# WAVE - DevOps Final Project

Guidelines:

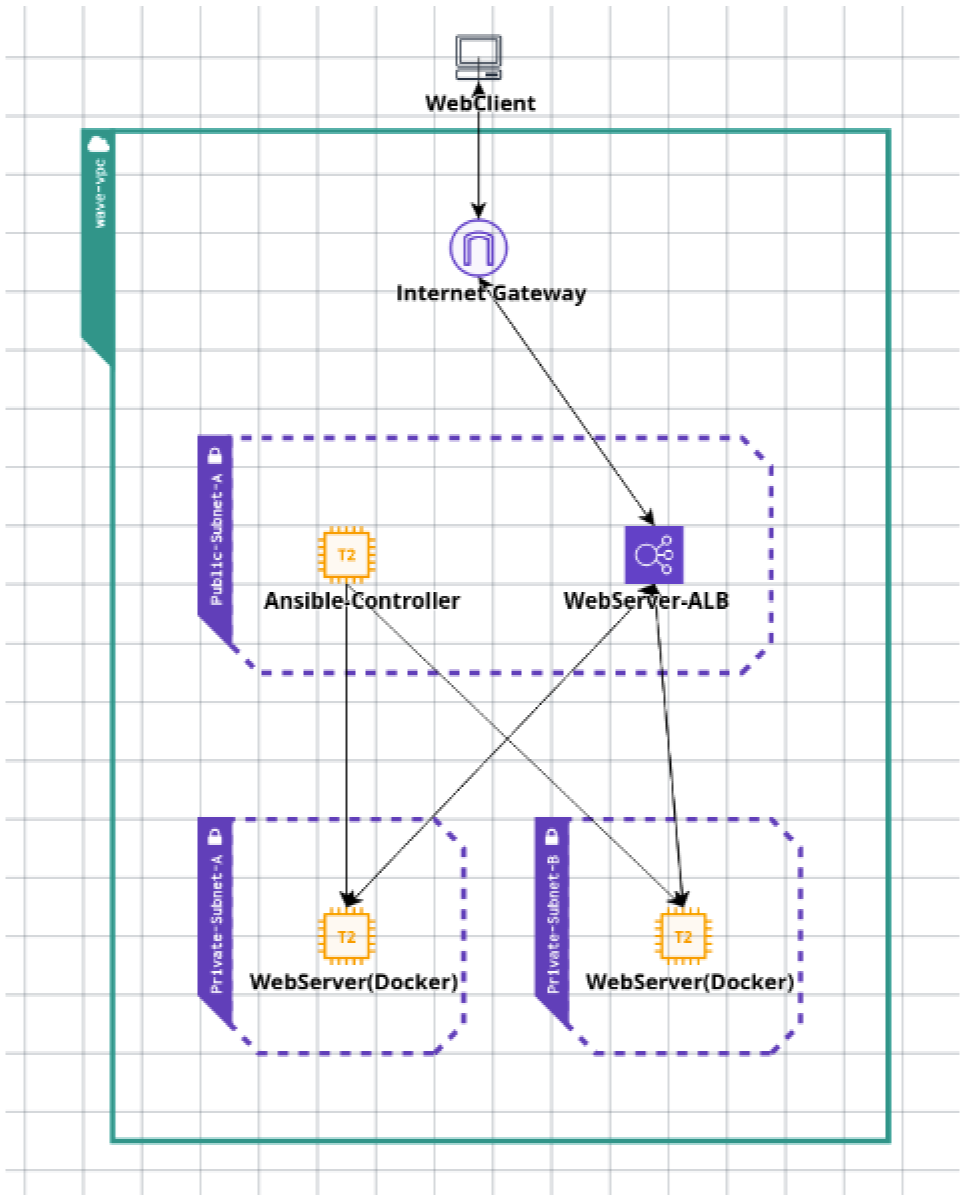
- All the resources will be created in us-east-1 (N. Virginia) region
- All IaaC will be hosted in your github and be taken as major part of the solution and examination mark
- Functionality of your environment is important for the final results
- Store your work in your Git Repository
- To automate infrastructure deployment use Terraform or other preferred Iaac Tool of your choice
- To automate configuration and run tasks you can use Ansible or CI/CD tool such as Jenkins, you are also free to choose any open source automation tool of your choice and explain why you choose each tool

**You have to solve all these problems using scripting and automation.**
**This is not a step-by-step exam, you are free to choose the course of your solution as long as it aligns with the requirements.**
**Whichever tools you use, you should be able to destroy your infrastructure and rebuild it quickly by running your automation.**
**For each of your solutions, include a README.md file in your repository with working steps to follow to deploy your solution.**
**To demonstrate your working environment for each step, record a short demo video for each part.**

**All IaaC and configuration should be existing in your repository.**

**TIP:** You can always shut down your instances or scale your ASG to zero in order to reduce cost and get back to work later

## Part I - Website



**Objectives**:

1. Deploy a static website running in a docker container on an ec2 instance in a private subnet (no access from the web)

2. Making the website accessible from the external web by ALB on ports 80 and 443
3. The static website should be highly available (multi-AZ)
4. Static website server should be configured using ansible playbooks
5. AWS Infrastructure components should be created using terraform

By the end of this section you should have a static website running in a docker container on two private ec2 instances in multi AZ. The website should be accessible using the ALB.
Remember to create the necessary SG rules so that you can access the website.

**You answer should include**:
1. Your Git repository with the infrastructure and deployment automation code including a step-by-step README.md that explains how to deploy your infrastructure
2. A demonstration of a working environment (Recorded Demo)

## Part 2 - EKS

**Objectives**:
1. Create an EKS cluster in a dedicated VPC, the configuration should include:
   (You can choose the tool that you prefer, eksctl/cloudformation/terraform)
   a. Dedicated VPC
   b. 2 public subnets
   c. 2 private subnets
   d. 2 availability zones (Public & Private in each az)
   e. Use 100% spot instances
2. Infra Autoscaling - Worker nodes should be automatically added and removed from your nodegroup when required. **Up to 10 nodes**. You should choose the appropriate type of instances for the same.

**You answer should include**:
1. Overview of your solution (Architecture diagram can be included)
2. List of tools and technologies you used and why did you choose them
3. Your Git repository with the infrastructure and deployment automation code including a step-by-step README.md that explains how to deploy your infrastructure
4. A demonstration of a working environment (Recorded Demo)

## Part 3 - Deploy application on EKS

**Objectives**

1. Look for any application that includes frontend and backend. You can choose any popular GitHub application to dockerize (fronted & backend) or just a demo/sample application in any other source
2. Create ECR for your application images
3. Create a pipeline with your preferred open source CI/CD tool that builds a container image for the application and pushes it to your ECR
4. Create a pipeline with your preferred open source CI/CD tool that deploys your application images to the EKS cluster
5. Make your application accessible using the ingress of your choice

**Notes for part 3**

1. You can use any custom application that you find that consists of at least two microservices, it can also be your own application.
2. You should use only Open Source tools and not cloud proprietary CI/CD tools

**You answer should include**:

1. Overview of your solution. (Architecture diagram can be included)
2. List of tools and technologies you used and why did you choose them
3. Your Git repository with the infrastructure and deployment automation code including a step-by-step README.md that explains how to deploy your infrastructure
4. The pipelines that you created in this Part should be saved in your repository
5. A demonstration of a working environment (Recorded Demo)