

National College of Ireland
Project Submission Sheet

Name	Chethan Mundigehalla Prabhakar
Student-id	X23297395
Module	Cloud Dev-sec-ops
Year	2024
Course	Cloud Computing
Lecturer	Adriana E. Chis
Project Title	Job Portal
Submission Date	09-12-2024
Due Date	09-12-2024
Word Count	2531

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the references section. Students are encouraged to use the Harvard Referencing Standard supplied by the library. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action. Students may be required to undergo a viva (oral examination) if there is suspicion about the validity of their submitted work.

Signature	Chethan Mundigehalla Prabhakar
Date	09-12-2024

PLEASE READ THE FOLLOWING INSTRUCTIONS:

1. Please attach a completed copy of this sheet to each project (including multiple copies).
2. Projects should be submitted to your Programme Coordinator.
3. You must ensure that you retain a **HARD COPY** of ALL projects, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. Please do not bind projects or place in covers unless specifically requested.
4. You must ensure that all projects are submitted to your Programme Coordinator on or before the required submission date. Late submissions will incur penalties.
5. All projects must be submitted and passed in order to successfully complete the year. Any project/assignment not submitted will be marked as a **failure**.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

AI Acknowledgement Supplement
Cloud Dev-Sec-Ops
Job Portal

Your Name/Student Number	Course	Date
Chethan Mundigehalla Prabhakar	MSc in Cloud Computing	09-12-24

This section is a supplement to the main assignment, to be used if AI was used in any capacity in the creation of your assignment; if you have queries about how to do this, please contact your lecturer. For an example of how to fill these sections out, please click [here](#).

AI Acknowledgment

This section acknowledges the AI tools that were utilized in the process of completing this assignment.

Tool Name	Brief Description	Link to tool
NA	NA	NA

JobPortal

<http://x123456789jobportal-env.eba-3cssn4ck.us-east-1.elasticbeanstalk.com/>

Chethan Mundigehalla Prabhakar

Student-Id: 23297395

Cloud Dev-Sec-Ops, MSc in Cloud Computing

National College of Ireland Dublin, IRELAND

Email: x23297395@student.ncirl.ie. URL: www.ncirl.ie

Job Portal

Given Name Chethan Mundigehalla Prabhakar

StudentId: 23297395

Cloud DevOpsSec, MSc in Cloud Computing

National College of Ireland Dublin, IRELAND

Email: x23297395@student.ncirl.ie. URL: www.ncirl.ie

Deployed Application URL: <http://x123456789jobportal-env.eba-3cssn4ck.us-east-1.elasticbeanstalk.com/>

Abstract : The Job Portal project is a Web application. It aims to connect job seekers with employers who have positions to offer. There is server-side processing software such as Python Django in the backend service structure, and HTML and Bootstrap in the front end. Parts of the project such as this one use Continuous Integration and Continuous Deployment (CI/CD) with AWS Code Pipeline for ongoing development and installation automation. A private GitLab repository hosts source code, to ensure version control and collaboration is well organized. Sonar Cloud is used for static code analysis, which records code quality, reliability, and safety issues and links them directly to the project GitHub repo. This allows us to integrate integrity of code easily and makes development cycles faster and higher standards of software quality achievable. Introduction: Job Portal is a scalable, secure and maintainable platform that precisely meets today's dynamic job market requirements.

I. INTRODUCTION

With the job market moving at the speed of light and fiercely competitive employers requiring graduates on tiptoe to work for them, it is absolutely essential that qualified candidates find the right employment. Towards this huge unfilled demand, Job Portal solves

the problem by providing a Web based platform with rich features. This satisfies both sides of what perl-Employer staff and Perl candidate employees to communicate. It includes an HTML and Bootstrap-based web interface that is simple to learn and provides seamless navigation both on the desktop and across many devices. For the backend of the application, Python Django is used as a well-equipped web framework with the slogan 'Being pragmatic and ambitious'. It is responsible for user authentication, job listings, application submissions, management of administrative subscriptions and other key functions that means that neither robots can undermine the platform nor other humans break it down.

To ensure high-level code quality It depends upon AWS Code Pipeline for an extremely comprehensive CI/CD pipeline implementation. This automated procedure allows for code to build itself, test it and deploy. In a privately held GitHub repository, source code is safely stored and becomes the basis upon which to collaboratively develop great software that includes robust versioning control.

In addition, Sonar Cloud is integrated for continual static code analysis, making sure the application remains secure, maintainable, and free of critical vulnerabilities. The tight union of GitHub and Sonar Cloud provides real-time scanning and feedback assistance. This will help developers maintain high coding standards throughout the entire life cycle.

II. ARCHITECTURE & DESIGN

First of all let's talk about the frontend Job Portal uses the HTML and Bootstrap as frontend and Python Django as backend framework. The Job portal is a user friendly application which has signup page in which a user can either register as recruiter or an Applicant

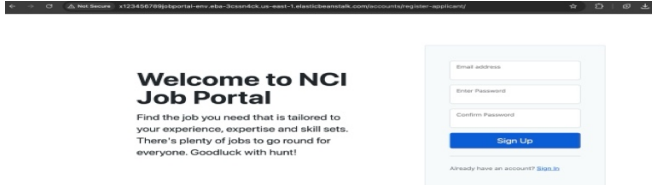


Fig 2.1 Signup page

Once the user login to the application using the login portal of the application user will be landed in recruiter page or Applicant page based the information given by user on the time of Register.

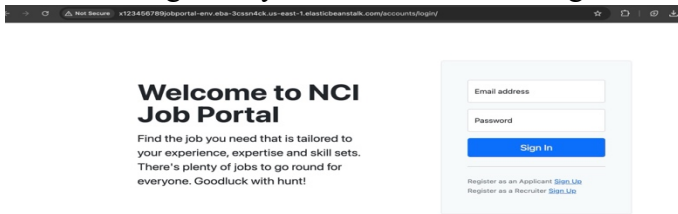


Fig 2.2 Login Page

The recruiter can register the company and post the jobs according to the needs of the company. And view the applicants who applied for the job as well.

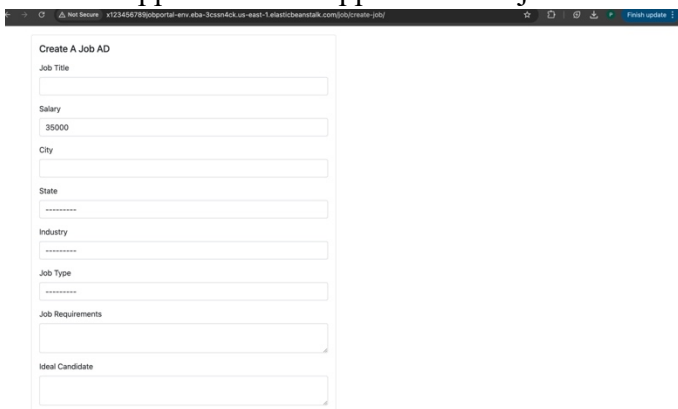
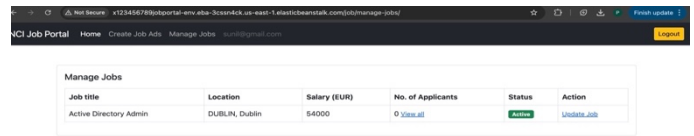


Fig 2.3 Create a Job



Job title	Location	Salary (EUR)	No. of Applicants	Status	Action
Active Directory Admin	DUBLIN, Dublin	54000	0 Views	Active	Unassign Job

Fig 2.4 View the Jobs

The applicant can login and view for all the jobs available once he uploads his CV he will be able to apply for the jobs available. The recruiter can download the CV of applicant and view it on his local computer. If the recruiter find he is the ideal candidate he can contact him.

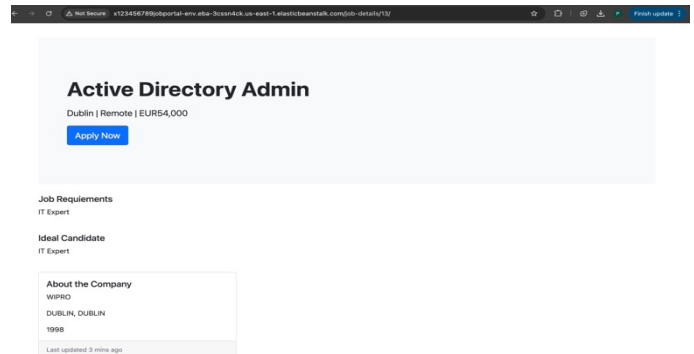


Fig 2.5 Apply for Jobs

The application uses the default database given by Django i.e. SQLite . Attached picture of Django admin-site.

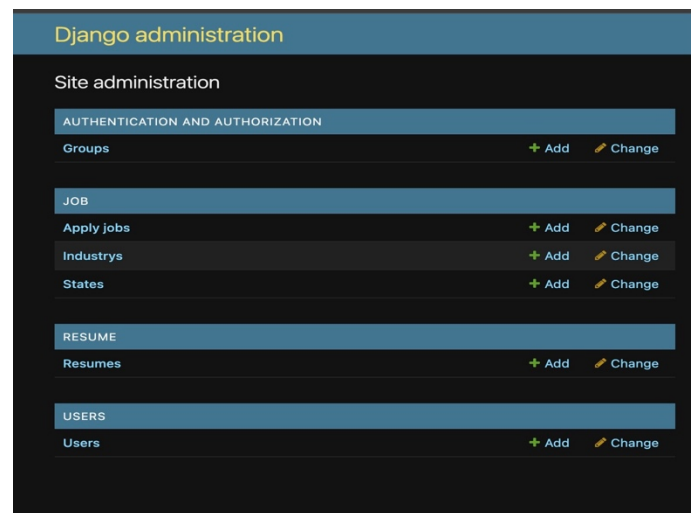


Fig 2.6

As you can see there are around 3 database created Jobs, resume and users in which users is inbuilt. The user database stores the user data such as mail-id, Password and it also stores if a user is a recruiter or an applicant. The Jobs database has 3 fields Apply job field stores if any user has applied and which position he has applied. The industry field can store the industry in my application I have 3 industries construction, Teaching and Technology. State field stores the states such as Dublin, Galway, Waterford etc. The resume database stores the resume of the user who has uploaded.

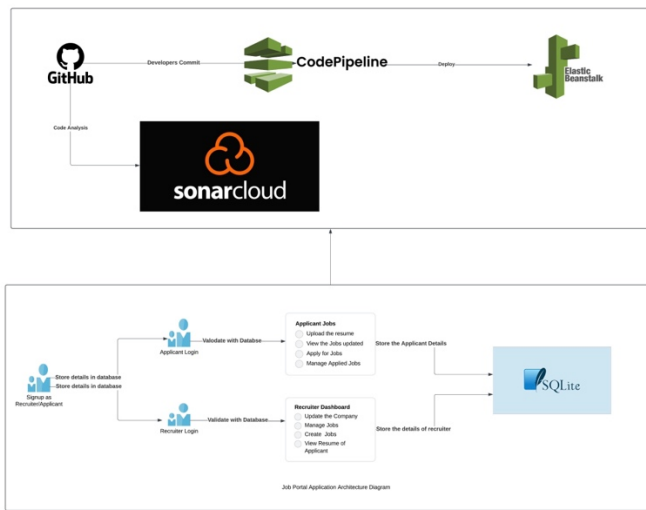


Fig 2.7 Architecture Diagram

The application uses the AWS code pipeline for CICD as soon as the developer commits the changes to the GitHub the pipeline will trigger and starts Integrating with the application. If the Integration is successful next step will be the deployment, the application will be deployed in the elastic beanstalk. In this application we are using cloud9 as IDE.

III. CONTINUOUS INTEGRATION, CONTINUOUS DELIVERY AND DEPLOYMENT

One important job for Continuous Integration(CI) and Continuous Deployment(CD) is to use them together as conveyor belts to complete work for Job Portal. In doing so, any changes to the source code

trigger an automated process while requiring as little human input as possible. All of this depends on the deployment methods, environment configuration and project setting into structure before everything passes through Amazon resources. A complete CI/CD pipeline has been built with AWS Code Pipeline that connects to GitHub for source code control and Elastic Beanstalk to handle application deployment. The CI / CD process is a fully automated work flow after every code change in the GitHub repository. When developers commit their changes this triggers the pipeline – building, testing and deploying the application with AWS Elastic Beanstalk. The arrangement ensures that we deliver continuously and make high demands on reliability and scalability.

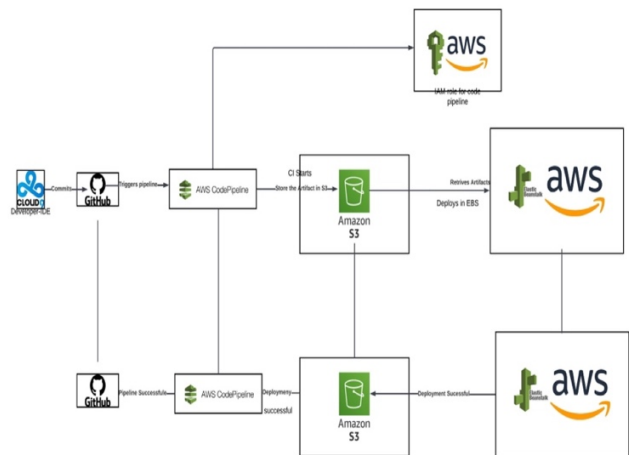


Fig 3.1 Architecture Diagram of CI-CD

The Job Portal Application uses AWS code pipeline for continues integration and continues deployment. Any push done by the developer to GitHub from cloud9 environment triggers the AWS code pipeline and deploys the application on Elastic beanstalk. The GitHub is integrated with EC2 instance of the clou9 environment using **ssh key** [1], so there is no need to enter any username and password every time he pushes the code. The developer uses `git add`, `git commit -m "any specific comments"` and `git push` command to push the code to the GitHub [2]. Attached screenshot of ssh key added to GitHub.

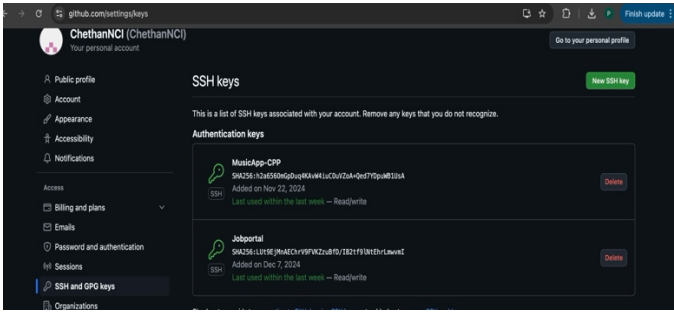


Fig 3.2 ssh key GitHub

The Job Portal application uses Elastic Beanstalk for deployment create a .ebextensions folder and create a django.config file inside this folder. screenshot attached below.

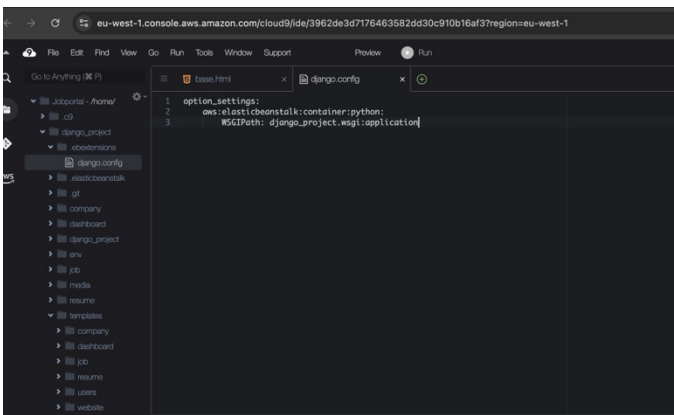


Fig 3.3 django.config file

The following command are used for deployment.
`eb init -r us-east-1 -p python-3.9 x23297395jobportal` : The `eb init` command is specifically used to set up and configure a new Elastic Beanstalk application. Here `-r` represents the region and `-p` represents the platform version and `x23297395jobportal` is the application name. The Jobportal application uses `us-east-1` as its region and `python 3.9` as its platform version.

`eb create <environmentname>`: This will create a environment for your application and you should replace the `<environment name>` with any related name to your application.

After this open the Elastic beanstalk and try to open the URL generated if your application is deployed successfully you will be able to access your application without any issues. [3]

Next step is to create AWS code pipeline. Create a pipeline with name in my case pipeline name is Jobportal. Setup the source stage you can either use AWS code commit, bitbucket or GitHub webhooks Job portal uses GitHub webhooks via OAuth App next you need to select the repository for AWS code pipeline In build stage you can select either select code build or you can skip I have skipped this part and in the deployment stage I have selected Elastic beanstalk and selected my application. Review your pipeline configuration and click on create pipeline. For this pipeline I have used the existing IAM role which is mentioned in the Moodle. You can verify your pipeline by changing the source code push it to GitHub and see if changes are getting reflected in deployed application.

Attached screenshot of successful code pipeline of Jobportal.

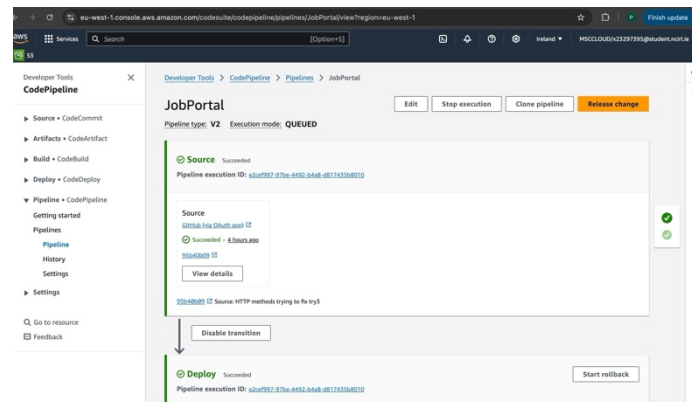


Fig 3.4 code pipeline

Job Portal Website URL :

<http://x123456789jobportal-env.eba-3cssn4ck.us-east-1.elasticbeanstalk.com/>

IV. STATIC CODE ANALYSIS

As the speed of software development accelerates the assurance of quality and code stability becomes increasingly important. Static code analysis is a popular and effective way for increasing code quality. It involves the examination of source code

without running it, identifying possible problems and giving developers practical tips on procedures to make their code more maintainable easier get along with, less buggy, and more hack-proof. When it comes to statistical code analysis, Sonar Cloud, a new cloud-based platform from Sonar Source, is the tool of choice for experts. [4]

Access project information with the Sonar Cloud dashboard It means the quality gate has been passed, signifying that the code conforms to specified quality standards. Additionally, important measures internet new issues detected, which point to recent compilation problems in the code, secondly, accepted issues, what indicates the dose as unresolved, but problems that have been recognized. It also shows the code coverage percentage, which tells you something about how much of the code is tested, and duplication metrics to make sure you have no duplicated code. It also keeps an eye on security hotspots to find potential vulnerabilities. The developer dashboard provides clear and actionable insights along with compliance to radar cloud coding standards, helping the developer stay on top of maintaining a high-quality and secure code. [5]

In Job portal application the Sonarcloud is integrated with my GitHub account so that every time I commit the code to GitHub it carries a analysis and provide the report with the security fixes that needs to be carried and all bugs that are present in the code. You can review the code see for solution as well [6].

The first analysis of sonar cloud was failed because of major security vulnerability of not including @loginrequired in the company view.py file. After adding and fixing this issue the second analysis was passed here is the screenshot of first fail.

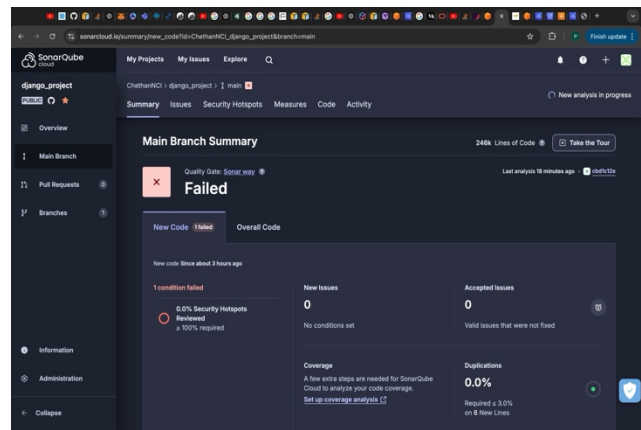


Fig 4.1

Django secret key in setting also had a vulnerability of security This has been resolved by adding an environment variable in Elastic beanstalk environment and invoking that environment variable in settings. py file. I'm going with 'HTTP' as I am using EBS for deployment purpose with public http domain limitations on both from implementing HTTPs as we need SSL certificate. Finally, the quality gate is passed after all changes are committed to the GitHub. Attached screenshot below.

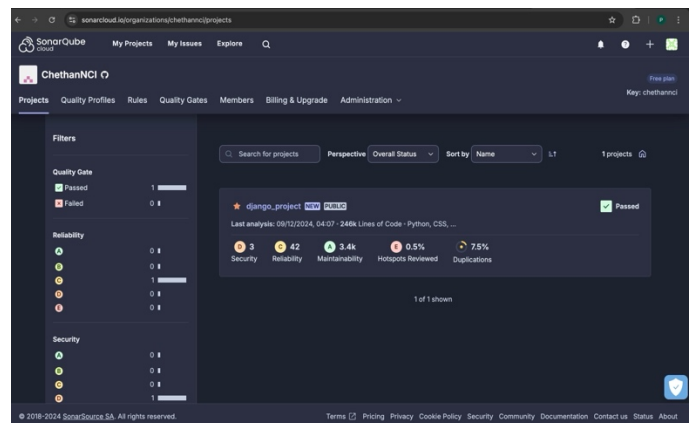


Fig 4.2

V. CONCLUSIONS

Through this Project I learned a lot about building a scalable, secured and feature-rich web app as Job Portal. How to handle authentication, job listings and admin / seamless user experience using Python Django. I also looked into how HTML and Bootstrap can be used to make a responsive, user-friendly frontend. We learned how to set up logging and monitoring for AWS services, which was particularly important for ensuring the health of our application. Monitoring services allowed us to capture logs for further analysis, while tools like Cloud Watch enabled us to get a view into the application. I realize that CI/CD is the arrival of a lighting drilled process across Building, Testing and deploying. Even better, versioning on GitLab made collaboration very efficient, which promoted high-quality source code and seamless integration and delivery of features. With Sonar Cloud integration for static code analysis we had another layer of assurance around secure coding and high code quality. By identifying and providing the means to address key vulnerabilities through the whole process, the integration has ensure better coding practices, including maintaining the product's life and security over time.

In future if I get a chance to implement this project again I would use docker to containerize the application which helps us application being platform independent. Seek information about job applications, interview timing and status updates through live email or SMS alerts. Push notifications help you stay mobile-friendly. container orchestration with Kubernetes to scale and high available the application. Tools such as Prometheus, Grafana or AWS CloudWatch can be integrated to monitor application performance and provide alerts. Design infrastructure for fail-safe rollback to a stable state in the event of a deployment failure.

References

- [Github, "Github," [Online]. Available: 1 <https://docs.github.com/en/authentication/connecting-to-github-with-ssh/adding-a-new-ssh-key-to-your-github-account?tool=cli&platform=linux>. [Accessed 2024 6 December].
- [Github, "Github," [Online]. Available: 2 <https://docs.github.com/en/migrations/importing-source-code/using-the-command-line-to-import-source-code/adding-locally-hosted-code-to-github>. [Accessed 6 December 2024].
- [AWS, "Amazon," AWS, [Online]. Available: 3 <https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/eb3-cmd-commands.html>. [Accessed 6 December 2024].
- [[Online]. Available: <https://medium.com/@alvinxrw/code-quality-static-code-analysis-with-sonarcloud-fcc51311ab71#:~:text=Let's%20set%20it%20up%20on%20sonarcloud&text=Configure%20the%20sonar%2Dproject,paath%20to%20the%20coverage%20report.&text=Execute%20the%20SonarScanner%20co>. [Accessed 7 December 2024].
- [A. Austin. [Online]. Available: 5 <https://medium.com/@alvaro.austin/next-level-testing-harnessing-jest-with-next-js-for-superior-quality-on-sonarcloud-b02ed3cd7612>. [Accessed 7 December 2024].
- ["Sonarqube," [Online]. Available: 6 <https://docs.sonarsource.com/sonarqube-cloud/getting-started/github/>. [Accessed 7 December 2024].