭票投诉状态

得到 显示与特定状态的所有投诉

放 在投诉数据更新答案

删除 删除投诉

得到 显示所有注册用户

REST URL

表5包含了REST的URL。该URL用于前端和后端之间的通信。

表5. REST URL。

休息网址 微服务需求

/客户/新 创建一个新的客户数据

/公民/上传 建立公民ID数据

/ REST /公民/ allby / {} idkokab 显示市民ID数据的列表

/ REST /公民/删除/用/ {} idkokab 删除公民ID数据

/ REST /管理/分类/新 创建类别

/ REST /管理/分类/删除/ {}类别ID 删除类别

/ REST /管理/分类/所有/ {} idkokab 显示分类列表

/公民/ checknik 获取公民ID数据

/投诉/新 创建一个新的投诉

/投诉/新 拿到票号

/投诉/查找/票/ {} ticketcode 凭票休息/投诉/ kokabandstatus展会投诉情况 显示与特定状态休息/投诉/更新/答案/ {ID}所有投诉 在投诉数据更新答案 投诉/删除/ {ID} 删除投诉

/休息/用户开发/客户/所有 显示所有注册用户

架构图

前端可分为四个用户界面（UI），管理员分别管理仪表板（管理UI），操作员界面为政府工作单位，所使用的公民建立和检查标准的UI投诉，和供应商的用户界面，是由供应商使用。每个UI将一些微服务如图相关联。3。

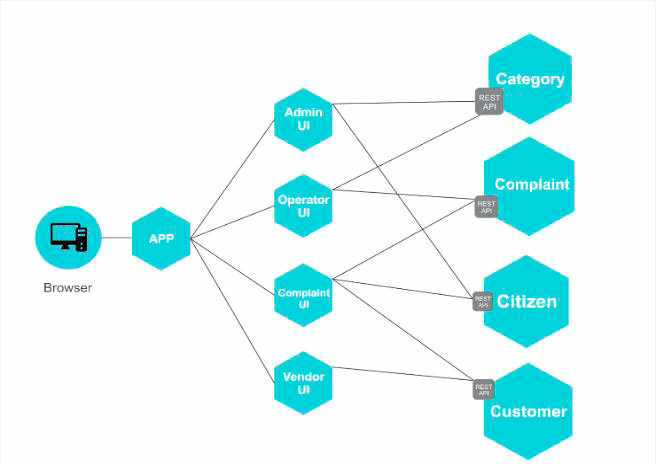


图3微服务achitecture。

保护的microService

图1描述了管理员，政府的工作单位，并且供应商必须登录功能。登录本身是用来固定微服务为那些演员。这些演员必须填写用户名和密码，然后系统会比较他们在数据库中保存的数据。如果用户名和密码匹配，系统会

发送杰森网络令牌（JWT）。JSON网络令牌（JWT）是表示权利要求双方其是在后端和前端[14]之间传送的一个紧凑的，安全网址装置。JWT包含了一些要求，诸如指的是用户的演员和他们的仪表盘，过期的令牌时的用户范围。所以，在时机成熟时，将需要再次登录过程，这是这是必需的敏感信息不一些额外的信息。密码本身需要被安全地保存在数据库中。哈希函数必须这样做在数据库中的密码是不是只是简单的文本，因此很容易被偷走。这项研究是利用BCrypt算法。BCrypt是一个不可逆的算法与低哈希冲突，使用它我们的密码与适当的保护[15]的保护。Java语言提供bcrypt库，

结论

在此基础上有与本研究已经完成的过程，结论如下：1）公众投诉的应用与使用springboot框架微服务架构开发的服务。因此，这个应用程序部署在云中。在部署应用程序，一个jar文件从后端制造。前端也被部署到使用相同的云系统以及数据库应用程序。访问应用程序，它需要一个浏览器，它会创建通信是在云应用。2）在开发微服务的应用程序，功能需求应当分解为相互连接形成的业务流程，已经确定了统一的应用程序的多个微服务。3）用微服务架构的优点是更多功能需求可以被添加，并且还微服务在其内，而不会影响其他微服务。因此，开发人员可以独立添加更多的功能，以及创建节省时间进一步发展的商业价值。