A summary report detailing the approach taken, challenges faced, and any suggestions for improvement.

Aman Kumar Singh

amankumar7355197337@gmail.com

Approach Taken

The approach taken to create this github app was to create a new probot app using this command: npx create-probot-app <app_name>. After that I modified the permissions of the github app from app.yml file. This app will run when a user will open/close/reopen the issue of a repository and initially I am getting the body of the comment from the payload. Following which I am checking if the body includes the command(/execute) or not. If the body does not include the command, the bot will generate a comment saying ‘No /execute command found’. If the body consists of the command, it will check if the body has any language and code. If the body does not have any code or language, the bot will generate a comment saying ‘code and/or language not found in the expected format’. If the body has both the code and language, it will extract the code and language from the body. After extracting the code and language. The ‘piston public api’ is used to execute the code extracted from the comment body. This api takes two parameters, (i) language, (ii) source code. After executing the code with the help of piston api, the bot will generate a comment with the output of the code.

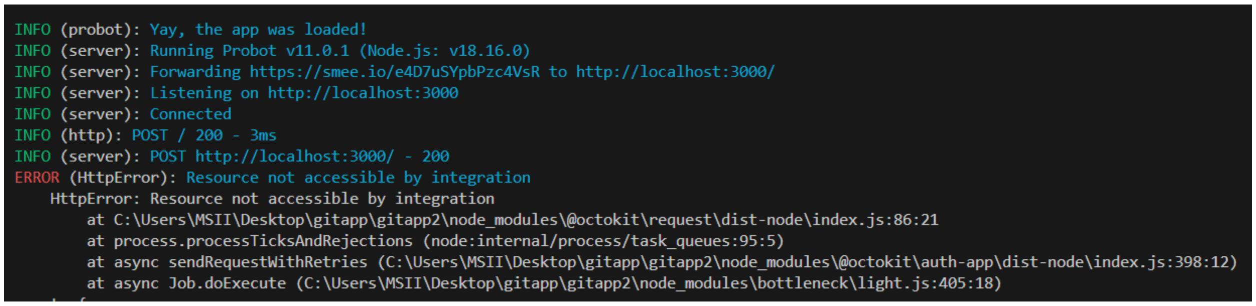
Challenges Faced

There were difficulties switching to a "pull_request" event triggering. Rethinking event triggers was necessary in order to comprehend payload structure and requests for pull requests as opposed to issues. I found it difficult to modify current logic to process pull request events. At the time of pull_request I encountered the following error:

Code:

```
/**
 * This is the main entrypoint to your Probot app
 * @param {import('probot').Probot} app
 */

module.exports = (app) => {
  // Your code here
  app.log.info("Yay, the app was loaded!");
  app.on(["issues.opened", "issues.closed", "issues.reopened",
"pull_request.opened"], async (context) => {

    const issuePayload = context.payload;
    const owner = issuePayload.repository.owner.login
    const repo = issuePayload.repository.name
    const number = issuePayload.number
    const sha = issuePayload.pull_request.head.sha
    const res = context.issue({
      owner: owner,
      repo: repo,
      pull_number: number,
      commit_id: sha,
      body: `Hello, ${owner}`,
      path: 'README.md',
    })
    return context.octokit.issues.createComment(res)
  });

  // For more information on building apps:
  // https://probot.github.io/docs/

  // To get your app running against GitHub, see:
  // https://probot.github.io/docs/development/
};
```

Error:



```
INFO (probot): Yay, the app was loaded!
INFO (server): Running Probot v11.0.1 (Node.js: v18.16.0)
INFO (server): Forwarding https://smee.io/e4D7uSYpbPzc4VsR to http://localhost:3000/
INFO (server): Listening on http://localhost:3000
INFO (server): Connected
INFO (http): POST / 200 - 3ms
INFO (server): POST http://localhost:3000/ - 200
ERROR (HttpError): Resource not accessible by integration
    HttpError: Resource not accessible by integration
        at C:\Users\MSII\Desktop\gitapp\gitapp2\node_modules\@octokit\request\dist-node\index.js:86:21
        at process.processTicksAndRejections (node:internal/process/task_queues:95:5)
        at async sendRequestWithRetries (C:\Users\MSII\Desktop\gitapp\gitapp2\node_modules\@octokit\auth-app\dist-node\index.js:398:12)
        at async Job.doExecute (C:\Users\MSII\Desktop\gitapp\gitapp2\node_modules\bottleneck\light.js:405:18)
```