Trinity WayFinders

WayFinder Application

Technical Architecture Document

Revision History

Version	Created Date	Author	Comments
1.0	07-02-2019	Zihan Huang	

Authors List

Sl. No.	Name	Role
1.	Yifan Xu	
2.	Zihan Huang	
3.		
4.		
5.		
6.		

Table of Contents

1. Objective

This document gives a description of the technical architecture of the project, including the software and technologies to be used.

2. Architecture Overview

The technical architecture includes our source code management method, document management, plan, track and defect management and build strategy that we will use in backend and frontend.

3. Technical Architecture

3.1. Source Code Management

Git & Github https://github.com/trinitywayfinders/WayFinder

3.1.1. Version Control

We plan to use Git and Github as source code management in our project. We created an organization called "TrinityWayFinders" with all team members. A project repository called "WayFinder" was created to store and manage the source code of our project. Each user member should only have reading privilege to the Master branch, changes can only be made through creating a new branch then merge into the master branch.

3.1.2. Branching Strategy

Git provides a completed branching strategy of source code management. Regarding a member contributing to this project, a member should create a new branch and put the modification into the branch. The name of the branch should be related to the feature that the member is working on. As it can be seen in figure 1 below, branches are named to represent one task.

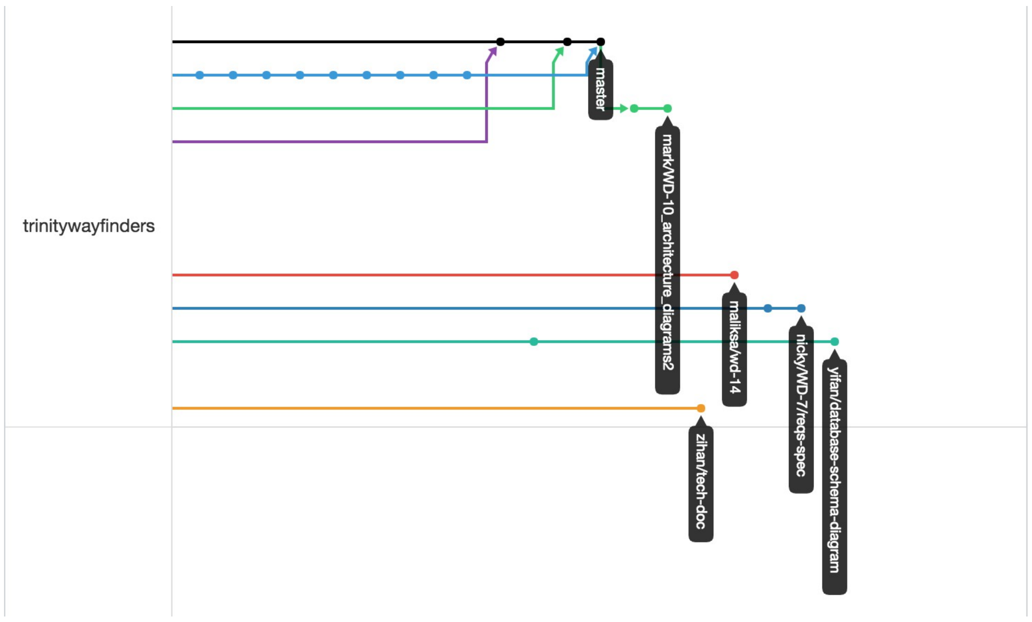

Figure 1. Network Graph

3.1.3. Merging Strategy

Github provides the pull request feature to allow a member to merge the modification with the existing source code in the master branch in a shared repository, even without writing privilege of the master branch. The pull request must be reviewed and approved by at least other two team members before the change can be merged into the master branch. As it can be seen in Figure 1 above, only after pull request been approved, it merged into the master branch, and the branch should be deleted after merging is done.

3.1.4. Conflict Resolution

Merge conflicts occur when competing changes are made to the same line of a file, or when one person edits a file and another person deletes the same file. When conflict happens, Git keeps two versions of change and requests the user to manually resolve the conflict. The code should be committed after the conflict resolved.

3.2. Document Management

We plan to Github Wiki to manage our document so that users can easily access from a public network. On the other hand, Github Wiki supports Markdown tags and user can format their document(Contributors, M, 2019).

Github Wiki: https://github.com/trinitywayfinders/WayFinder/wiki

3.3. Plan, Track and Defect Management

As an agile team, we will use JIRA to track the project and teammates. Jira Software is built for every member of your software team to plan, track, and release great software.

Project Track: JIRA http://34.247.159.151:8080/

3.4. Build Strategy

Because our application consisting of a few different components with various technologies, we have to choose different build management tools to build each single components.

3.4.1. NPM

For the front end, we use a cross-platform JavaScript framework called "Ionic", the NPM is default package manager provided by the framework, which can manage node.js package and dependencies automatically(Npmjs.com, 2019).

3.4.2. Gradle

For the backend, we will use spring boot. There are two options for project management, namely gradle and maven. Considering the popularity and accessibility of gradle, we have selected it to manage the Jar packages(Features, G., Plugins, C., Kotlin, G., Gradle, M., Resources, M., Training and Enterprise, G, 2019).

3.4.3. Jenkins

We will deploy Jenkins as our build engine on the cloud. It allows members to access and request to build at some points, eg. a new commit is pushed to Github repository. Jenkins also provides many useful plugins to support gradle and npm, which completely fits our requirement. (Jenkins, 2019)

3.5. Deployment Pipeline

3.5.1. Docker

Freedom of choice, agile operations, and integrated container security for legacy and cloud-native applications. Docker Compose is a tool for defining and running multi-container Docker applications. (Docker.com, 2019)

3.6. Communication Strategy

We use a WhatsApp group chat as an instant messaging platform.

3.7. Tool and Technologies

3.7.1. Frontend

Language: Javascript
Required run-time environment: Node.js
SDK: Ionic Framework(Ionic Framework, 2019)
Framework: Cordova(Cordova.apache.org, 2019)
IDE: Atom

3.7.2. Backend

Language: Java
Framework: Spring Boot Framework(Spring.io, 2019)
IDE: Eclipse IDE for JEE (Eclipse.org, 2019)
Build Manager: Gradle

3.7.3. API Design

Following OpenAPI 3.0(GitHub, 2019) standard.

3.8. Peer Review Process

3.8.1. Github branch strategy

Github branching strategy ensures that every commit is reviewed and approved by at least two team members.

3.8.2. Meetings

In general, when making the decision, team will discuss together and vote. If there is a conflict of interest, for example, a tie in voting considering we have six members, team will seek suggestions from TA.

- **Scrum Meetings**

Daily scrum meetings help to keep the team on track and highlight any issues/blocks early.

- **Backlog Grouping**

Before each sprint, we hold meeting deciding on the estimation of each task to be done in next sprint and split up them to each pair. The mechanism of estimating the point for tasks, each member votes for the point using poker.

- **Weekly Reflection**

We have weekly reflection meeting which is for summarising the team performance for the week before and improvements can be made to the next.

4. Assumptions

The technical architecture is sufficient for the project to proceed, it covers the front end and back end and also the strategy that makes sure the team works as a whole.

5. References

Markdown Guide. [online] Markdownguide.org. Available at: https://www.markdownguide.org/ [Accessed 7 Feb. 2019].

npm. [online] Available at: https://www.npmjs.com/ [Accessed 7 Feb. 2019].

Gradle Build Tool. [online] Gradle. Available at: https://gradle.org/ [Accessed 7 Feb. 2019].

Jenkins. [online] Available at: https://jenkins.io/ [Accessed 7 Feb. 2019].

Docker. [online] Available at: https://www.docker.com/ [Accessed 7 Feb. 2019].

Build Amazing Native Apps and Progressive Web Apps with Ionic Framework and Angular. [online] Available at: https://ionicframework.com/ [Accessed 7 Feb. 2019].

Apache Cordova. [online] Available at: https://cordova.apache.org/ [Accessed 7 Feb. 2019].

Spring Projects. [online] Available at: https://spring.io/projects/spring-boot [Accessed 7 Feb. 2019].

Eclipse IDE for Java EE Developers | *Eclipse Packages*. [online] Available at: https://www.eclipse.org/downloads/packages/release/kepler/sr2/eclipse-ide-java-ee-developers [Accessed 7 Feb. 2019].

OAI/OpenAPI-Specification. [online] Available at: https://github.com/OAI/OpenAPI-Specification/blob/master/versions/3.0.0.md [Accessed 7 Feb. 2019].

The Best APIs are Built with Swagger Tools | *Swagger*. [online] Available at: https://swagger.io/ [Accessed 7 Feb. 2019].