# Deploy and Host a website to AWS using EC2 and Docker

We want to create an EC2 instance in AWS and, deploy and host a simple website using docker images.

## Plan skeleton

1. Create a simple node js API to return products array to UI.
2. Create a simple react js UI application to display products and details.
3. Dockerize the code.
4. Create and IAM role for accessing the S3 bucket from EC2.
5. Store data(products to be displayed) in S3 bucket.
6. Upload docker image to Amazon ECR(Elastic Container Repository).
7. Create EC2 instance.
8. Create and Associate Elastic IP address to the instance.
9. Connect to the EC2 instance.
10. Install Docker.
11. Pull docker image from Amazon ECR.
12. Run the Docker container on the EC2 instance.

## Availability/Pricing of services to be used

1. **ECR** 🡪 Free Tier available in AWS.
2. **EC2** 🡪 Free Tier available in AWS.
3. **S3 🡪** Free Tier available in AWS.
4. **IAM 🡪** Free Tier available in AWS.
5. **Docker 🡪** Free Community version available.

## Analysis

1. **ECR** 🡪
   1. [Amazon Elastic Container Registry service.](https://docs.aws.amazon.com/ecr/)
   2. Amazon Elastic Container Registry (Amazon ECR) is a fully managed container registry that makes it easy to store, share, and deploy your container software anywhere. When you use Amazon ECR, your images and artifacts remain highly available and reliably deployable.
2. **EC2** 🡪
   1. [Elastic Compute Cloud (ECC)](https://docs.aws.amazon.com/ec2/?nc2=h_ql_doc_ec2)
   2. It is a cloud computing service offered by the Cloud Service Provider AWS. You can deploy your applications in EC2 servers without any worrying about the underlying infrastructure
3. **S3 🡪** 
   1. [Amazon Simple Storage Service](https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html).
   2. It is an object storage service that offers industry-leading scalability, data availability, security, and performance. It provides management features so that you can optimize, organize, and configure access to your data to meet your specific business, organizational, and compliance requirements.
4. **IAM 🡪** 
   1. [Amazon Identity and Access Management](https://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html).
   2. Securely manage identities and access to AWS services and resources
5. **Docker 🡪** 
   1. [Docker](https://aws.amazon.com/docker/) is a software platform that allows you to build, test, and deploy applications quickly. Docker packages software into standardized units called containers that have everything the software needs to run including libraries, system tools, code, and runtime. Using Docker, you can quickly deploy and scale applications into any environment and know your code will run.

## Solution Plan

1. Create a simple node js API to return products array to UI.
   * Create data file in json format for product details.
   * Create a node js project in local system.
   * Write API code to fetch data from a specified path and return the product details array.
     + If data file exists,
       - Then, get data from the file
     + Else,
       - use a public API to get the date.
2. Create a simple react js UI application to display products and details.
   * Create a react js project in local system.
   * Use Boostrap to make the UI components and make it responsive.
   * Integrate the API to display product details.
3. Dockerize the code.
   * Down docker community version in local system.
   * Create docker account.
   * Create docker images for the UI and API projects.
   * Store the images in local.
   * Run the docker containers in local to test.
4. Store data (products to be displayed) in S3 bucket.
   * Create S3 bucket in AWS.
   * Upload the data files to S3 bucket.
5. Create and IAM role for accessing the S3 bucket from EC2.
   * Create IAM role in AWS.
   * Add the IAMReadOnlyAccess permission to it for S3 bucket.
6. Upload docker image to Amazon ECR (Elastic Container Repository).
   * Create account.
   * Create a repo in ECR using Aws console
   * Login to ECR from local cmd.
   * Tag docker images to upload.
   * Push Image to ECR.
7. Create EC2 instance.
   * Create EC2 instance with needed configuration for Free tier.
   * Assign the IAM role with permission to access the S3 bucket.
   * Copy the data from S3 bucket.
8. Create and Associate Elastic IP address to the instance.
   * Create and associate an Elastic IP to the EC2 instance.
     + Note: Don’t keep any created IP un-associated. Delete the IP when logging off as it will start billing.
9. Connect to the EC2 instance.
   * Login to aws from local cmd and connect to the EC2 instance created.
10. Install Docker.
    * Download and install Docker into the EC2 instance.
11. Pull docker image from Amazon ECR.
    * In the EC2 instance, Connect to Amazon ECR and download the docker images.
12. Run the Docker container on the EC2 instance.
    * Run and test the Application.

## Timeline Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Task | Details | Estimated duration | Comment |
| 1. | Create AWS account and familiarize with it. | * Create AWS account * Browse the interface and familiarize with it. | 2nd Dec |  |
| 2. | Obtain data for application. | * Research and find suitable data from the web. * Create data file. | 2nd Dec |  |
| 3. | Create and write code for UI and API | * Create and write code for API * Both scenarios – fetch data from public API, as well from data file. * Create and write code for UI application * Fetch and Display data | 3rd Dec – 4th Dec |  |
| 4. | Dockerize the code | * Download Docket * Create Docker account * Create Docker images | 5th Dec – 6th Dec |  |
| 5. | Upload images to Amazon ECR. | * Create Repo in Amazon ECR * Login from local system * Upload the Docker Images to the Repo | 9th Dec |  |
| 6. | Create and upload to AWS using S3 | * Create S3 bucket. * Create upload data files to S3 bucket. | 10th Dec |  |
| 7. | Create IAM role to access S3 bucket. | * Create IAM role. * Add read-only access for S3 bucket to the IAM role. | 11th Dev |  |
| 8. | Create EC2 instance | * Create EC2 instance * Associate IAM role with EC2. * Import the data file from S3 to EC2. | 12th Dec – 13th Dec |  |
| 9. | Install required application packaged in EC2 instance. | * Install nodejs, reactjs, Docker in EC2 instance | 16th Dec |  |
| 10. | Import Images from Amazon ECR to EC2 instance. | * Login to Amazon ECR from EC2 instance. * Pull the Docker Images | 17th Dec |  |
| 11. | Run and test the application. | * Run the Docker Containers * Test the Application | 18th – 20th Dec |  |
|  |  |  |  |  |