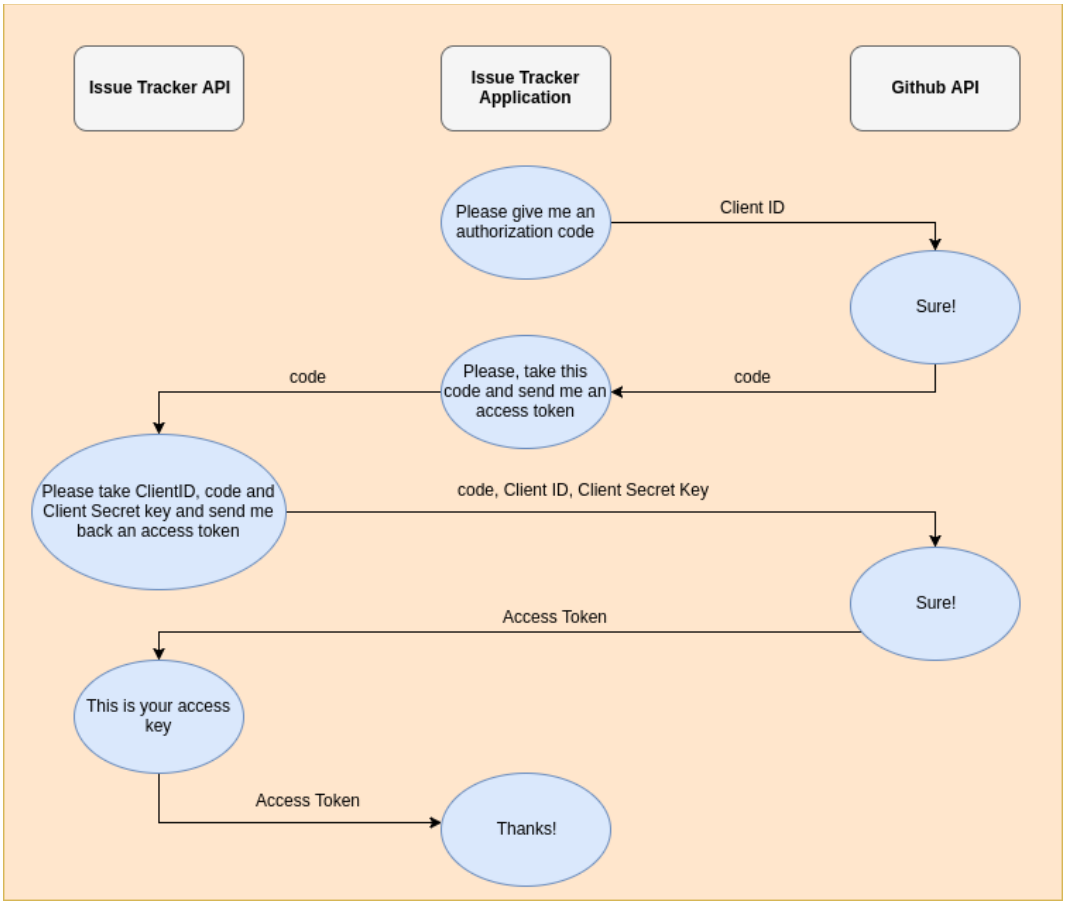
Issue Tracker Report

Introduction

In this report I would like to point out some features relatively with my choices as far as the project implementation is concerned, as well as same fundamental features that I had in mind realizing this project.

Security

The application communication through the API of Github demands some security rules to be served. For example, the system would be really vulnerable if during contact the user submitted the username/password. In this view, OATH architecture was used to cover the universal adoption of strong authentication. However, to implement this with absolute security, it is vital that such information as client secret key, should be transferred via a back end application for the reason why this personal information should be kept secret. So, a micro-sevice was realized in node.js having as a target to offer an access key which will be used for authorization during the communication of front-end and Github API. The whole procedure is depicted into the following scheme.



Deployment

API node was deployed into EC2 amazon server (public IP: 35.156.92.27) while the static react application uploaded in S3 amazon server at the link (http://asterios-issue-tracker.s3-website.eu-central-1.amazonaws.com). At the same time, the code of both them is found in Github in the following two github repos:

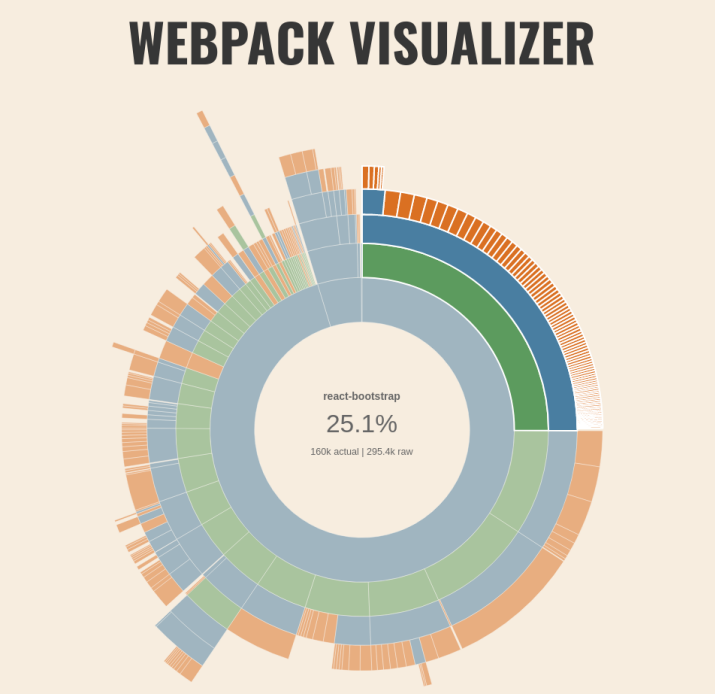
API: https://github.com/avounotr/issue-tracker-api

Application: https://github.com/avounotr/issue-tracker-app

Improvements

According to the needs of each project, we should make the appropriate decisions. When a project is quite demanding regarding the decrease both of its size and the time needed for its loading, then it is important that some improvements take place in the code and more specifically there should be used some tools and libraries which do not burden the system. Moreover, in some other cases chunking of the information uploaded on the system is absolutely necessary. So, a quite large-sized library can be uploaded when and if this is thought to be necessary during the users browsing of the application.

So, to make such decisions, a tool which estimates the volume of every library on the disk is thought to be of great importance. In this view, stats-webpack-plugin is used for this reason and in our case it gives the following results.



As we can see, react-bootstrap takes up the ~25% of the total space of the project.