Deployment 4 (Jenkins and AWS Elastic Beanstalk)

What is the Purpose of this Project?

This deployment consists of 3 separate Flask Web applications. The first app is a "URL Shortener" which was given to us in class. The second application is a To Do List Flask App. The third is a Blog Site made with Flask as well. All 3 of these projects were successfully deployed using Jenkins with Elastic Beanstalk in AWS.

Process Documentation:

1. **Flask Application:** Your first major goal (same for all 3 projects) is to ensure your Flask Application is running successfully on a local server.
    a. Use Flask to run your project locally
    b. Connect your local project folder to a Github Repository

2. **Elastic Beanstalk:** The next major goal is to set up Elastic Beanstalk in AWS
    a. Navigate to Elastic Beanstalk as an IAM user in your AWS account.
    b. Create a new "application" in the Beanstalk service.
    c. Beanstalk will have you create a new "environment" along with the application
        i. You should select "web server application" when selecting your "environment tier", this will result in a boiler plate "app" being set up for you in AWS. Beanstalk will provide you with a url to access the web "app"
        ii. Take note of your "application name" and "environment name" in the beanstalk service

3. **Jenkins:** Configuring a Jenkins Item to work with Github and Elastic Beanstalk
    a. A prerequisite for this is to have an AWS EC2 instance running with Jenkins installed
    b. Once you log into your Jenkins instance, you should download 2 plugins:
        i. AWSEB Plugin, this will allow you to have a post build step in your deployment process that sends your python application to AWS Elastic Beanstalk for deployment
        ii. Cloudbees Credentials Plugin, will let us use credentials as an extension to our deployment process and allow us to use environment variables
    c. Create a Freestyle Jenkins Item
        i. Select Git as your source manager

ii. Use AWS credentials to validate your credentials and make sure you select the correct region.
iii. Very Important: Make sure your application name and environment name in Jenkins match whats in AWS EB.
iv. The root object will need a single . as a path

4. Important piece to the puzzle: AWS looks for a particular application name when deploying a python app to Elastic Beanstalk. The following naming conventions were needed in our Flask App for the deployment to succeed

```
18
19  application = Flask(__name__)
20
```

```
95  if __name__ == "__main__":
96      application.run()
```

Links Are Below

ToDo App Repo: https://github.com/Fordonez20/ToDoApp-Flask

Blog Flask App Repo:https://github.com/Fordonez20/SimpleBlog-FlaskApp