

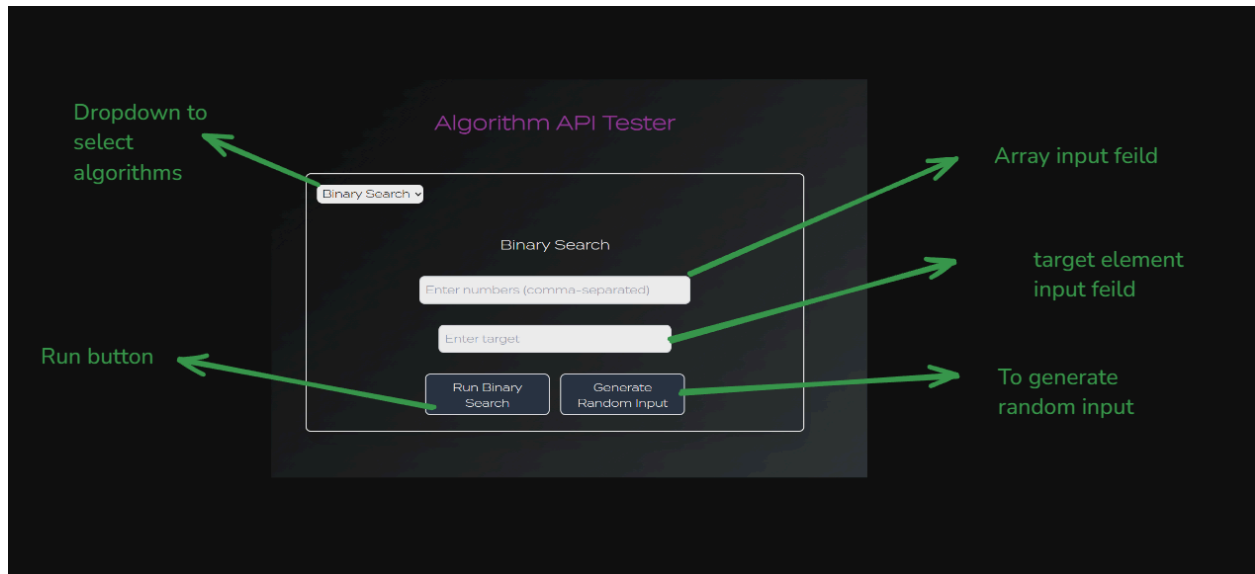
# Guide to use project

## Deployed URLs

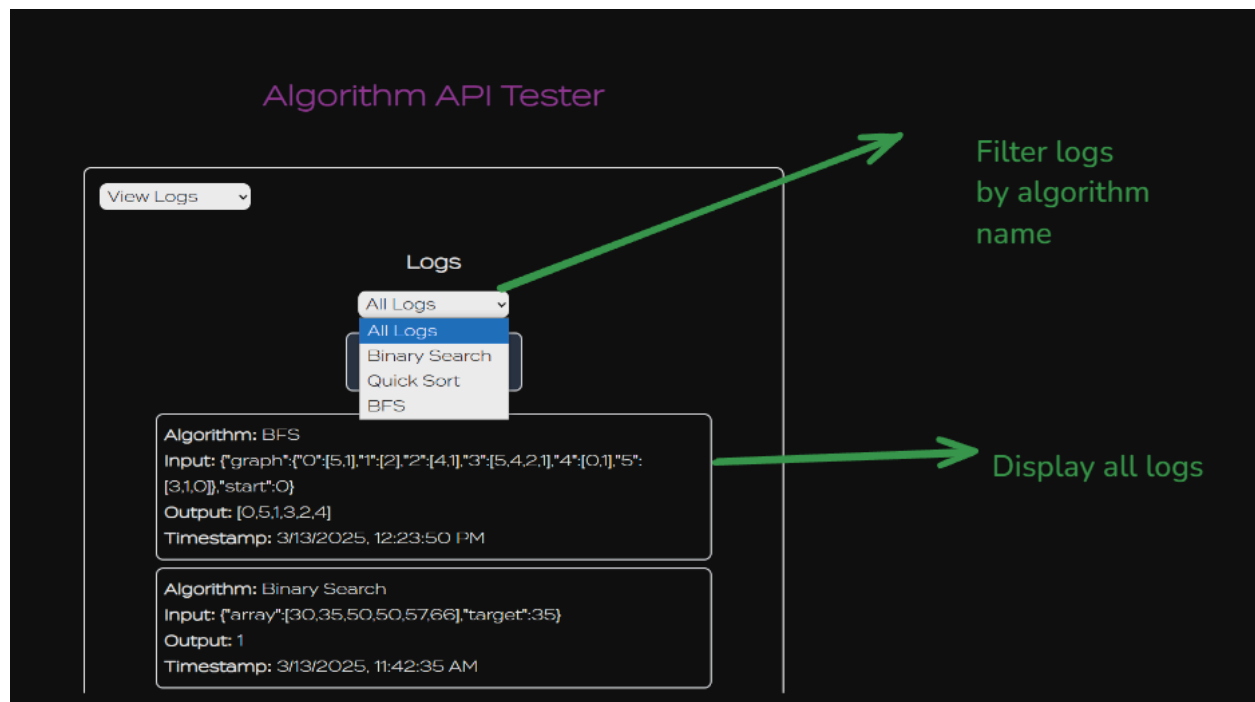
Frontend -> <https://algo-api-sigma.vercel.app/>

Backend -> <https://algo-api-vg1d.onrender.com/>

## Home Page



## Display Logs



## Backend:

Folder structure :

index.js -> starter file for backend

Api :

ApiError -> To send consistent error messages responses from backend to frontend.

ApiResponse -> To send consistent responses from backend to frontend.

asynchHandler -> Simplifies error handling in async Express routes.

config :

dbConfig.js -> This file is responsible for connecting the database.

controllers :

bfs.js -> Implementation of BFS algorithm.

binarySearch.js -> Implementation of BFS algorithm.

quickSort -> Implementation of quickSort algorithm.

Logs -> create, get all logs, get logs by name.

models :

log.js -> MongoDB Schema to store logs in the database.

```
const mongoose = require("mongoose");

const LogSchema = new mongoose.Schema({
  algorithm: String,
  input: mongoose.Schema.Types.Mixed,
  output: mongoose.Schema.Types.Mixed,
  timestamp: { type: Date, default: Date.now },
});

module.exports = mongoose.model("Log", LogSchema);
```

routes :

algorithm.js -> Define algorithm routes,

```
const express = require("express");
const { binarySearchAPI } = require("../controllers/binarySearch");
const { quickSortAPI } = require("../controllers/quickSort");
const { bfsAPI } = require("../controllers/bfs");

const router = express.Router();

router.post("/binary-search", binarySearchAPI);
router.post("/quick-sort", quickSortAPI);
router.post("/bfs", bfsAPI);

module.exports = router;
```

logRoutes -> Define logs routes,

Dockerfile -> To run backend through Docker

## Frontend:

App.jsx -> Starter file for FrontEnd,

Public :

fonts -> this folder contains all fonts I used in this project.

Components :

Bfs.jsx -> Display BFS output,

BinarySearch.jsx -> Display binary search output,

Quick sort -> Display quick sort output,  
Logs -> Display Logs output,

Dockerfile -> To run Frontend through Docker

Docker-compose.yml -> This file runs both the frontend and backend in a single command.

`docker-compose up --build`