International Institute of Information Technology, Bangalore

Software Production Engineering (CS 816)

Project Report

Problem Solving Tracker Application

Course Instructor: Prof. B. Thagaraju

Teaching Assistant: Yashasvi Khandelwal

Saravan Sriram

IMT2018521

Sriman Reddy Pingili

IMT2018525

Table of Contents

Abstract 3
- Important project links: 3

System Configuration 3
- Server Details 3
- CPU and RAM 3
- Programming Languages, Frameworks and Libraries 3
- Dependency Resolvers 4
- DevOps Tools 4

Application details 4
- App Functionalities/Features 4
- Application Screenshots 4
- Database functions 5

React Web Framework 6

LocalBase (Firebase style DB for offline storage) 8
- Usage with Keys: 9

Integration and Deployment Approach: 10

Source Code Management - Git/Github 10

CI/CD - GitHub Actions 11
- Continuous Integration: 12
- Generic pipeline template 12
- Github Actions 13
- Setting up github actions 13

Containerization-Docker and Dockerhub 14
- Dockerfile 14

Deployment-Ansible 15

Logging 16

Cloud-AWS EC2 17
- Steps: 18

CI/CD Workflow 20

Monitoring: ELK stack 21

Setup 22

Steps 22

Scope and Future Work: 25

References: **26**

Abstract

Preparing for interviews can be a tedious process. There are many resources but most of them are unstructured. For example, leetcode and geeksforgeeks have thousands of questions which can be overwhelming to someone preparing for an interview. The idea is to make this process less intimidating by providing users with a small set of questions based on tags/topics chosen by them.

In the following sections,we will get into the details of functionalities of the application itself followed by the DevOps pipeline starting from building to the continuous deployment and the logging.

Important project links

- GitHub Link to Project
- DockerHub Link

System Configuration

Server Details

- Ubuntu 18.04.04
- Bionic Beaver.
- v5.3 based Linux kernel

CPU and RAM

1 Core CPU, 1GB RAM on AWS EC2 (Infrastructure as a service)

Programming Languages, Frameworks and Libraries

- Frontend: React(HTML, CSS, Javascript)
- Backend: LocalBase (Firebase style DB for offline storage)
- Database: Local Storage

Dependency Resolvers

npm (is a package manager for the JavaScript programming language)

DevOps Tools

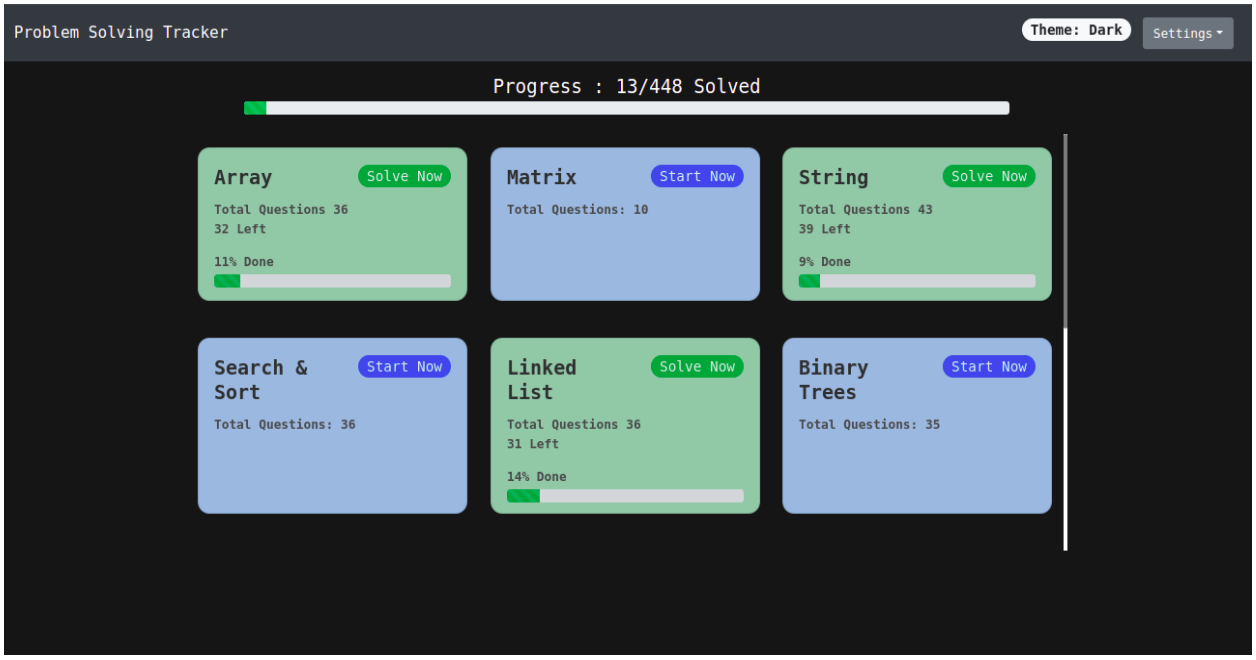
- Source Control Management - GitHub
- Continuous Integration - Github Actions
- Containerization - Docker
- Continuous deployment - Ansible
- Continuous monitoring - ELK Stack (Elastic Search, Logstash, Kibana)
- Cloud - AWS EC2

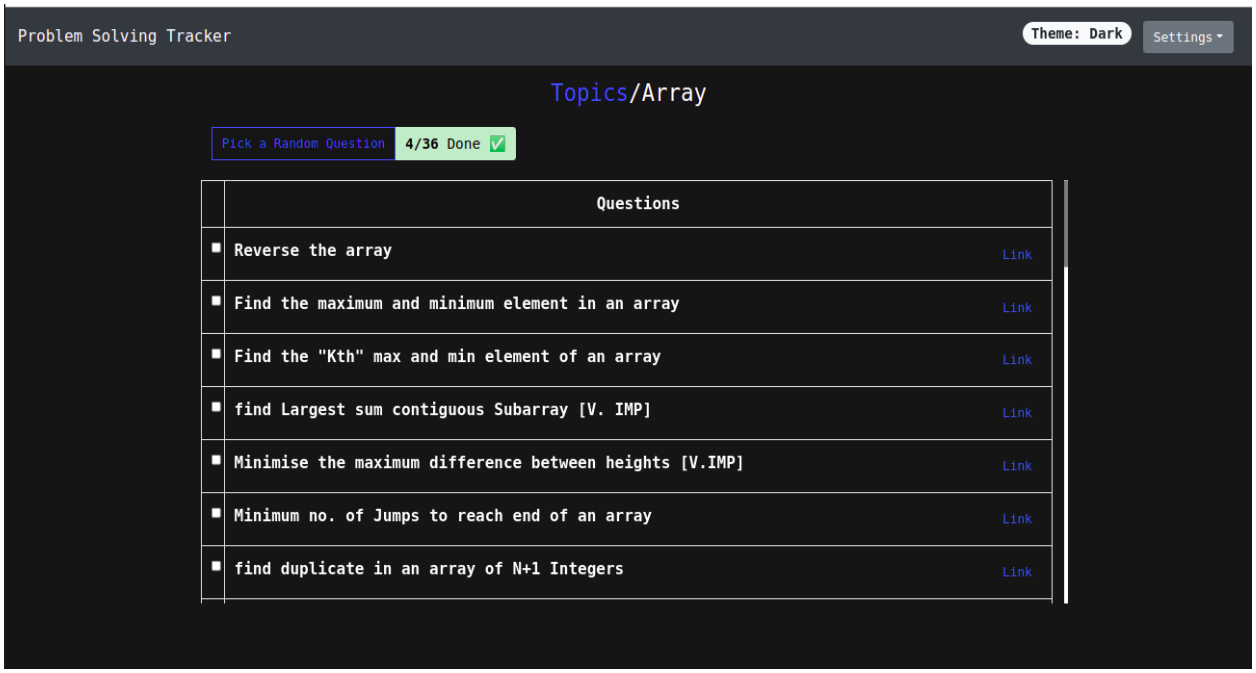
Application details

App Functionalities/Features

- Track progress: Shows number of questions solved and number of remaining questions overall and for each topic(Eg. Arrays, DP, etc).
- Users can click on a tag-card to get a list of questions.
- Users can click on the link button which redirects them to the external link of the actual question on a new tab.
- Users can click on the check-box to mark as finished. This increments the number of questions solved. Users can also uncheck the questions.
- The “marked as finished” questions go down the list of the other questions.
- Clean and simple UI.
- Ability to reset, import and export progress.
- Complete Local storage and hence it works offline.

Application Screenshots





Database Functions

- InsertData: Insert document data into collection
- GetData: Get data from database
- GetTopicData: Get all the questions from a specific topic
- UpdateDbData: Update database
- ResetDbData: Reset database to original state
- ExportDbData: Export database data into JSON file
- ImportDbData: Import new data from external file

React Web Framework

We use node.js to create the runtime environment for react web applications. Node.js is a JavaScript runtime for server-side programming. It allows developers to create scalable backend functionality using JavaScript. Node.js lets developers use JavaScript to develop a wide variety of applications like network applications, command line tools, web API, web applications.
React is an open-source, front end, JavaScript library for building user interfaces or UI components.React can be used as a base to develop the website pages or any mobile applications.

Installing nodejs v14.16.1 version and react 17.0.2 version:

First run the command to update your base system with the latest available packages.

```
apt-get update -y
```

Then install the required dependencies with the following command:

```
apt-get install curl gnupg2 -y
```

Once all the dependencies are installed, add the Node.js repository:

```
curl -sL https://deb.nodesource.com/setup_14.x | bash -
```

Once the repository is added, install Node.js:

```
apt-get install nodejs -y
```

Once Node.js is installed, update the NPM to the latest version:

```
apt-get install nodejs -y
```

You can now verify the installed version of Node.js

```
node -v
```

You need to install the create-react-app tool to create a new project in React.

```
npm install -g create-react-app
```

create a new project with the following command:

```
Create-react-app SPE-FinalProject
```

Then you will get “successfully created application ” as an output and after that go inside that directory and then run the command

```
npm start
```

Then this will start your web application on the port 3000 and you can access the web app by typing the following URL.

https://localhost:3000

LocalBase (Firebase style DB for offline storage)

Localbase gives you an offline database with the simplicity & power of Firebase, all stored in the user's browser (in an IndexedDB database). You can create as many databases as you like.

Databases are organised into Collections and Documents (just like Firebase Cloud Firestore).

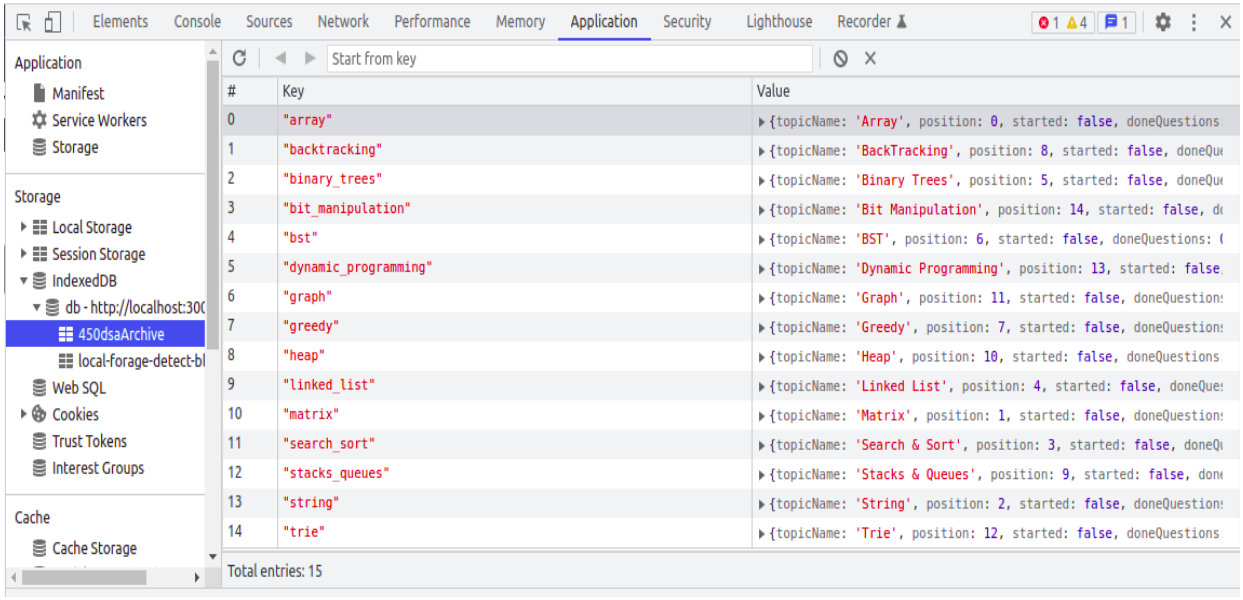
- Databases contain Collections
- Collections contain Documents

```
npm install localbase --save
```

```
import Localbase from 'localbase'
let db = new Localbase('db')
```

Usage with Keys:

Documents are stored in an IndexedDB store with keys:



By default, Localbase generates random, ordered, unique IDs for these keys.

LocalBase allows you to control the keys with the following functionalities:

- Specify your own key when you add a document.
- Use the key for selecting a document (when getting, updating, setting or deleting a document) instead of using some document criteria.
- Return all of the keys as well as the document fields, when getting a collection.

Integration and Deployment Approach:

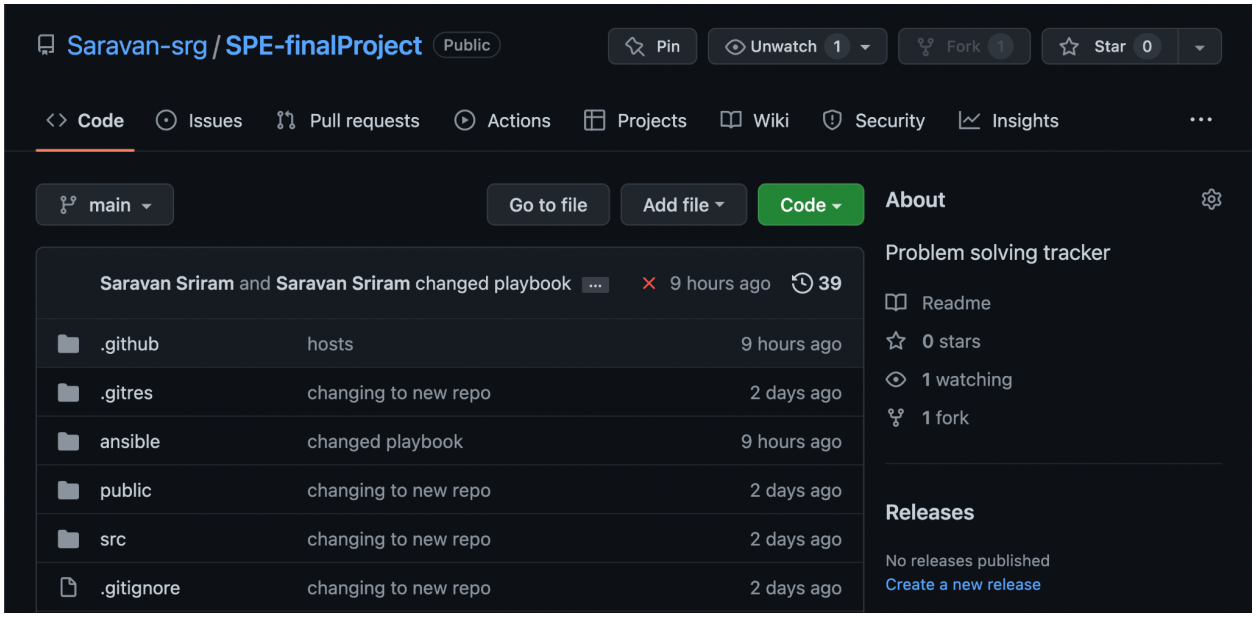
Since the application and its features/functionalities are very modular, they are subject to continuous changes with respect to feature additions. Therefore it becomes necessary to have a mechanism/pipeline that automates the steps starting from building the artifacts to deploying it on a server. Hence,we use the DevOps approach and DevOps tools to achieve the same. DevOps tools allow us to perform code version control, testing and building systems, configuration management and deploying the application in an automated way.

Source Code Management - Git/Github

GitHub is used as a source code management tool for the project. It is used for version control and source code management in the software industry. It makes the collaboration easier and simpler where the developers can work together from anywhere. A developer will initialize the local repository and push it to the remote repository using the following commands:

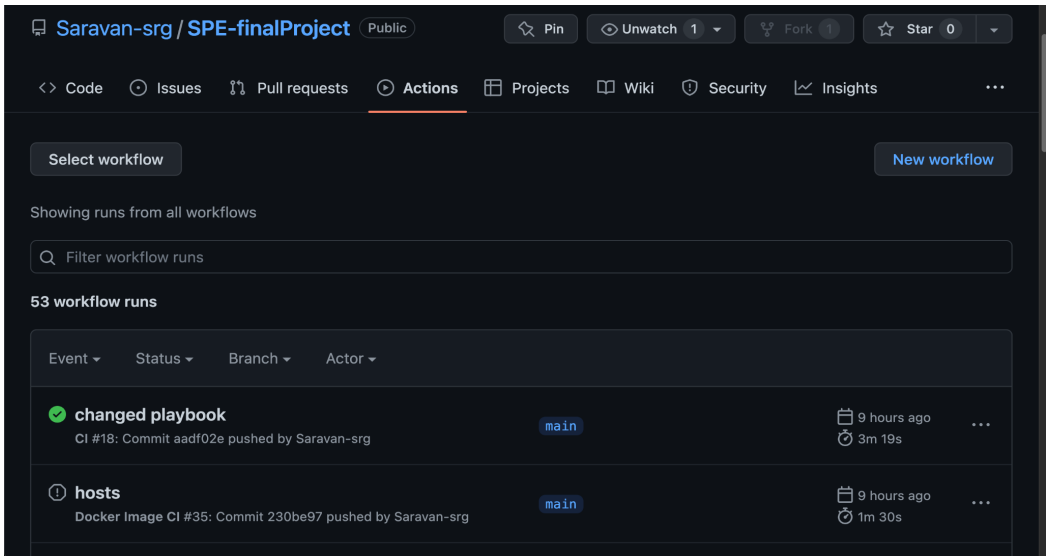
- git init : It creates a new repository for a new project.
- git add . : Add all files to the staging area
- git commit –m “message name”: Commit the added files to local repository.
- git remote add origin <repository_url>: Add the repository given as the working
- remote repository.
- git push origin master: For pushing local repository files to report GitHub repository.

Our remote GitHub repository: https://github.com/Saravan-srg/SPE-finalProject



CI/CD - GitHub Actions

Continuous Integration is a development practice where workflows like build, test and deploy are automated. Continuous Integration assures low integration risks, low down time, quality tests and automations, easy understanding between the teams and less time for deployment. Few Popular tools for CI are Jenkins, Travis CI, GitLab CI, Github actions and many more.



Continuous Integration

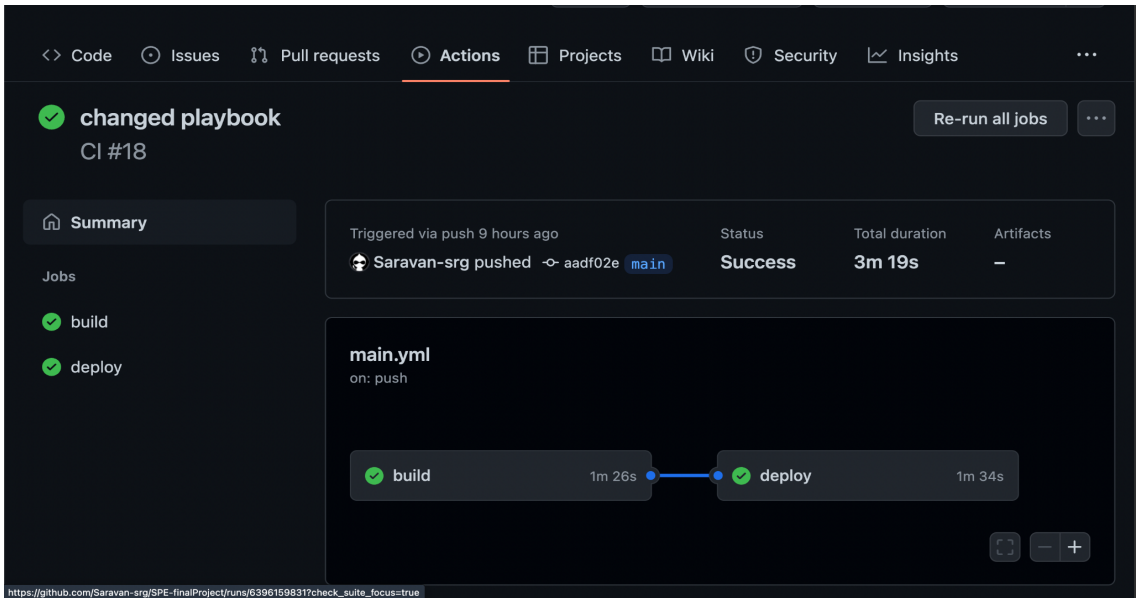
Continuous Integration (CI) is a development practice where developers integrate code into a shared repository frequently, preferably several times a day. We will be using Github actions for continuous integration. The pipeline will consists of independent workflows for each service which will build and push images to respective docker repositories.

Generic pipeline template

(a) Checkout code from github
(b) Perform unit tests and build step
(c) Build a docker image for our project using docker plugin
(d) Push the docker images to docker hub
(f) Send success/failure message to teams channel
(g) Deploy to cloud using ansible(more about Amazon EC-2 will be discussed later)

Github Actions

Github actions is a famous platform in recent times to automate developer workflows. CI/CD pipelines is one of the many workflows offered by github to automate work processes. Here the pipeline is two stages - build and deploy.



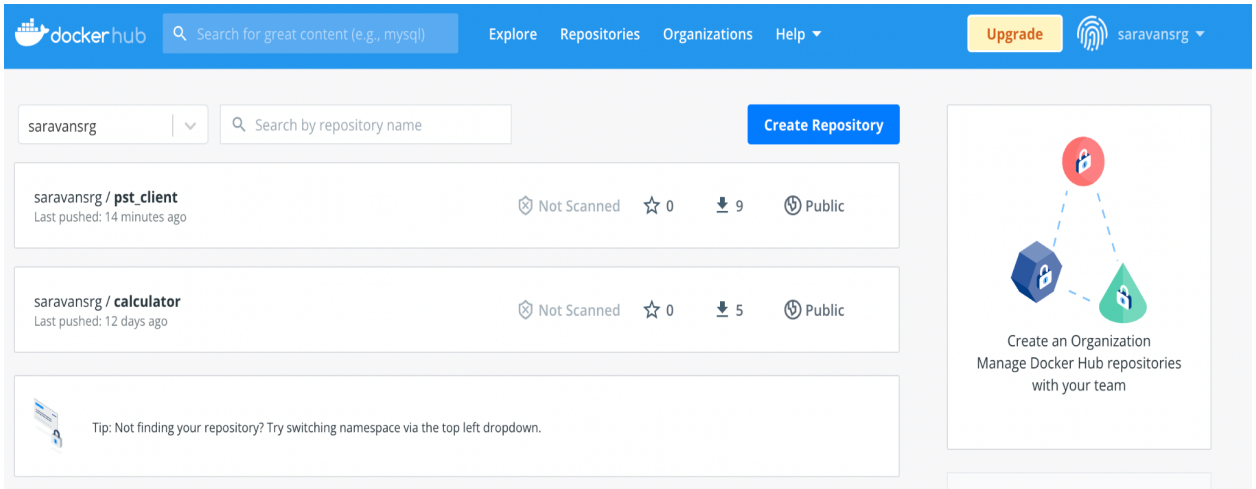
Setting up github actions

All the required YAML files are placed in the .github/ directory which will be triggered on github action basis. Workflow is a collection of jobs and these jobs will be runned on the trigger of an event. More about our project workflow will be discussed later.

Containerization-Docker and Dockerhub

Docker is an open source containerization platform. It enables developers to package applications into containers—standardized executable components combining application source code with the operating system (OS) libraries and dependencies required to run that code in any environment. Containers simplify delivery of distributed applications, and have become increasingly popular as organizations shift to cloud-native development and hybrid multi-cloud environments.

Dockerhub is a service provided by Docker for finding and sharing container images with your team.



Dockerfile

A dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image.

```
7 lines (7 sloc) | 118 Bytes
FROM node:alpine
WORKDIR /app
COPY package.json ./
COPY package-lock.json ./
COPY ./ ./
RUN npm i
CMD ["npm", "start"]
```

FROM: Tells docker what base image to use to build the new image. Here the base image is node:alpine
WORKDIR: Define the working directory in a docker container at any given time.
COPY: It is a docker command to copy files from local source location to a location in the docker container.
RUN: Executes any commands in a new layer on top of the current image and commits the results.
CMD: Specifies the instruction to be executed when a docker container starts.

Deployment-Ansible

Ansible is a tool that generates written instructions for automating IT professionals' work throughout the entire system infrastructure.
Ansible connects to nodes on a network (clients, servers, etc.) and then sends a little program called an Ansible module to each node. In this project we use ansible to pull the docker image from Dockerhub to a remote server(hosted on AWS EC-2) automatically.

```
8 lines (8 sloc) | 175 Bytes
---
- name: Pull docker image from dockerHub
  hosts: all
  tasks:
    - name: Pull docker image
      docker_image:
        name: saravansrg/pst_client
        source: pull
```

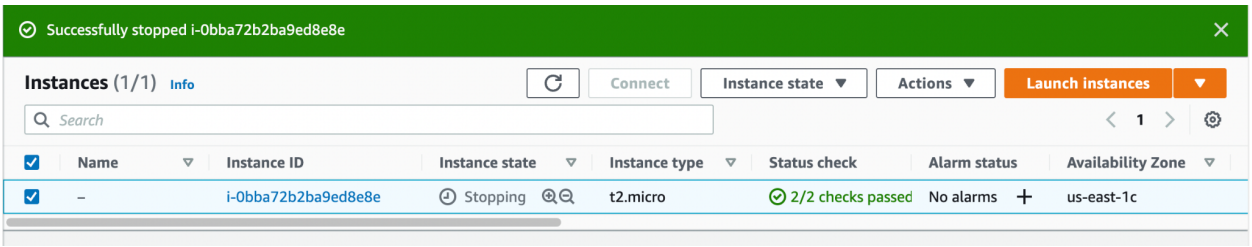
Logging

We used log4js which is similar to log4j in Java for logging purposes. This is how the log-file looks.

```
2022-05-11 22:53:33.223 [App.js] INFO  Click - Array
2022-05-11 22:53:35.223 [App.js] INFO  Click - Array
2022-05-11 22:53:42.223 [App.js] INFO  Click - Matrix
2022-05-11 22:53:52.223 [App.js] INFO  Click - Matrix
2022-05-11 22:53:52.223 [App.js] INFO  Click - Matrix
2022-05-11 22:53:52.223 [App.js] INFO  Click - BinaryTrees
2022-05-11 22:53:52.223 [App.js] INFO  Click - LinkedList
2022-05-11 22:53:52.223 [App.js] INFO  Click - LinkedList
2022-05-11 22:53:52.223 [App.js] INFO  Click - Array
2022-05-11 22:53:51.223 [App.js] INFO  Click - Array
2022-05-11 22:55:01.223 [App.js] INFO  Click - Trie
2022-05-11 22:55:03.223 [App.js] INFO  Click - Array
2022-05-11 22:55:04.223 [App.js] INFO  Click - Graph
2022-05-11 22:55:06.223 [App.js] INFO  Click - LinkedList
2022-05-11 22:55:07.223 [App.js] INFO  Click - Array
2022-05-11 22:55:07.223 [App.js] INFO  Click - Search&Sort
2022-05-11 22:55:08.223 [App.js] INFO  Click - Array
2022-05-11 22:55:08.223 [App.js] INFO  Click - BinaryTrees
2022-05-11 22:57:21.223 [App.js] INFO  Click - Array
2022-05-11 22:57:23.223 [App.js] INFO  Click - Heap
2022-05-11 22:57:25.223 [App.js] INFO  Click - Array
2022-05-11 22:57:27.223 [App.js] INFO  Click - Array
2022-05-11 22:57:28.223 [App.js] INFO  Click - Heap
2022-05-11 22:57:29.223 [App.js] INFO  Click - Trie
2022-05-11 22:57:30.223 [App.js] INFO  Click - Heap
2022-05-11 22:57:39.223 [App.js] INFO  Click - Array
```

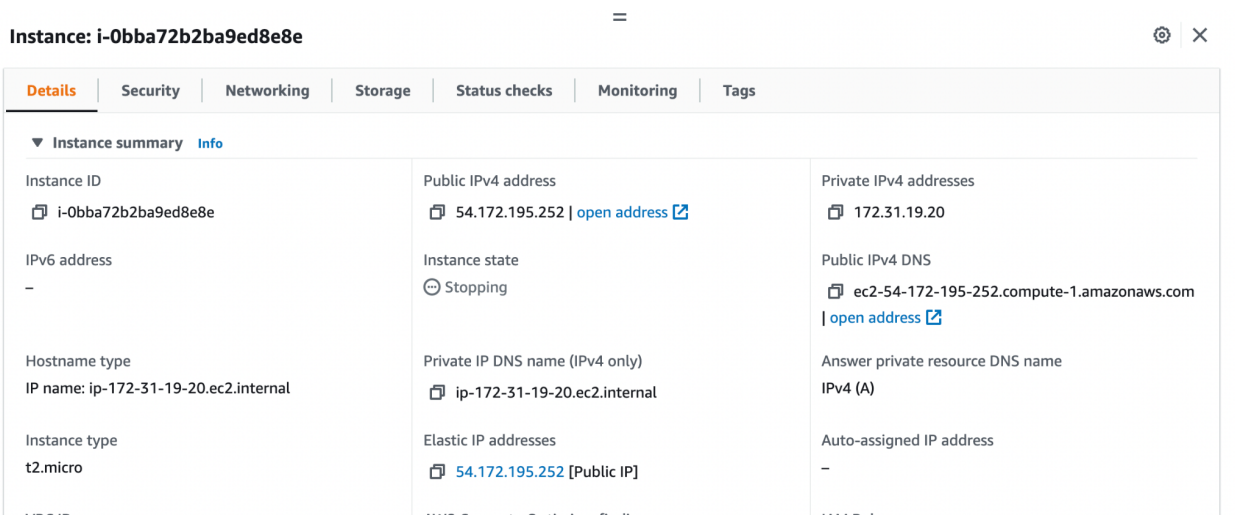
Cloud-AWS EC2

Amazon Elastic Compute Cloud (EC2) is a part of Amazon.com's cloud-computing platform, Amazon Web Services (AWS), that allows users to rent virtual computers on which to run their own computer applications. EC2 encourages scalable deployment of applications by providing a web service through which a user can boot an Amazon Machine Image to configure a virtual machine, which Amazon calls an "instance", containing any software desired(Ubuntu 18.04 server in this project). A user can create, launch, and terminate server-instances as needed, paying by the second for active servers – hence the term "elastic".

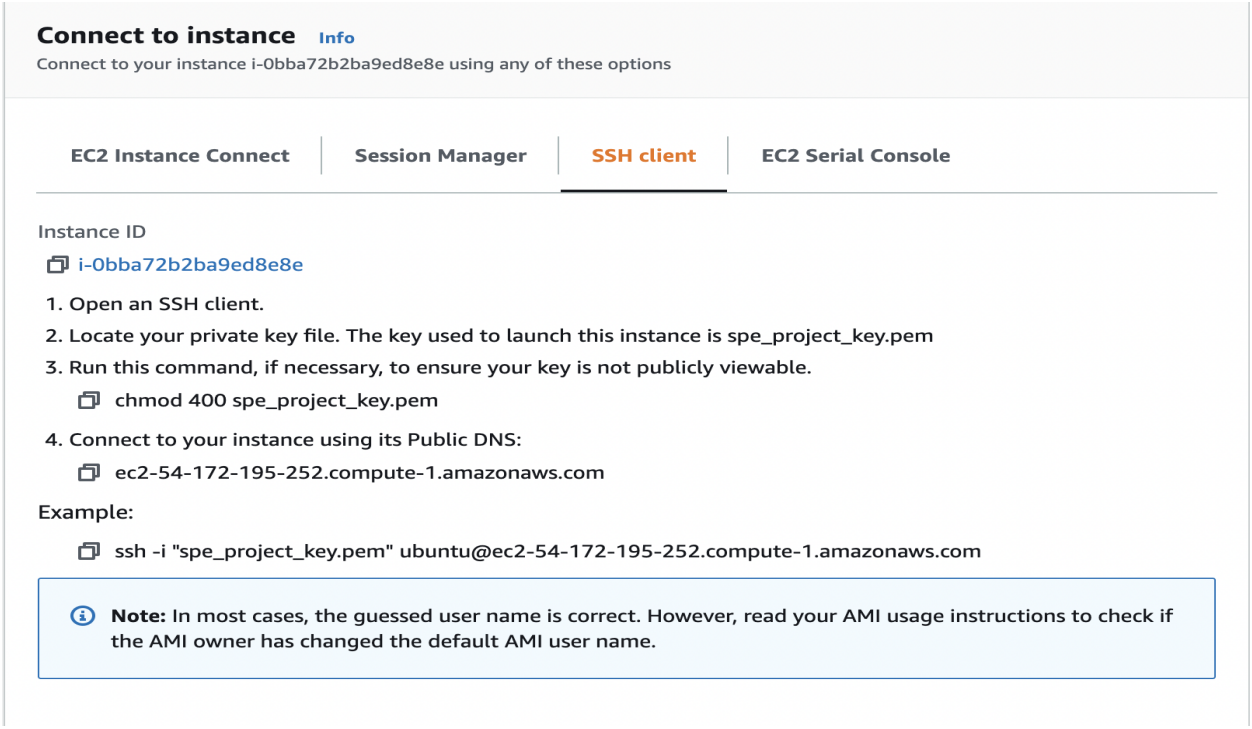


Setup

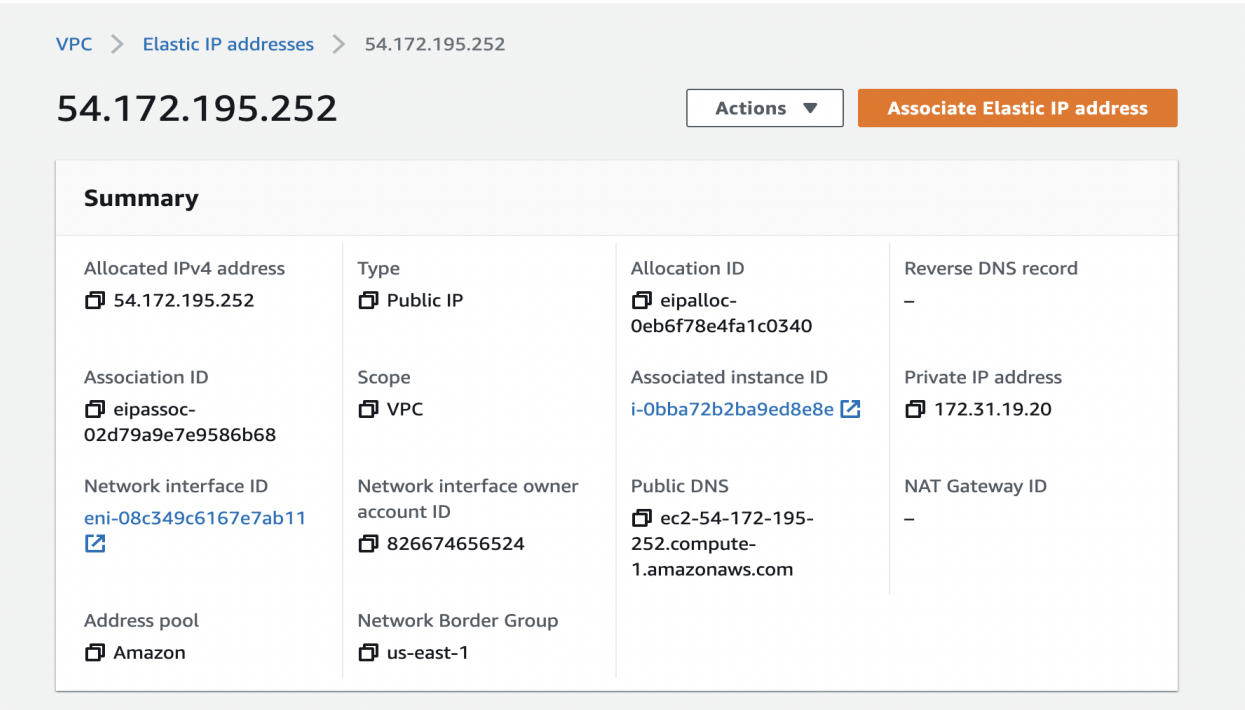
- Access the Management Console/Dashboard. Choose “Start Lab” and, when green, click on “AWS” (it will open a new console window)
- Under the “Build a solution”, choose “Launch a virtual machine with EC2”.
- Select an Amazon Machine Image. Preferably, Ubuntu 18.04 AMI (HVM). Then, choose the default (“t2.micro”) as Instance type.
- Select “Next” and review (or make changes to) the default configurations
- Finally, go to “Review and Launch” and launch your instance. While launching, you will be asked to select an existing key pair (“.pem” or “.cer” file). It is important to create a new key pair (if not already created before) and store it safely in your local file system.
- After launching the instance, click “View Instance”, to view the details of your instance and note the DNS and IP entries.



Connect to your EC2 instance via SSH as follows: Click on the “Connect” button in the top panel of the AWS EC2 console to get the instructions for running ssh. While executing the ssh command, make sure you provide the correct path to the “.pem” file in your local system. Also, always use the “Connect” button to know the correct ssh arguments to connect to your instance. The IP and DNS entries of your instance might change on stopping and restarting an instance.



It is possible to assign a static IP address to your instance using the Elastic IPs menu in the dashboard.



CI/CD Workflow

Github Actions is used for pipelining the complete project step by step. Push or pull requests trigger our github Actions but only for the main branch.

We can also use work_flow dispatch to run this workflow manually from the Actions tab. A workflow run is made up of one or more jobs that can run sequentially or in parallel. The workflow contains 2 jobs called "build" and “deploy”.

```yaml
# This is a basic workflow to help you get started with Actions
name: CI
# Controls when the action will run.
on:
  # Triggers the workflow on push or pull request events but only for the main branch
  push:
    branches: [ main ]
  pull_request:
    branches: [ main ]
  # Allows you to run this workflow manually from the Actions tab
  workflow_dispatch:
```

Build steps:

- Login to docker hub
- Build a docker image using Dockerfile
- Push the built docker image to Docker Hub

```yaml

# A workflow run is made up of one or more jobs that can run sequentially or in parallel
jobs:
  # This workflow contains a single job called "build"
  build:
    # The type of runner that the job will run on
    runs-on: ubuntu-latest
    # Steps represent a sequence of tasks that will be executed as part of the job
    steps:
      # Checks-out your repository under $GITHUB_WORKSPACE, so your job can access it
      - uses: actions/checkout@v2

      - name: Login to Docker Hub
        uses: docker/login-action@v1
        with:
          username: ${{ secrets.DOCKER_HUB_USERNAME }}
          password: ${{ secrets.DOCKER_HUB_ACCESS_TOKEN }}
```

```yaml
      - name: Set up Docker Buildx
        id: buildx
        uses: docker/setup-buildx-action@v1

      - name: Build and push
        id: docker_build
        uses: docker/build-push-action@v2
        with:
          context: ./
          file: ./Dockerfile
          push: true
          tags: ${{ secrets.DOCKER_HUB_USERNAME }}/pst_client:latest

      - name: Image digest
        run: echo ${{ steps.docker_build.outputs.digest }}

```

Deploy step:

- It triggers the Ansible playbook

```yaml
  deploy:
      needs: build
      runs-on: ubuntu-latest

      steps:
      - uses: actions/checkout@v2
      - uses: ./.github/actions/ansible
        env:
          SSH_PASSWORD: ${{ secrets.SSH_PASSWORD }}

```

Monitoring: ELK stack

ELK is the acronym for three open source projects: Elasticsearch, Logstash, and Kibana.

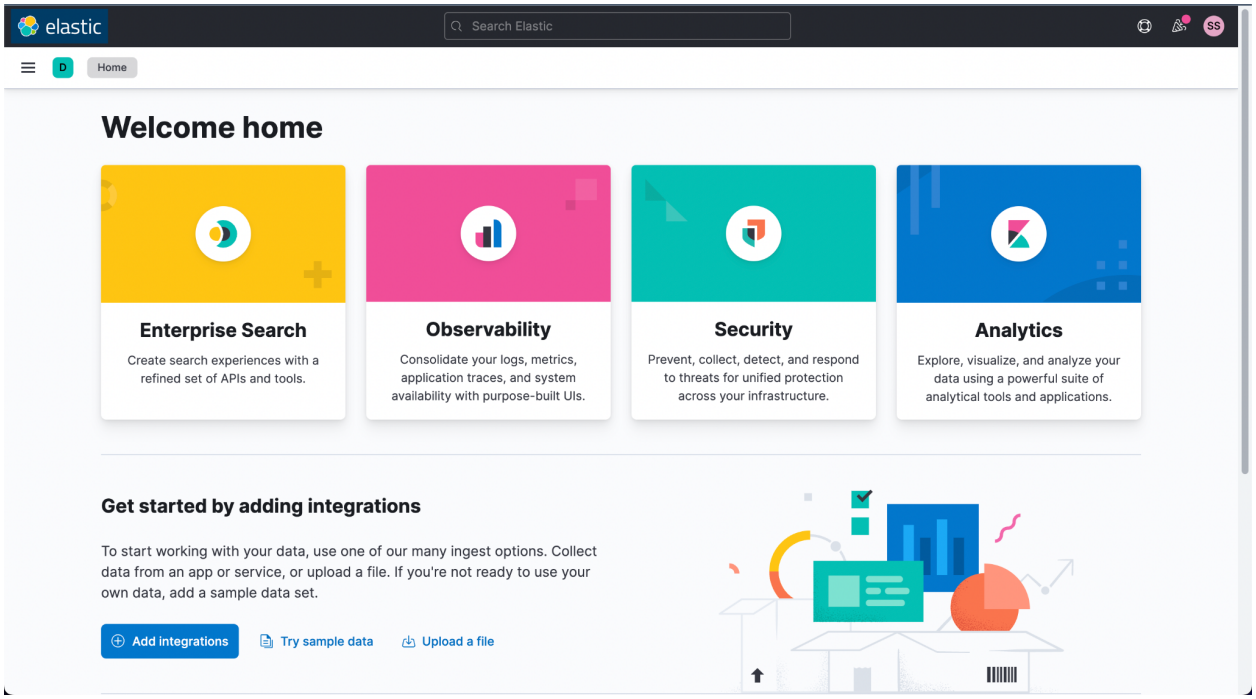
- Elasticsearch is a search and analytics engine.
- Logstash is a server side data processing pipeline that ingests data from multiple sources simultaneously, transforms it, and then sends it to a "stash" like Elasticsearch.
- Kibana lets users visualize data with charts and graphs in Elasticsearch.

Setup

Due to RAM constraints Elastic cloud was used for the project
Analysis was performed on the Unit Test cases used in the project.

Steps

1. Set up your elastic cloud account.
2. Go to the home-page and click on upload a file. This is how it looks.



3. Upload your log file and wait for it to parse and process it.
4. After upload is successful, the summary and file stats are displayed on the screen. Click on override settings and change the grok pattern according to how you wrote your log files. This is how it looks.

File contents

First 31 lines

```
2022-05-11 22:53:33.223 [App.js] INFO  Click - Array
2022-05-11 22:53:35.223 [App.js] INFO  Click - Array
2022-05-11 22:53:42.223 [App.js] INFO  Click - Matrix
2022-05-11 22:53:52.223 [App.js] INFO  Click - Matrix
2022-05-11 22:53:52.223 [App.js] INFO  Click - Matrix
2022-05-11 22:53:52.223 [App.js] INFO  Click - BinaryTrees
2022-05-11 22:53:52.223 [App.js] INFO  Click - LinkedList
2022-05-11 22:53:52.223 [App.js] INFO  Click - LinkedList
2022-05-11 22:53:52.223 [App.js] INFO  Click - Array
2022-05-11 22:53:51.223 [App.js] INFO  Click - Array
2022-05-11 22:55:01.223 [App.js] INFO  Click - Trie
2022-05-11 22:55:03.223 [App.js] INFO  Click - Array
```

Summary

Number of lines analyzed	31
Format	semi_structured_text
Grok pattern	%{TIMESTAMP_ISO8601:timestamp} \[.*?\] %{LOGLEVEL:loglevel} %{GREEDYDATA:Action} - %{GREEDYDATA:Topic}
Time field	timestamp
Time format	yyyy-MM-dd HH:mm:ss.SSS

Override settings Analysis explanation

Integrations > Upload file

File stats

All fields 5 of 5 total **Number fields** 0 of 0 total Field name 5 Field type 3

Type	Name	Documents (%)	Distinct values	Distributions
k	Action	31 (100%)	1	1 category
k	Topic	31 (100%)	8	8 categories

Action — DOCUMENTS STATS

count	31
percentage	100%
distinct values	1

Action — TOP VALUES

Click 31 (100%)

Topic — DOCUMENTS STATS

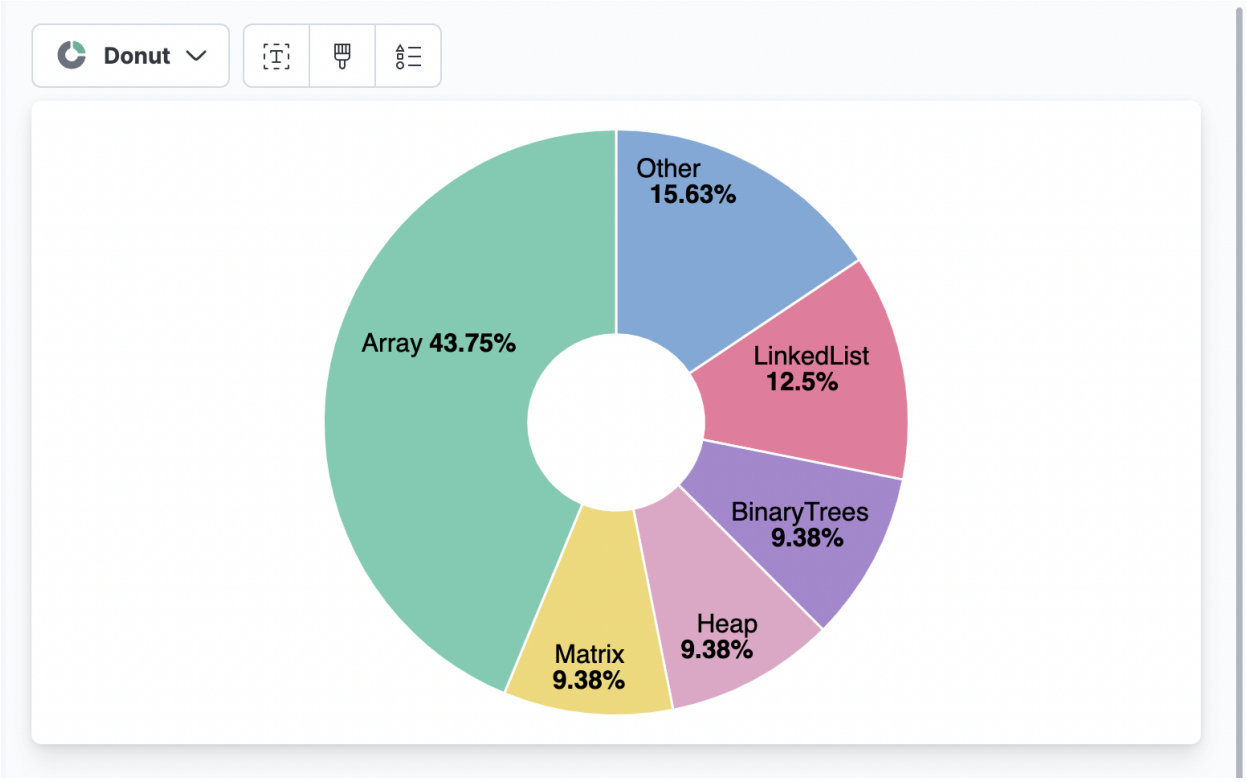
count	31
percentage	100%
distinct values	8

Topic — TOP VALUES

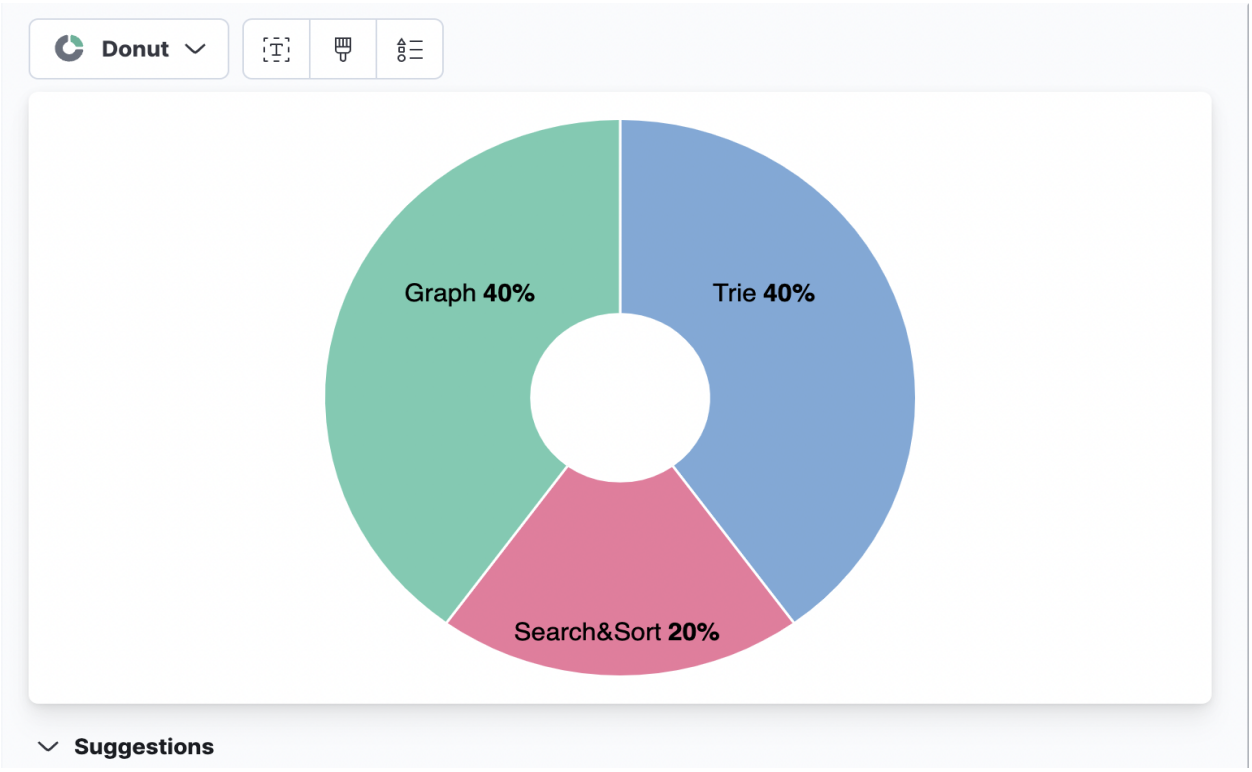
Array	14 (45.2%)
LinkedList	4 (12.9%)
BinaryTrees	3 (9.7%)
Heap	3 (9.7%)
Matrix	3 (9.7%)
Trie	2 (6.5%)
Graph	1 (3.2%)
Search&Sort	1 (3.2%)

5. Click on import and choose an index name. This allows you to open a lens dashboard to start visualizing. Below are some of the visualizations.

%Number of clicks of each topic:



%Number of clicks of Other topics:



%Bar graph of various fields together

Scope and Future Work

For now we have created a platform for the users to track the number of problems and to choose questions based on the topic accordingly with limitations. This application can be further extended with the following features to make the platform more user-friendly and interactive:

- Add *mark for review\favorite* feature
- Collaboration between users
- Better State Management
- Ability to add notes

References

1. https://www.npmjs.com/
2. https://reactjs.org/
3. https://hub.docker.com/
4. https://aws.amazon.com/ec2/
5. https://www.npmjs.com/package/localbase/
6. https://github.com/dannyconnell/localbase/
7. https://cloud.elastic.co/
8. https://stackoverflow.com/
9. https://medium.com/