Oregon State University

CS 461

Fall 2017

Tech Review

Author:
Shengpei Yuan

Instructor:
D. Kevin McGrath
Kirsten Winters

Abstract

1 Container

1.1 Overview

Container is one powerful virtualization technology that is helpful in the entire process of software development with the cloud environment. Generally speaking, they greatly help the environment constructions and deployment works for developers and maintainers to improve efficiency and reduce cost of developing complicated software systems.

1.2 Criteria

Container is one important technology in cloud computing. It provides independent running environment for various software applications. It is believed that container may replace virtual machines in the near future as the dominant technology for constructing cloud-computing environments. Basically, container works on the OS layer and provides runtime environments for programs in it. OS could distribute computing, memory and peripheral resources for containers, and it owns isolated namespace for programs in it.

1.3 Potential Choices

1.3.1 LXC

LXC is one of the earliest containers to provide virtual Linux runtime environment for software applications. The core characteristics of LXC are built on basis of the resource management and isolations techniques of Linux kernels. For instance, the cgroups feature of Linux kernel that divides processes into groups for management is the essential framework for managing resources for processes in LXC.

1.3.2 Docker

Docker is a mature container tool has almost replaced the conventional Linux Containers (LXC) and helped solving the availability problems of many software systems. It is claimed that Docker is prevailing in todays software building technology, especially for virtualization. However, with the rapid development of Docker ecosystem as a tool for cloud-computing sometimes, perhaps unavoidably, it makes constructing other businesses based on it harder. Docker is one of the best components for constructing a software system in virtual computing environments. One perspective is that Docker is or will become an ecosystem for software construction, making it vulnerable in many ways, whereas Rocket makes a fresh start and only focus on working as a component to help construct complex software system. Another perspective is that the core value of Rocket is exactly the start point of Docker, integrating complicated software systems into one independent platform. The founder of Docker, Hykes agrees that Rocket could be one important tool for customizing configurations of containers, and would define the future of containers technology.

1.3.3 Rocket

Rocket is a quite new open-source container technology for software developing created by CoreOS in 2014. It is implemented by Go. As a command-line tool, Rocket works quite like Docker to package software applications and their dependency libraries or systems to a plantable container. It is known that Rocket would have no compatibility issues with Docker. Also, rocket will bring new ideas for software containers technology and relevant markets. Rocket never tries to provide broad friendly functionalities like cloud acceleration tools, and integrated systems. It would become more purified as a standard container tool for the software industry. It has a start point that multiple heterogeneous container platforms could be integrated to improve availability. And moreover, it would try to alleviate the security issues of Docker. It should be noted that Docker and Rocket both have their own strength and weakness, and no one could be suitable under all contexts and solving all container issues of software systems. Generally speacking, Docker would be more competent at complicated and huge systems, while Rocket seems to be good at lightweight and small software applications.

1.4 Conclusion

The vast adoption of Rocket among great software systems proves that it satisfies the deep requirements of software world. Rocket is designed to boost the success of software systems, including specific measures like guaranteed security, flexible components and open standards. Different companies may choose different container tools between Docker and Rocket according to the specific contexts of their systems. As a result, for this project, we adopt Rocket as an independent container library, so that it could be easily integrated into the existing project AgBizClimate and the corresponding environment. In other words, we need a container tool to construct components for our system rather than to finish the entire software system.

2 Back end Design

2.1 Overview

There are several powerful and popular models for building back end of a web application. RESTAPI and MVC are two important technologies among them. Besides, we would also describe a little about MMVC, an extension version of MVC. Basically, we could introduce them in the order of the time they were proposed in order to help readers to understand them easily.

2.2 Criteria

The backend architecture technologies mainly solve the problem of decomposing the entire business functionalities of applications reasonably and efficiently for software developers. In other words, the ideal condition is that all modules have clear bounds to others and tight connections within themselves. Besides, it is important to present the structures and functionalities of whole applications clearly and easily to both developers and end-users.

2.3 Potential Choices

2.3.1 MVC

The MVC (Model, View and Controller) model was proposed as early as 1970s and used by Smalltalk, and it is still popular and used to develop large software applications in various fields today. Many software language and frameworks like the prevailing Java Struts and Spring MVC. The core concepts behind the MVC model is that no matter how simple or complex a software system is, it could always be stratified into three layers from the perspective of structure. First, the View layer, which lies on the top, is the one directly saw by the end users. It usually works as an interface between users and the program, and is also called the shell of the program. Secondly, the Model layer, which lies in the bottom, usually represents and stores the data or information of the program. For many programs, this layer is the core layer that largely dominates the structure of the other two layers. Thirdly, the Controller layer, which lies in the middle, is mainly responsible for the interactions between the Model and View layers. More specifically, it accepts user instructions from View layer, retrieves data and deals them respectively, and sends back the result data to view for it to present the final results to end users. The above three layers are tightly connected together as well as being independent. The internal operations within each layer never affect other two layers. And each layer provides appropriate interfaces for other layers to call on. The essential point of this design is that the entire system could be divided into independent modules so that neither changing appearances nor changing data would alter other two layers. And hence, it greatly reduces the cost of maintaining and upgrading.

2.3.2 RESTAPI

RESTAPI (Representational State Transfer) was proposed by Roy T. Fileding in 2000 as a simple and extendible standard for developing web applications. In fact the HTTP protocol is one typical application of the RESTAPI architecture. With the vast applications of cloud computing technology, RESTAPI is very popular used by many software architectures and developers to build large web applications. It is believed that RESTAPI has becoming one important way of implementing SOA. The most essential features of RESTAPI are resources, unified interface, URI and stateless.

2.3.3 MMVC

The MMVC is an extension of MVC that optimized the Model or data layer. We need to know three essential points about MMVC. Firstly, it is compatible with the existing MVC architecture. Secondly, it makes the software applications more testable. Thirdly, it works best with a binding mechanism. It should be mentioned that MMVC is a relatively new back-end architecture technology and is currently not used broadly in the software industry.

2.4 Conclusion

It can be seen that MVC and RESTAPI try to build a powerful general architecture for building various software systems. MVC tries to model the system based on actions, whereas RESTAPI mainly focus on data, the sate and transitions of different states. MVC emphasizes on dividing the internal implementations into three layers Model, View and Controller, and RESAPI decompose the web applications in terms of outside appearance, transitions of states of system. Therefore, it is possible to develop programs based on MVC as well as following the RESTAPI standards. It should be clearly stated that RESTAPI and MVC are not totally opposite and conflict with each other. Instead, they could be and are usually adopted together to build the software structure. And in fact, many web frameworks supporting MVC have the ability of RESTAPI. Consequently, this project would try to adopt MVC to develop web applications following RESTAPI standards.

3 Styling framework

3.1 Overview

The styling frameworks are higher programming languages based on CSS (Cascading Style Sheets) to make CSS (Cascading Style Sheets) development flexible and efficient. There are many popular styling frameworks for building beautiful and consistent appearances of web applications. We would talk about three popular ones LESS, SASS and Bootstrap.

3.2 Criteria

The core function of styling framework is to allow easier and efficient web UI design using CSS. A good styling framework may have following three features. First, it provides programming ability like logic examination, loops and functions for basic CSS codes. Second, it provides rich and practical predefined templates, components, and suite themes for quick design of web UI. Thirdly, it integrates other front-end languages like HTML and Javascript for flexible and easy developing.

3.3 Potential Choices

3.3.1 LESS

LESS extends CSS and introduces module conceptions into it. It provides much functionality for front-end developers beyond the basic CSS. It has similar grammar rules as basic CSS. It is affected by the SASS and also affected the grammar rules of SASS. Like LESS, SASS is also an extension of CSS3 that introduces rules, variables, selector and inheritance. It tries to generate well-formatted CSS codes that are easy to organize and maintain.

3.3.2 SASS

For front-end programmers of web applications, SASS provides more powerful functionality than LESS, which works more like a real programming language than LESS. However, for UI designers, LESS seems to be clearer.

3.3.3 Bootstrap

Unlike LESS and SASS, Bootstrap is one comprehensive and powerful front-end framework based on HTML, CSS and JavaScript. Generally speaking, Bootstrap tries to integrate tools like Compass, Blueprint and h5bp and become a comprehensive framework for web front-end development. It should be mentioned that Bootstrap use Normalize.css to reset CSS, which has becoming the practical standard (used more broadly than Eric meyer 2.0 of Compass). Also, it is compatible with h5bp, meaning one can use h5bp and Bootstrap concurrently in one project. Generally, Bootstrap has three key features. Firstly, it includes the complete basic modules of CSS, although not as powerful as Compass. This make it possible for programmers to use basic CSS attributes for simple customization of HTML elements. Secondly, it provides some suites of predefined CSS, including one grid layout system like blueprint but with a different style. This would help programmers to quickly define the overall look of their web applications. Thirdly, it provides a group of UI components based on jquery like dialogs, navigation menus, and edits boxes. And they are all both powerful good look. This may be the most powerful strength point of Bootstrap. In fact, it is becoming a practical standard of most jquery-based web projects.

3.4 Conclusion

For this project, since we are not professional UI designers and work as programmers for the Agbiz-Climate applications. The best combinations of the above tools seem to be SASS and Bootstrap so that we could employ the powerful programming ability of SASS as well as the rich UI components of Bootstrap. It should be mentioned that the basic Bootstrap framework is based on LESS. We may need to use one extended version Bootstrap-sass from GitHub.