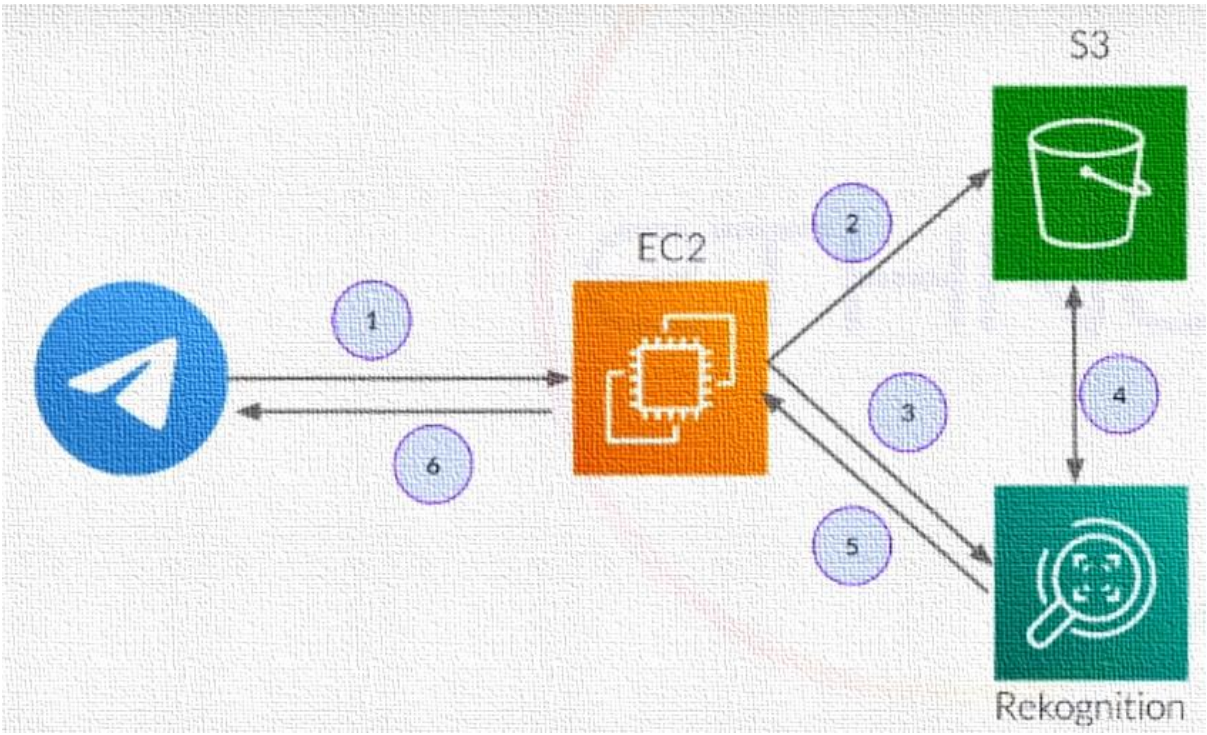
Face Detection using Amazon Web Services

Services Used to build the application:

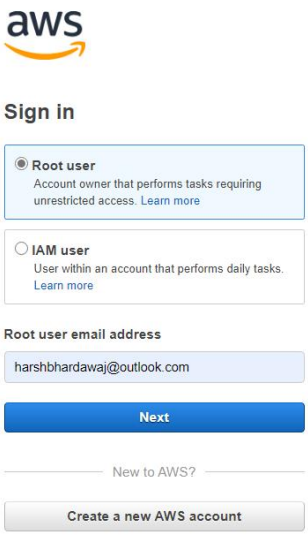
1. EC2 – Elastic Compute Cloud
2. S3 – Simple Storage Service
3. Rekognition
4. Telegram Bot

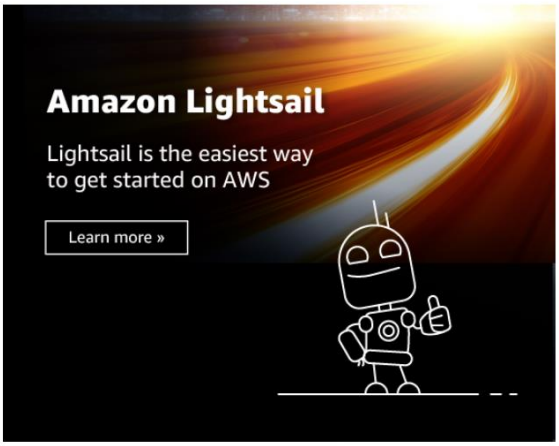
Application Architecture:

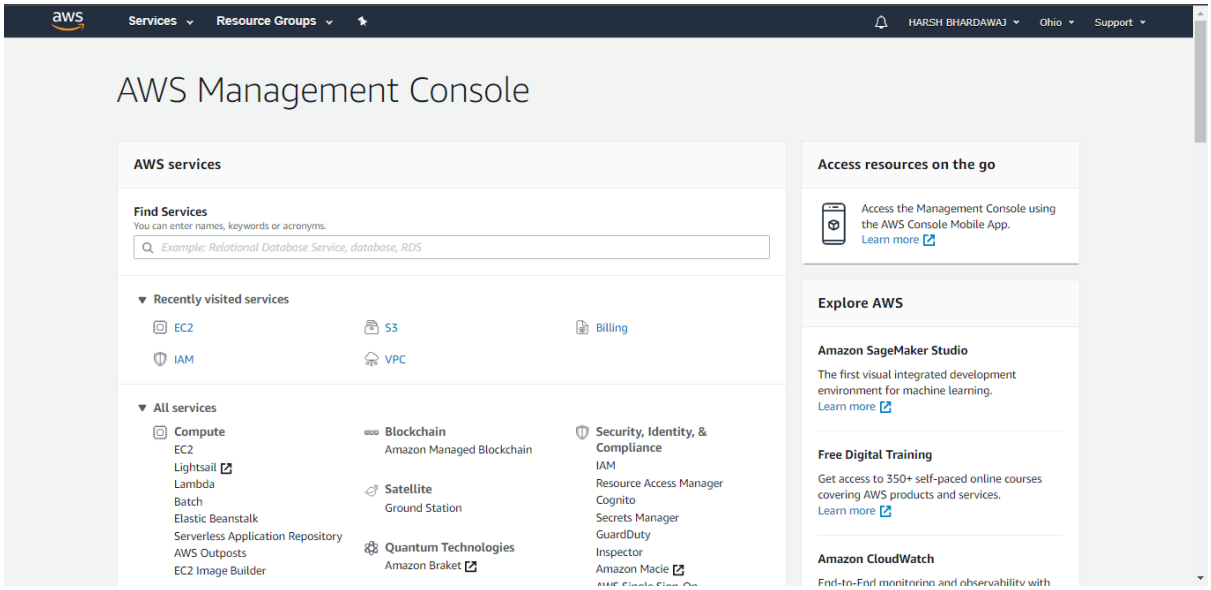


Dashboard Screenshots for all the used Services:

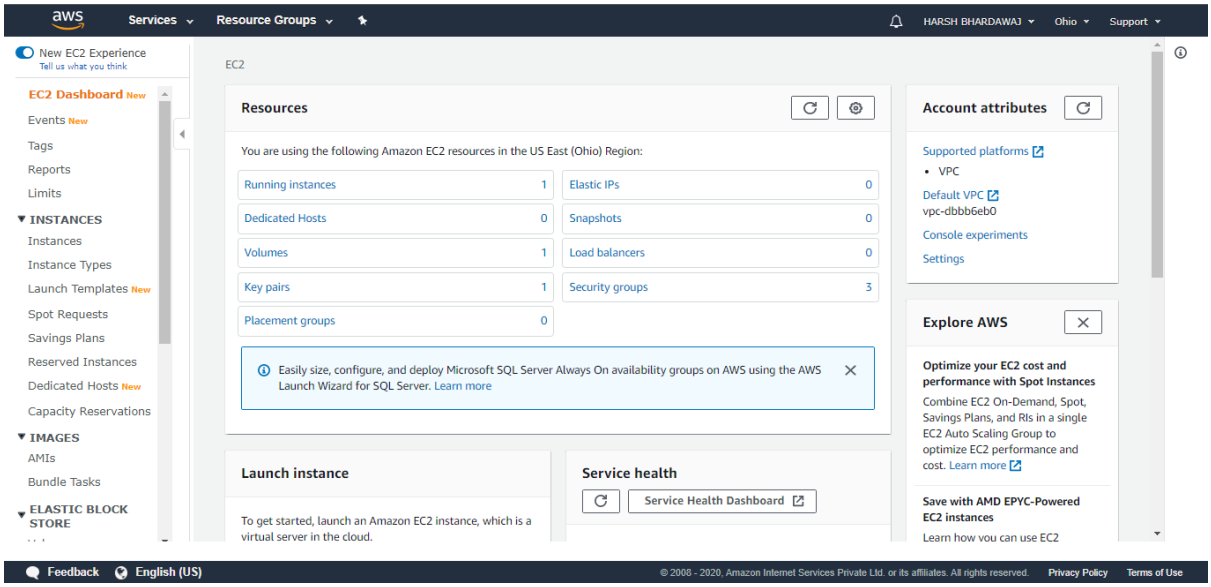
a) AWS Login screen with username:



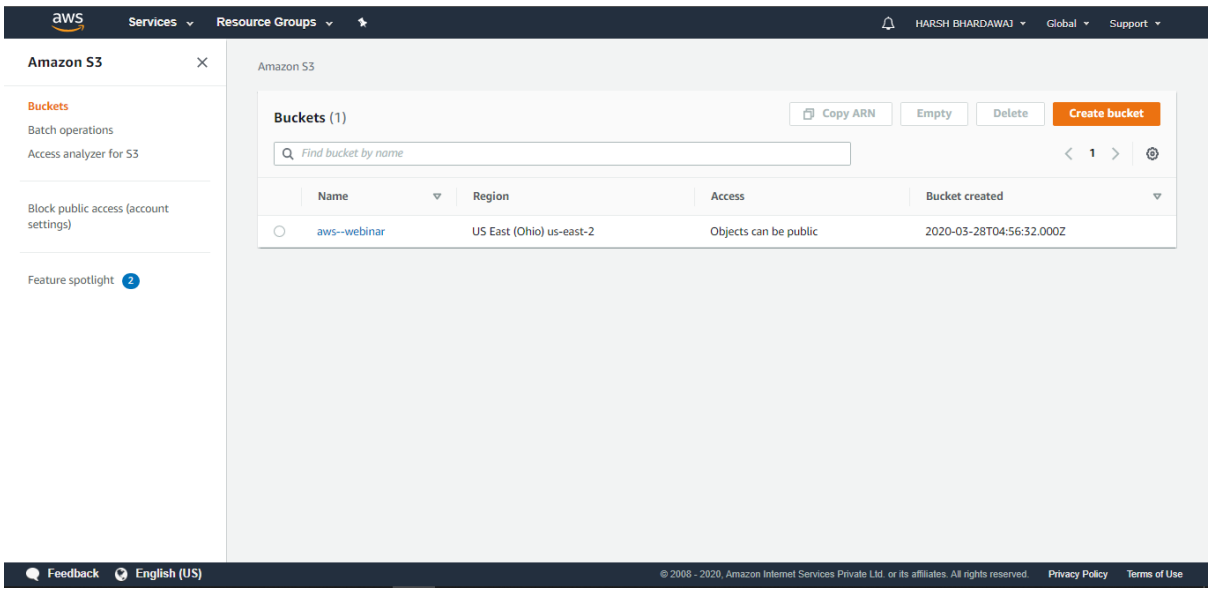




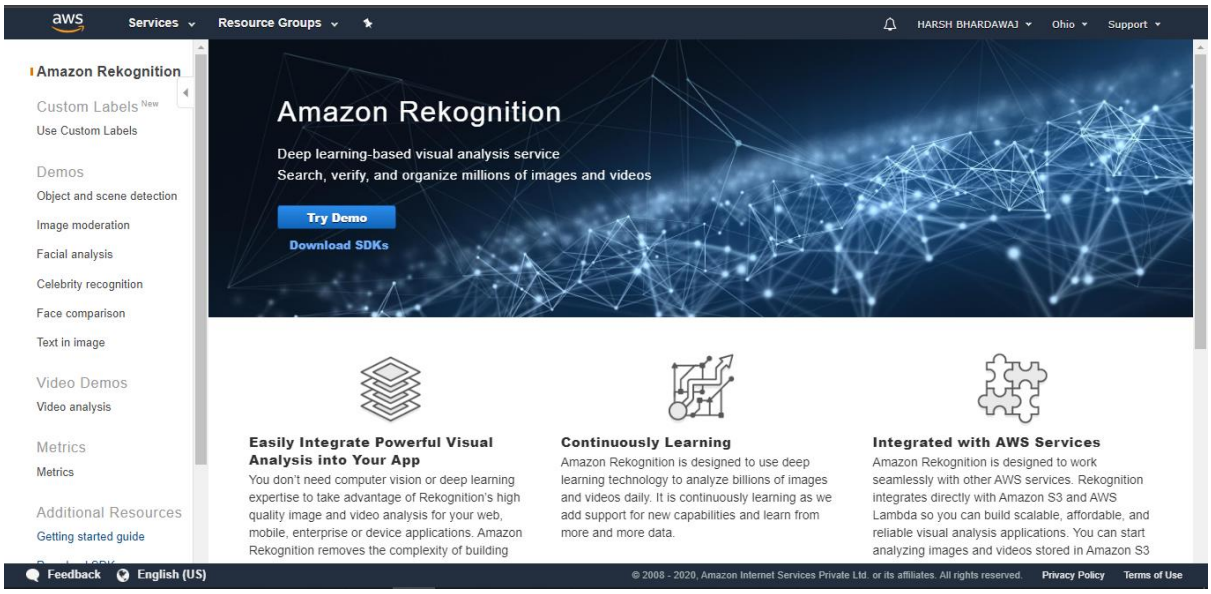
b) EC2 Dashboard:



c) S3 Dashboard:

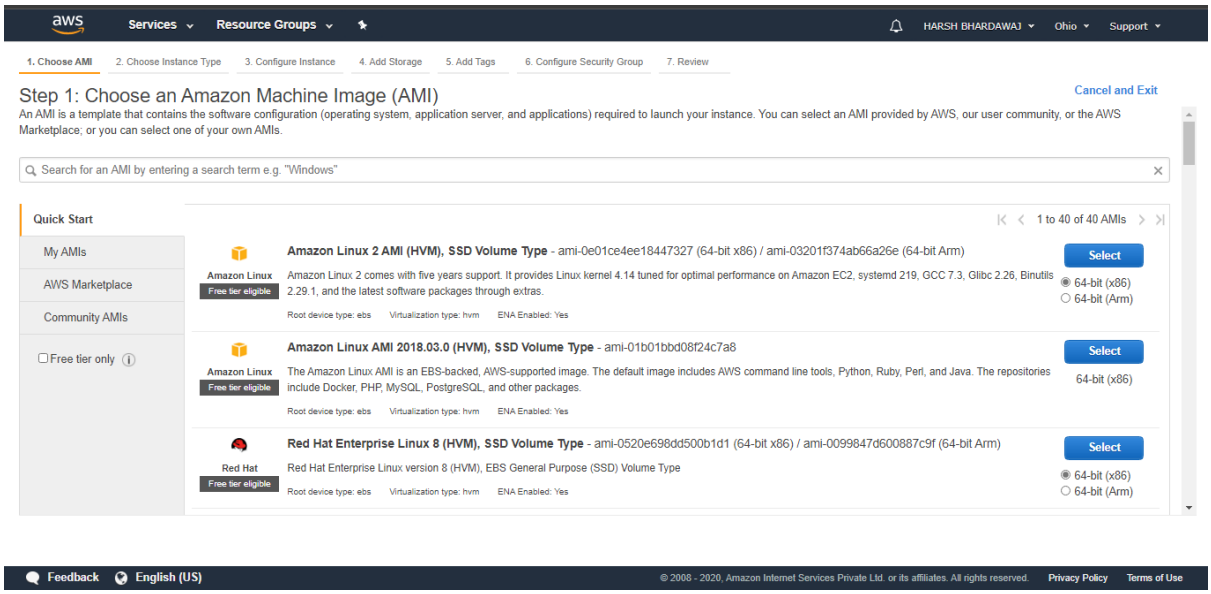


d) Rekognition Dashboard:

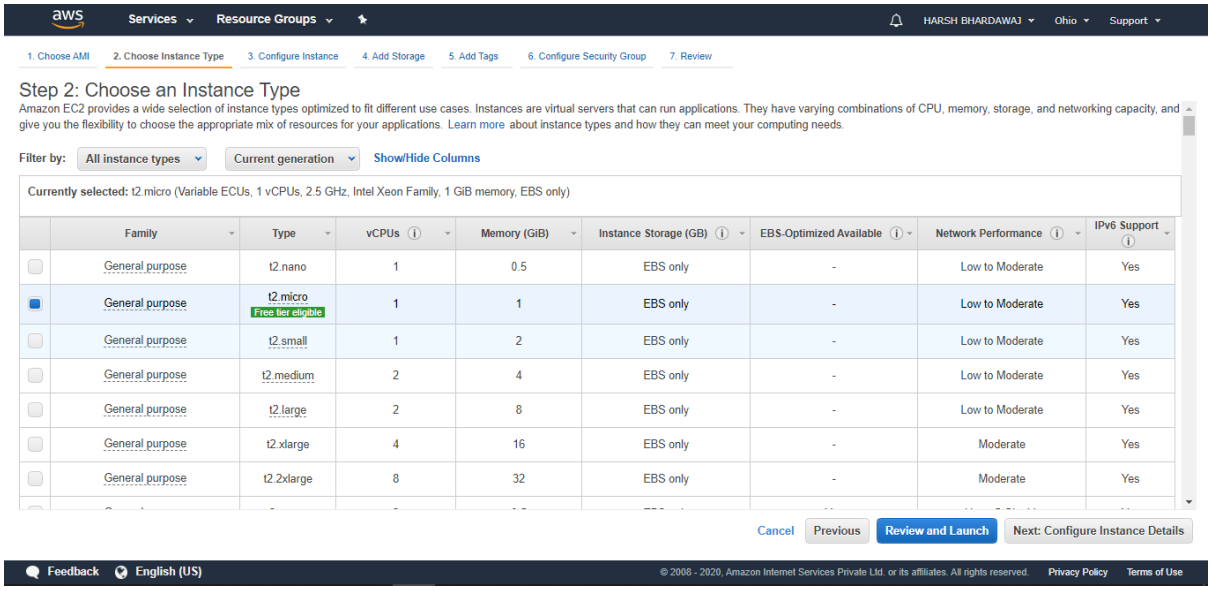


EC2 Configuration Screenshots:

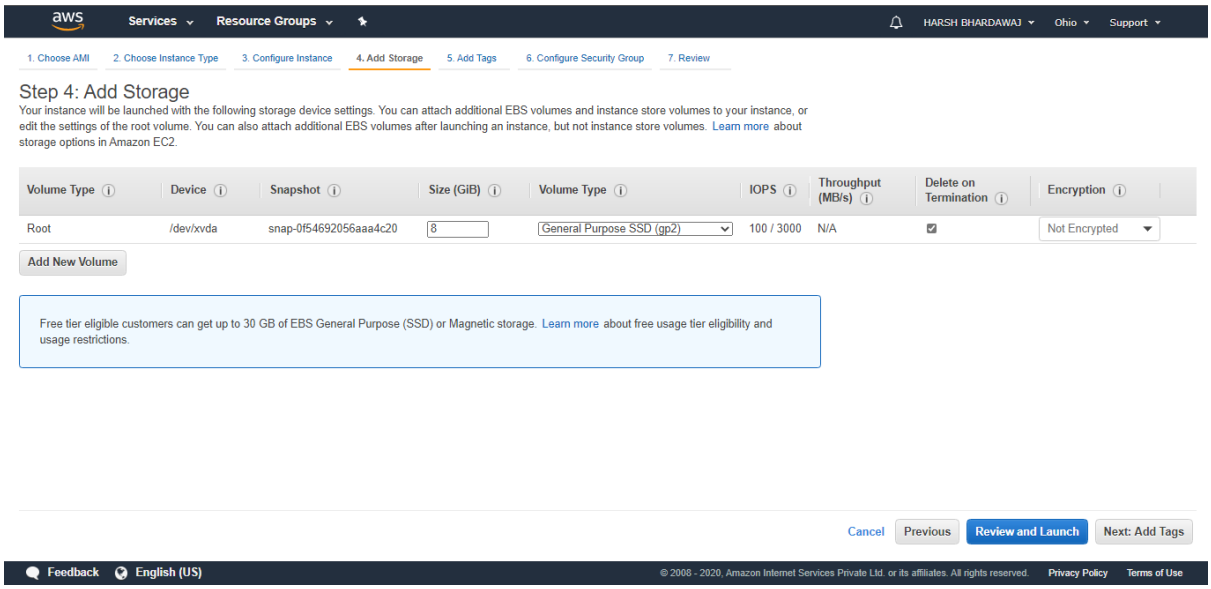
a) Choosing an AMI:



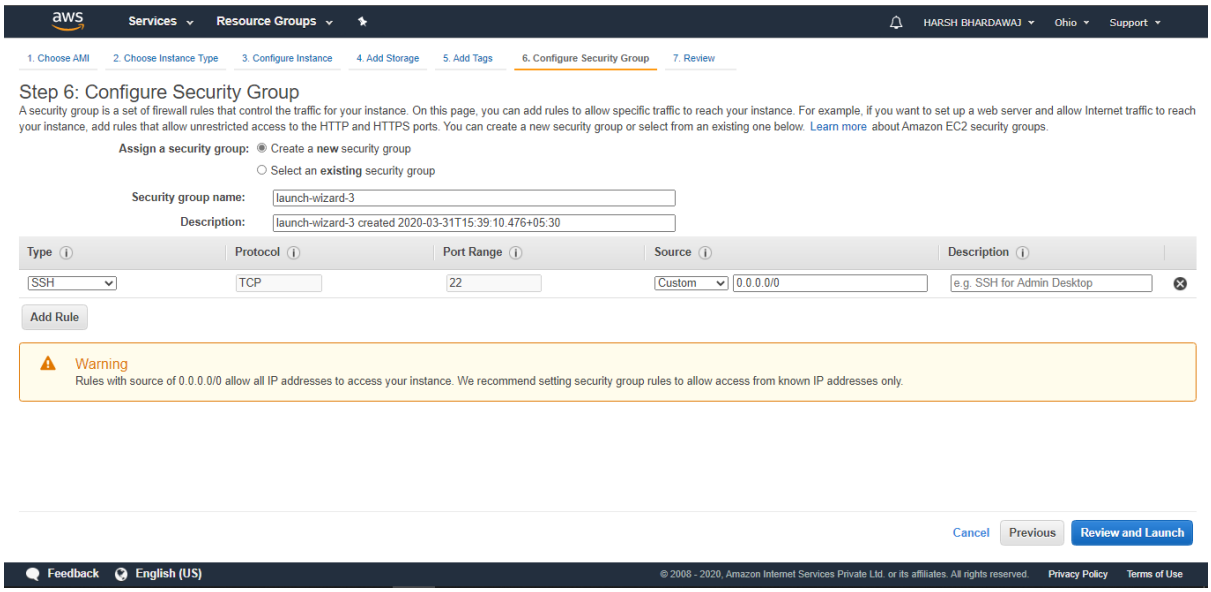
b) Choosing an instance Type:



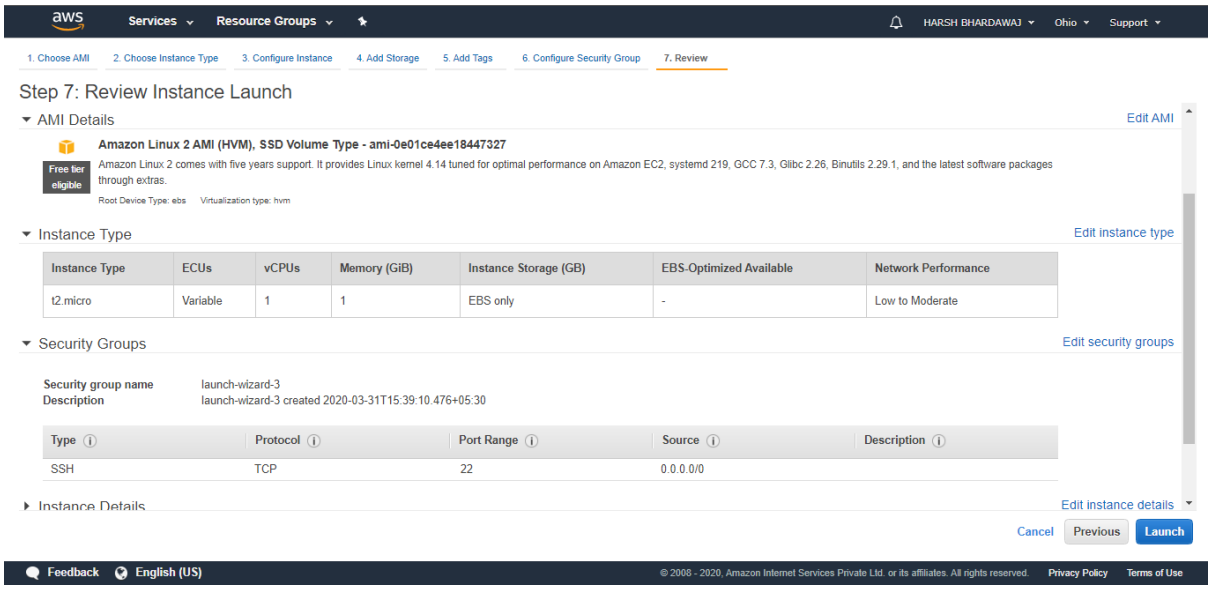
c) Adding Storage:



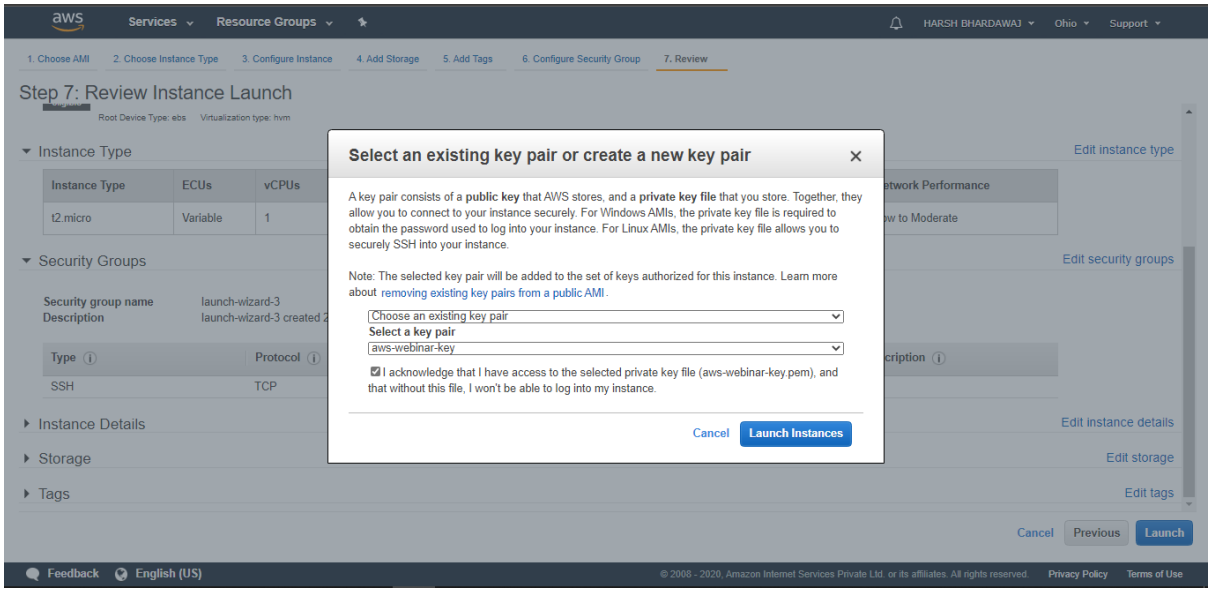
d) Configure Security Group:



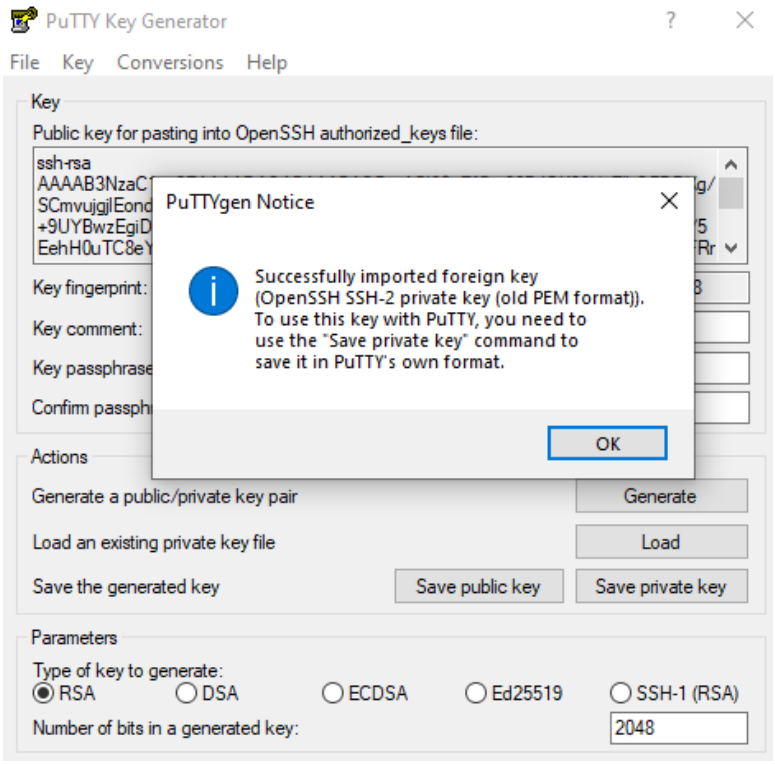
e) Review of Instance Launch:



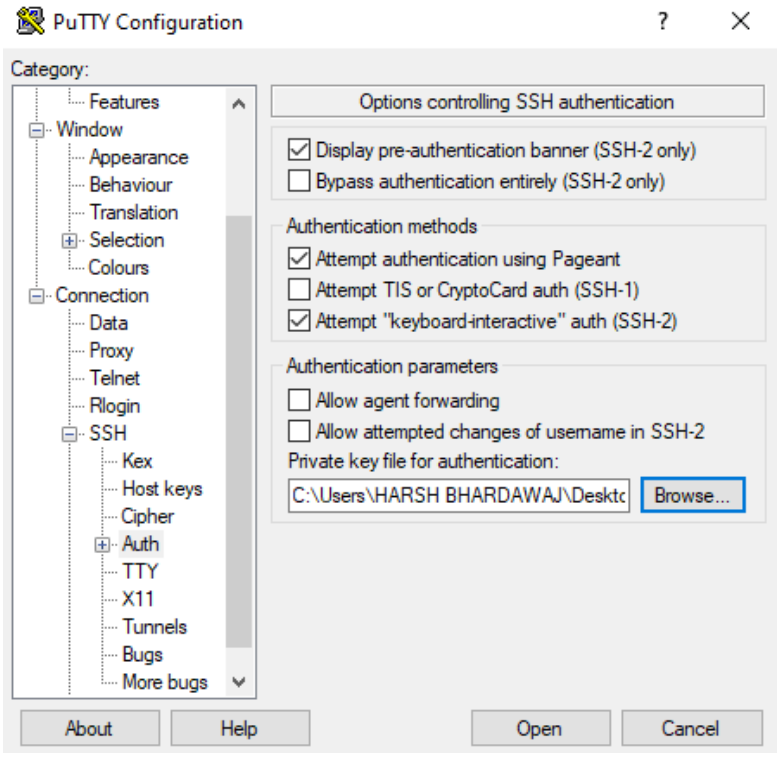
f) Selecting/Downloading a key Pair:



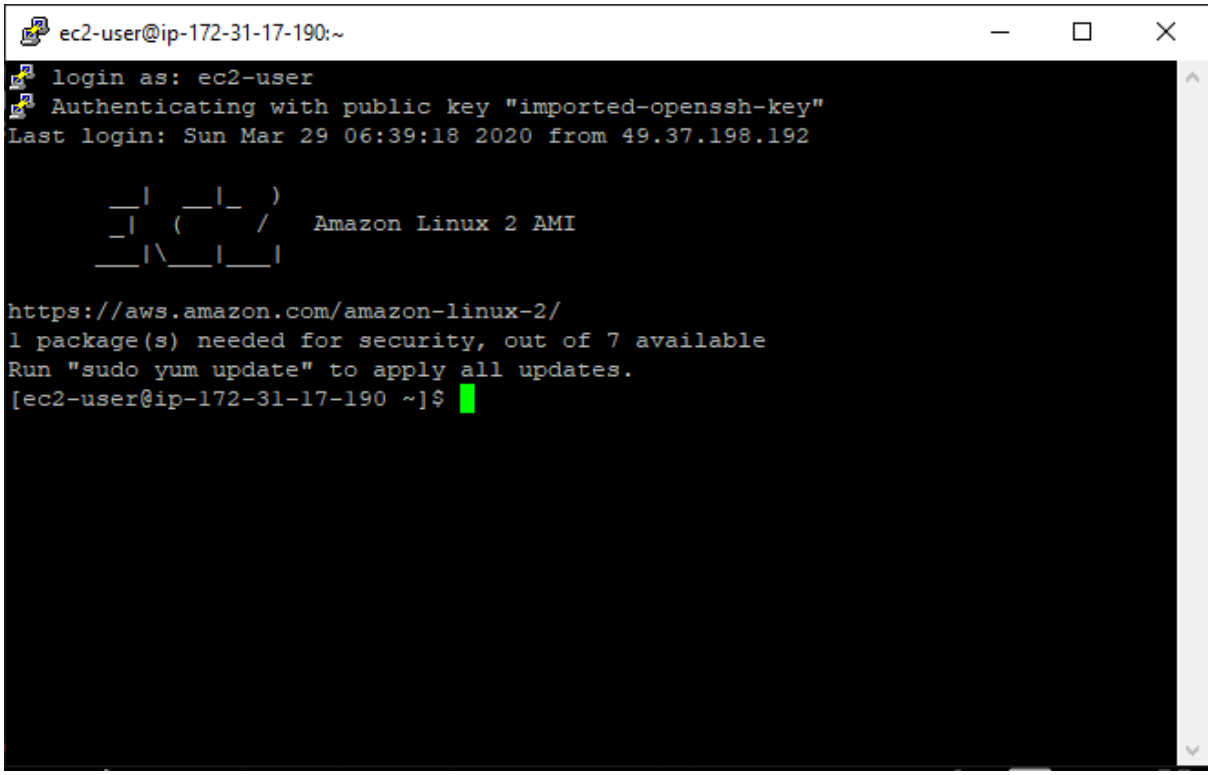
g) Puttygen Conversion of key from pem to ppk format:



h) Remotely Accessing EC2 using private key:

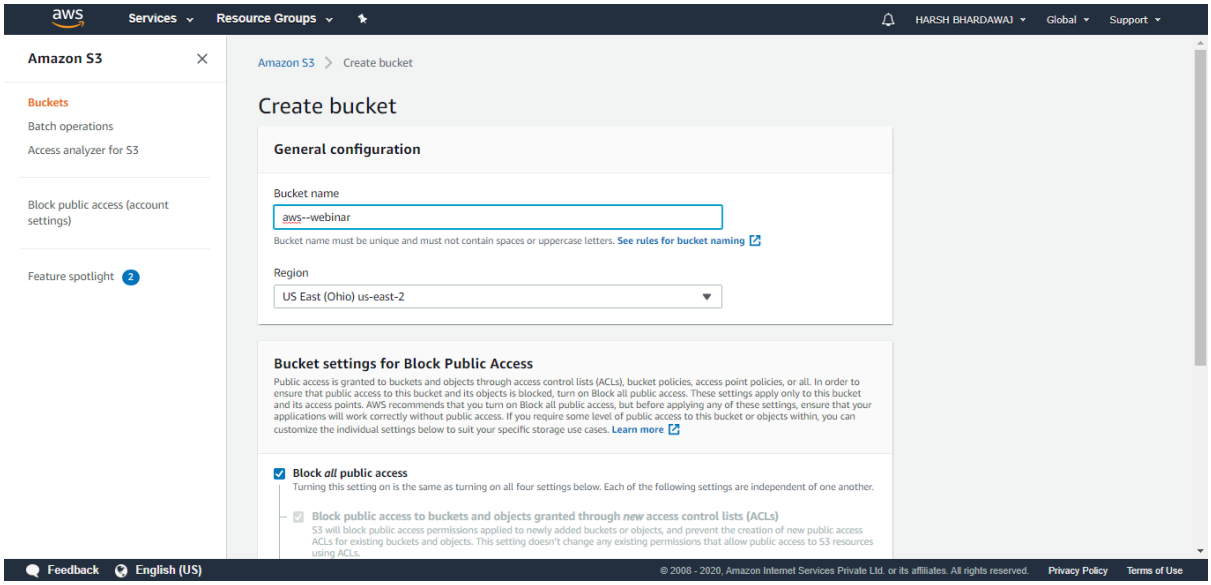


i) Logged in EC2 Screen:

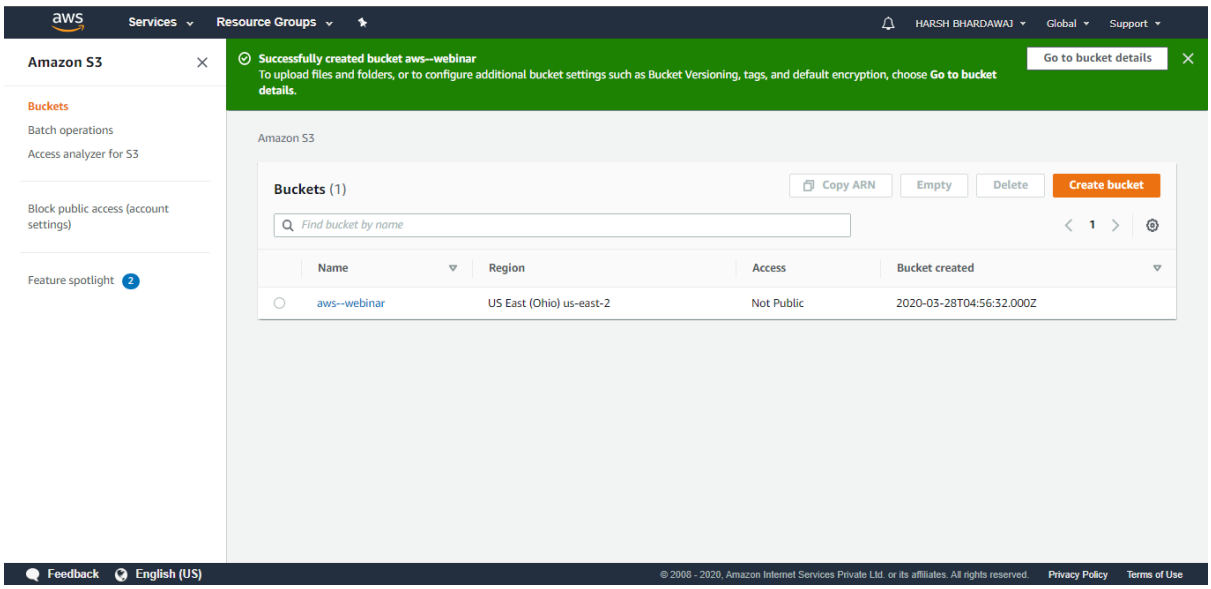


S3 Configuration Screenshots:

a) Creating a Bucket:

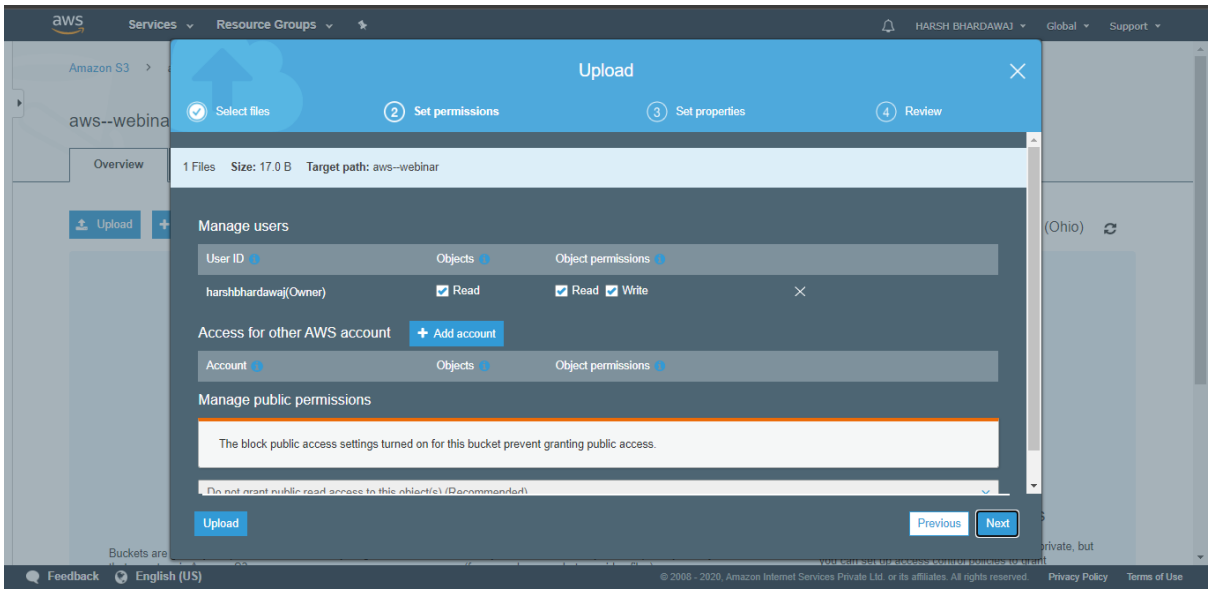


b) Successful Bucket Creation:

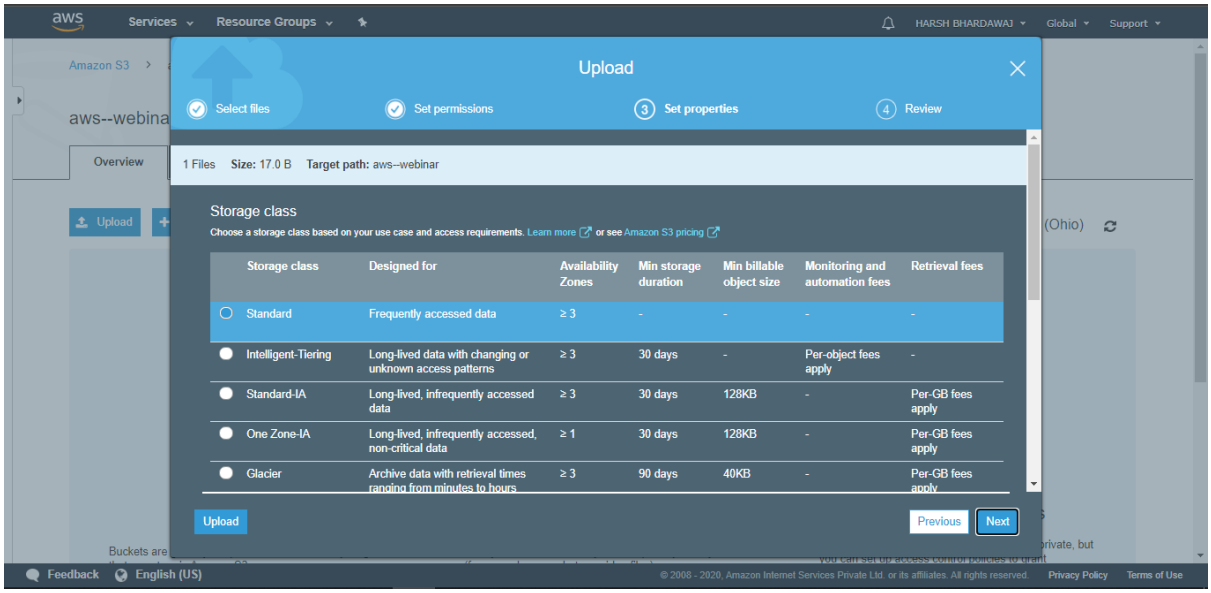


c) Uploading an object:

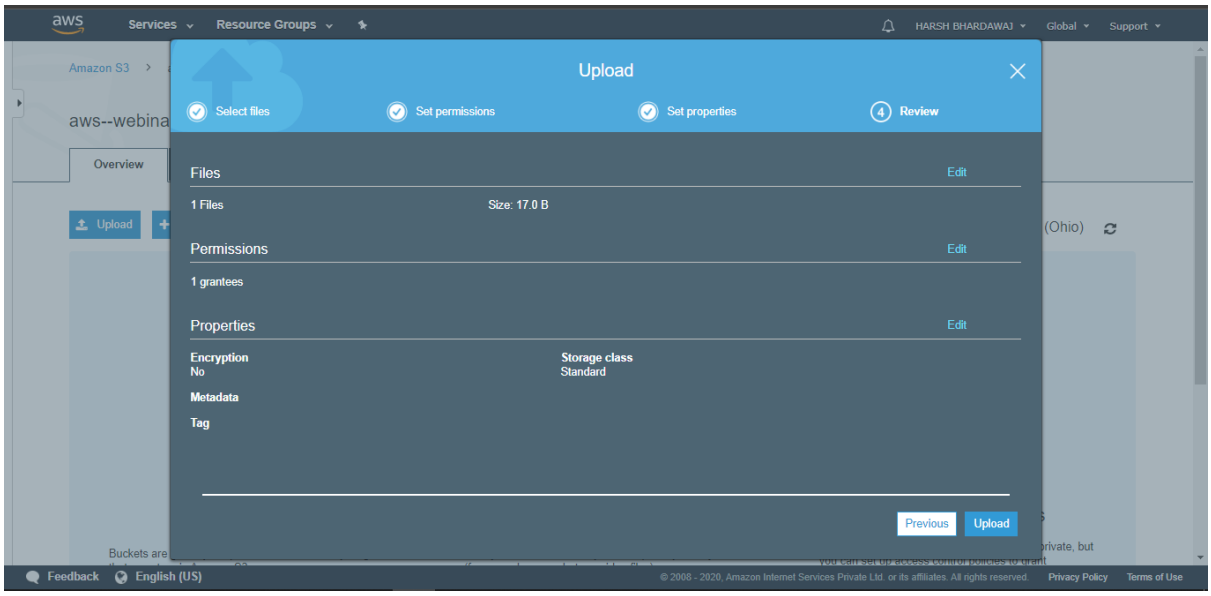
Step i: Selecting a file:



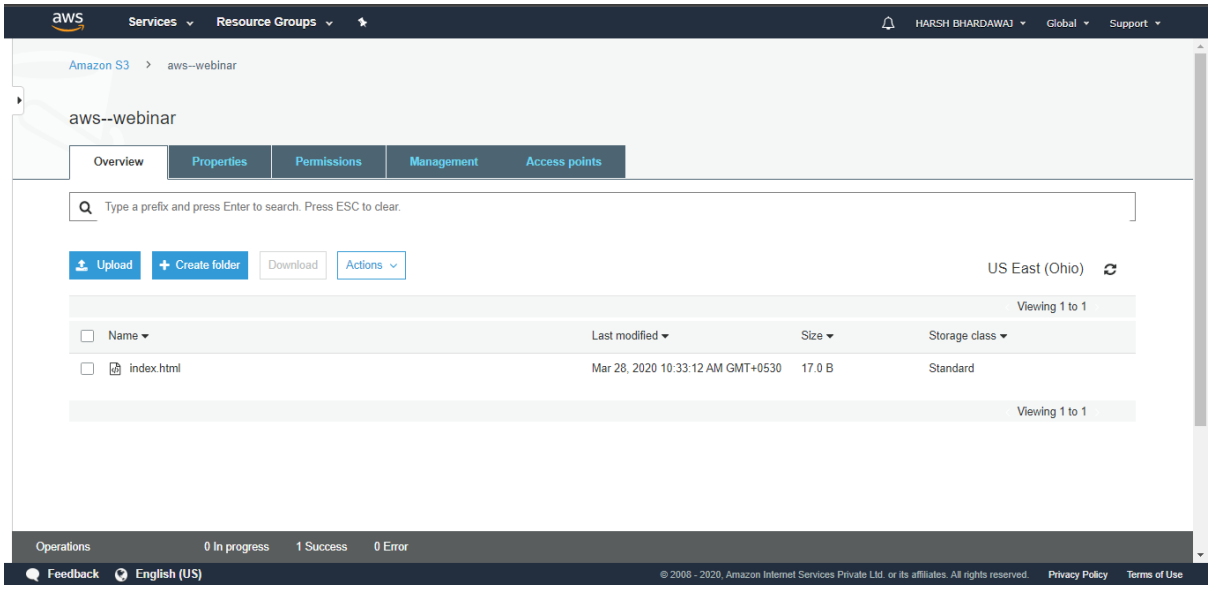
Step ii: Selecting a storage class:



Step iii: Review of upload:

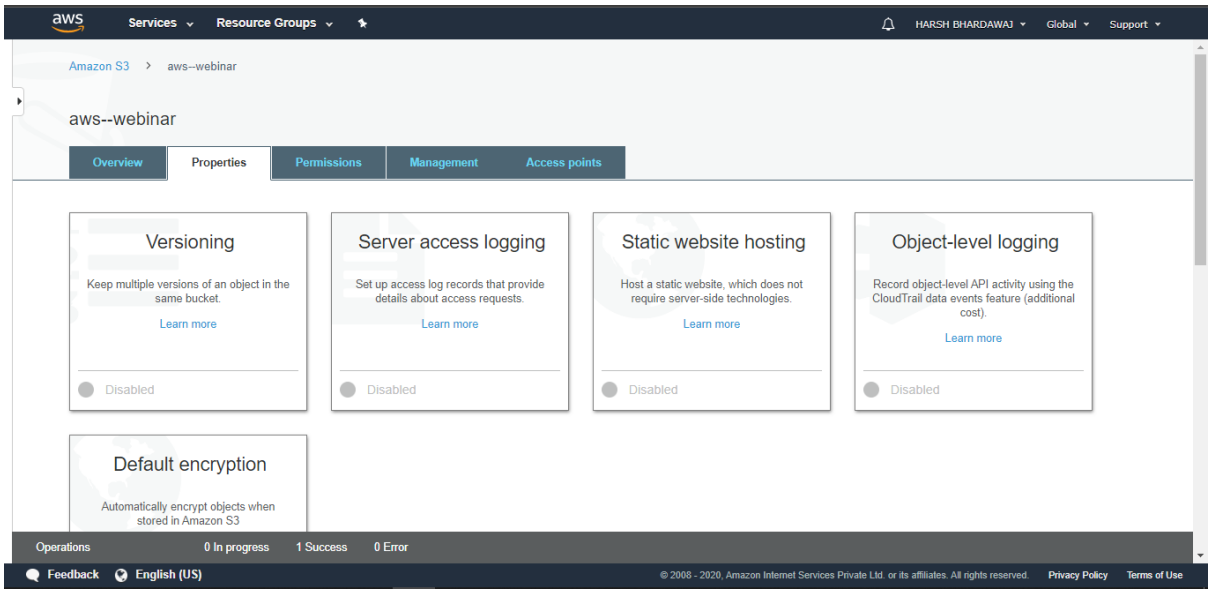


Step iv: After Uploading the Object:

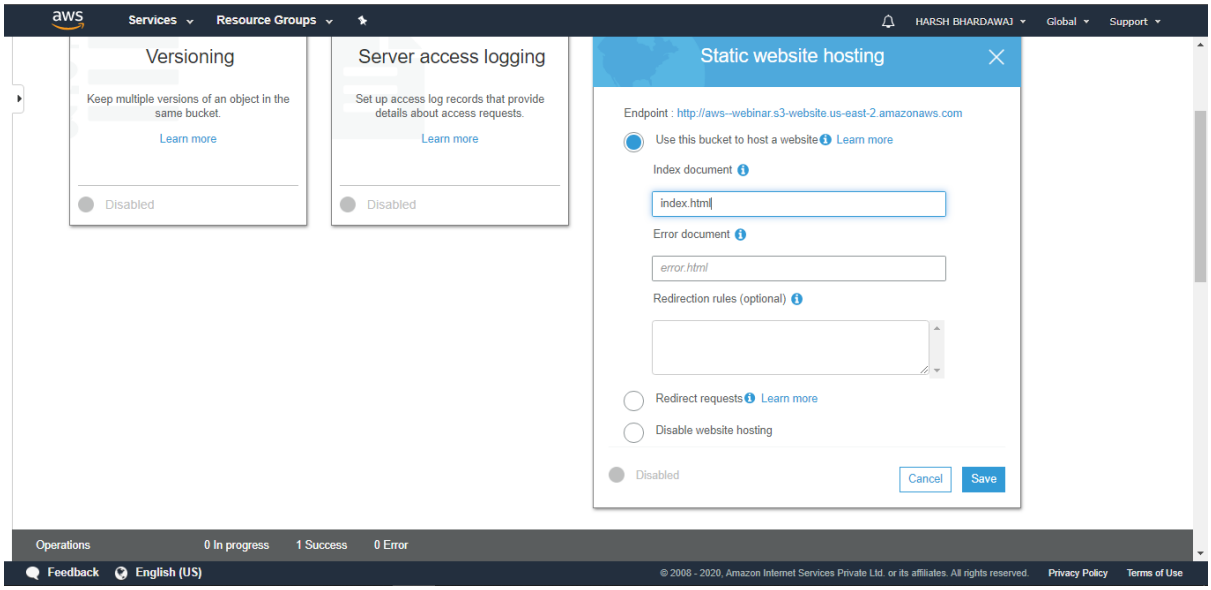


d) Enabling Static Website:

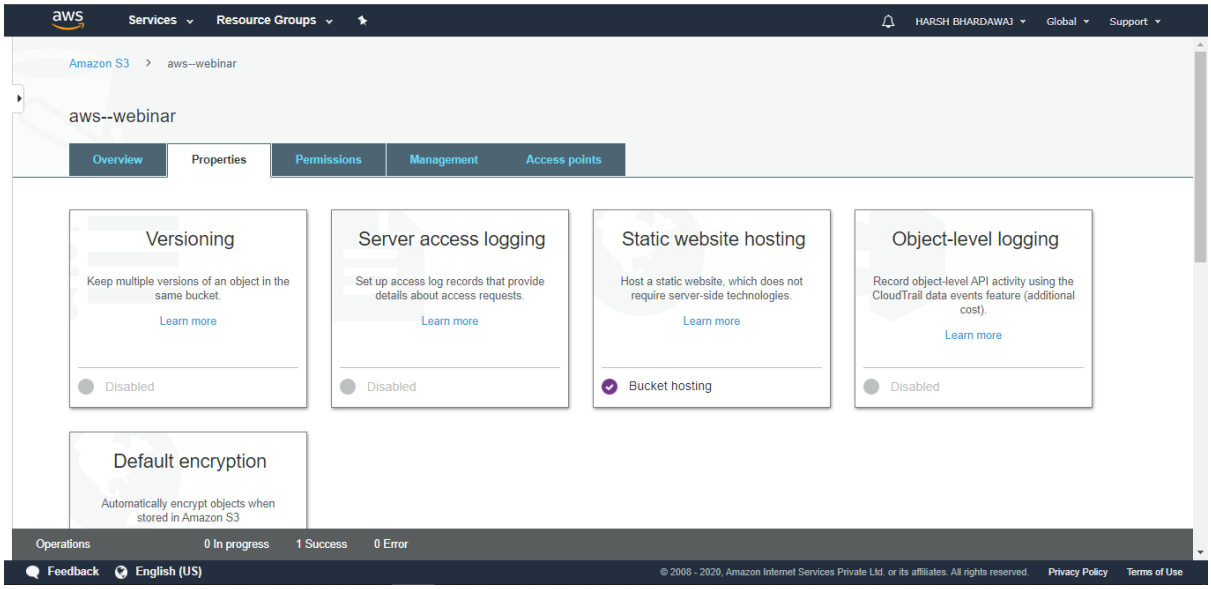
Step i: Properties of bucket created:



Step ii: Static website hosting:

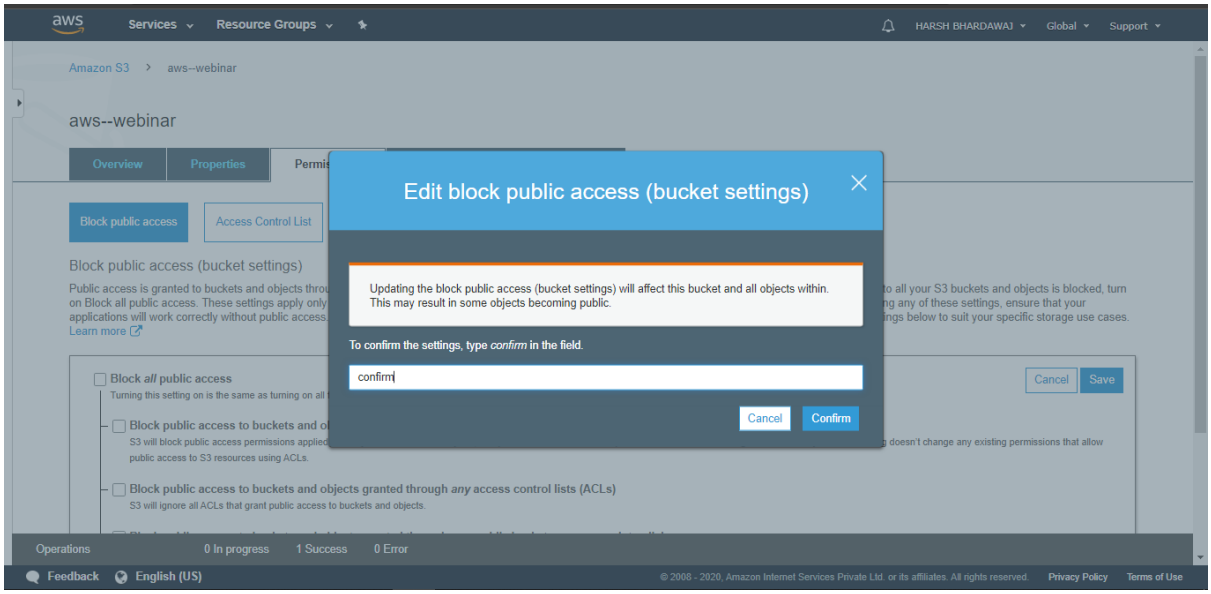


Step iii: Enabled Bucket hosting:

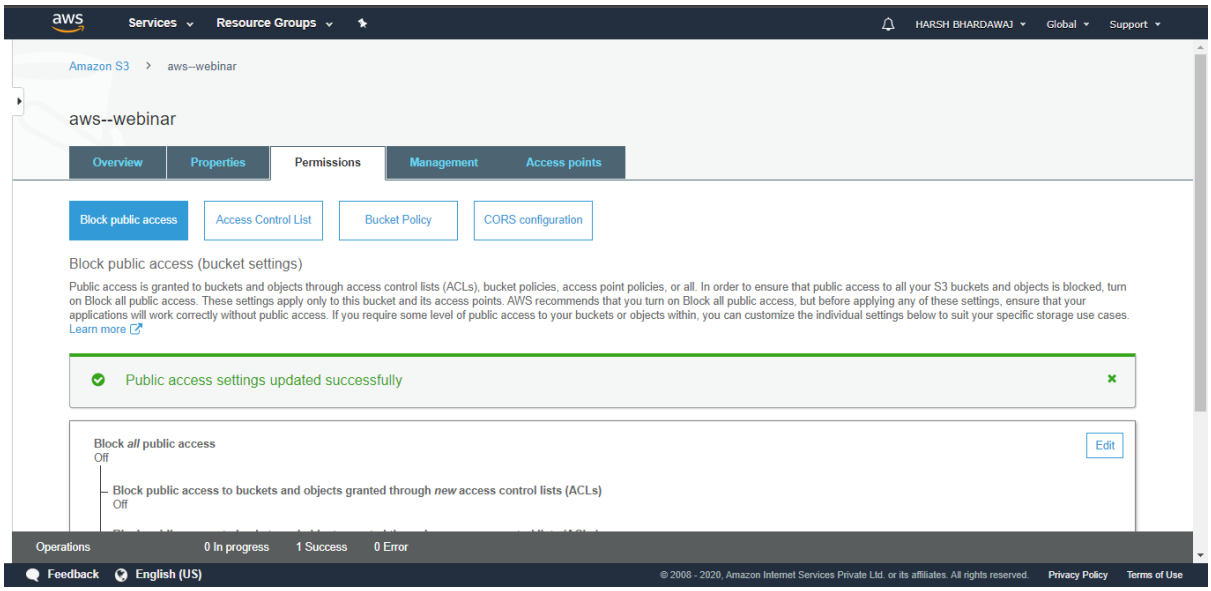


e) Making the Object Public:

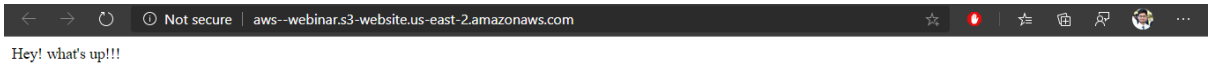
Step i: Disabling Public block:



Step ii: Successfully updated the public block permission:



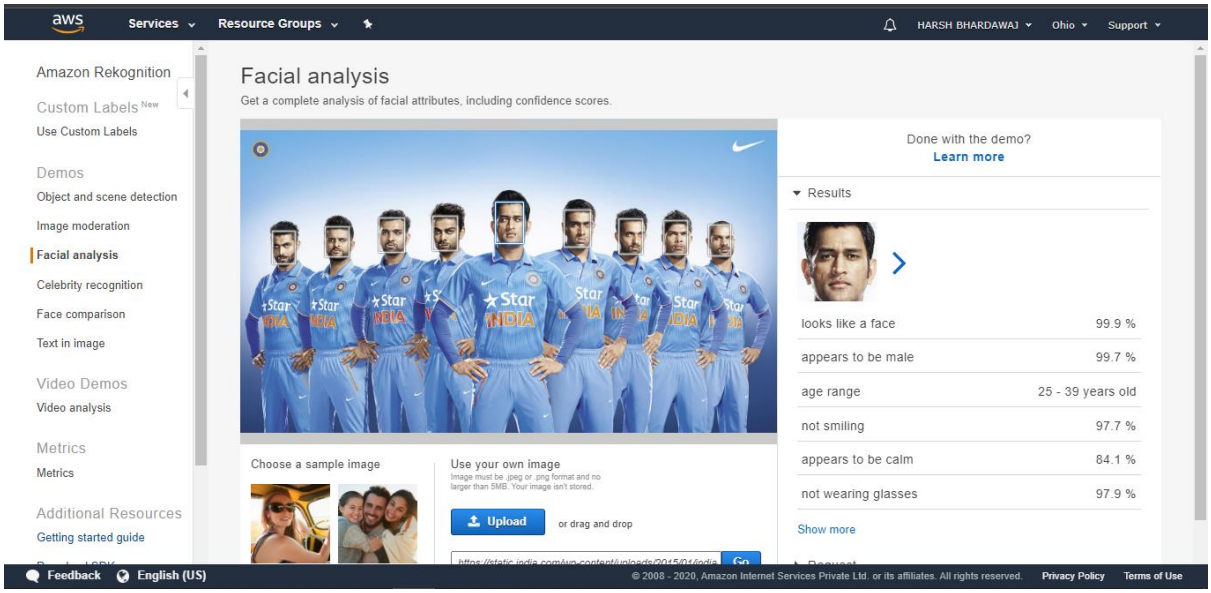
f) <u>Checking the S3 Link on the Browser:</u>



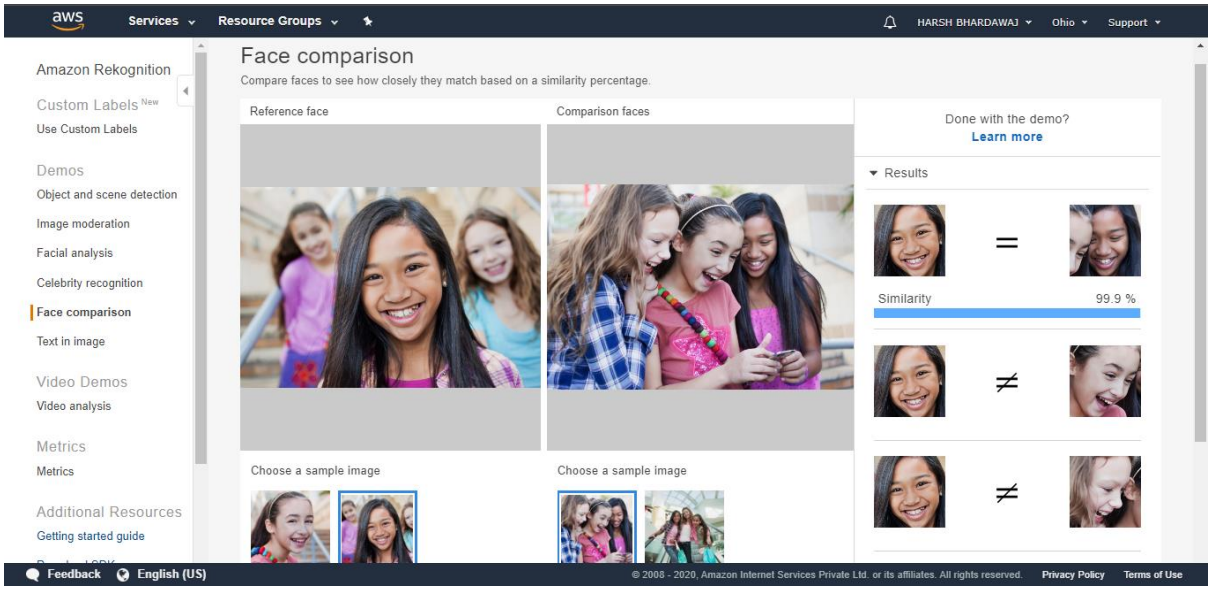
Hey! what's up!!!

Rekognition Screenshots:

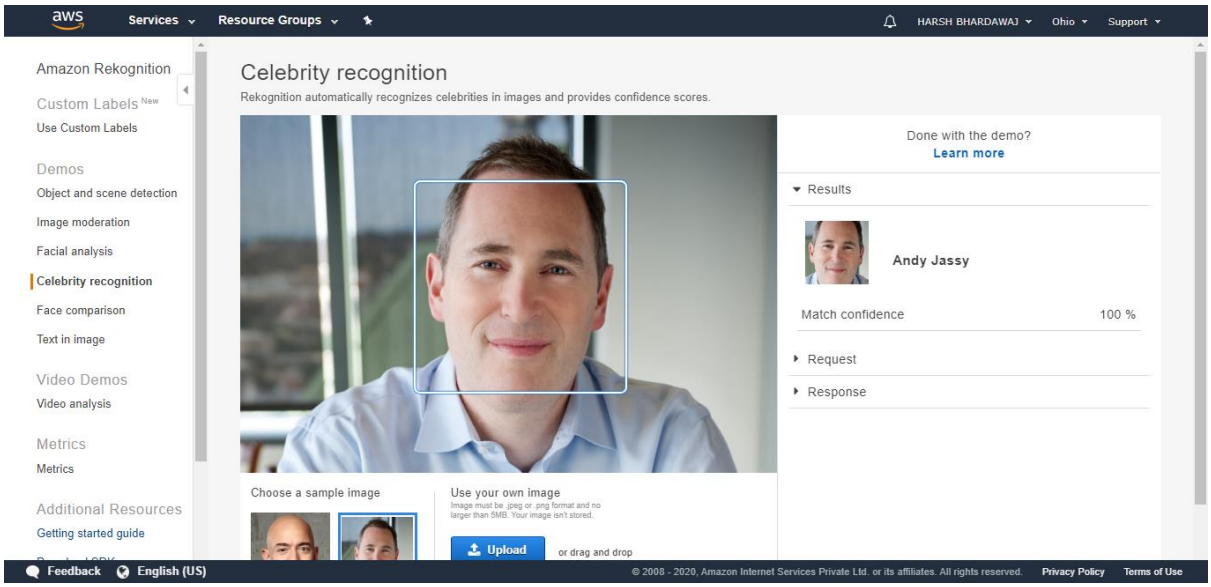
a) Face Detect:



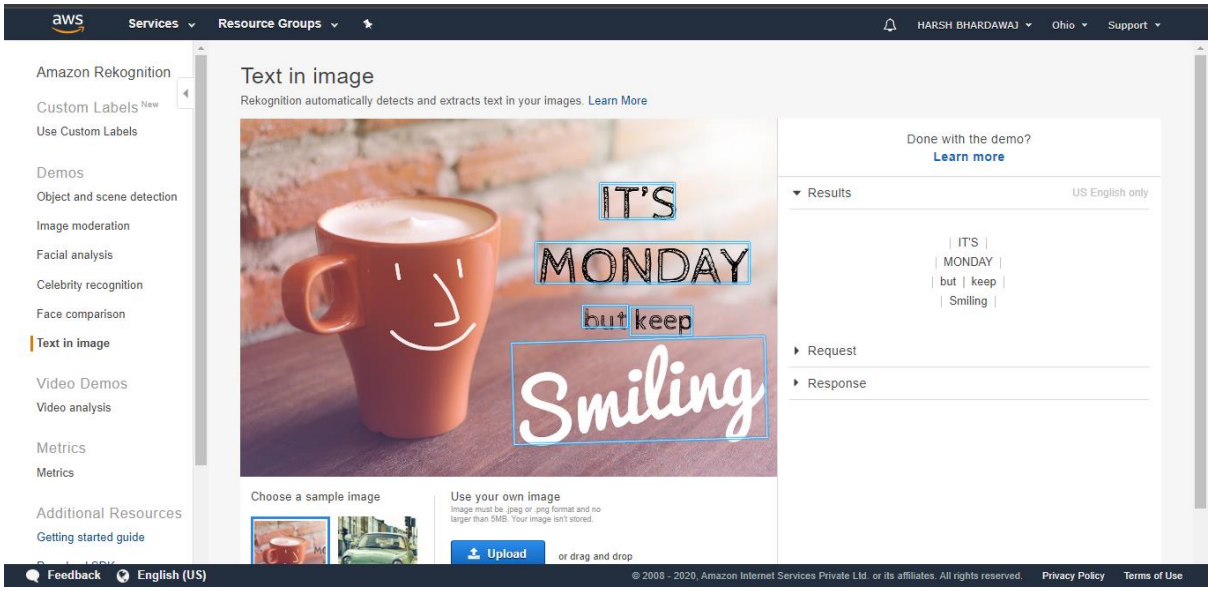
b) Face Compare:



c) Celebrity Recognition:

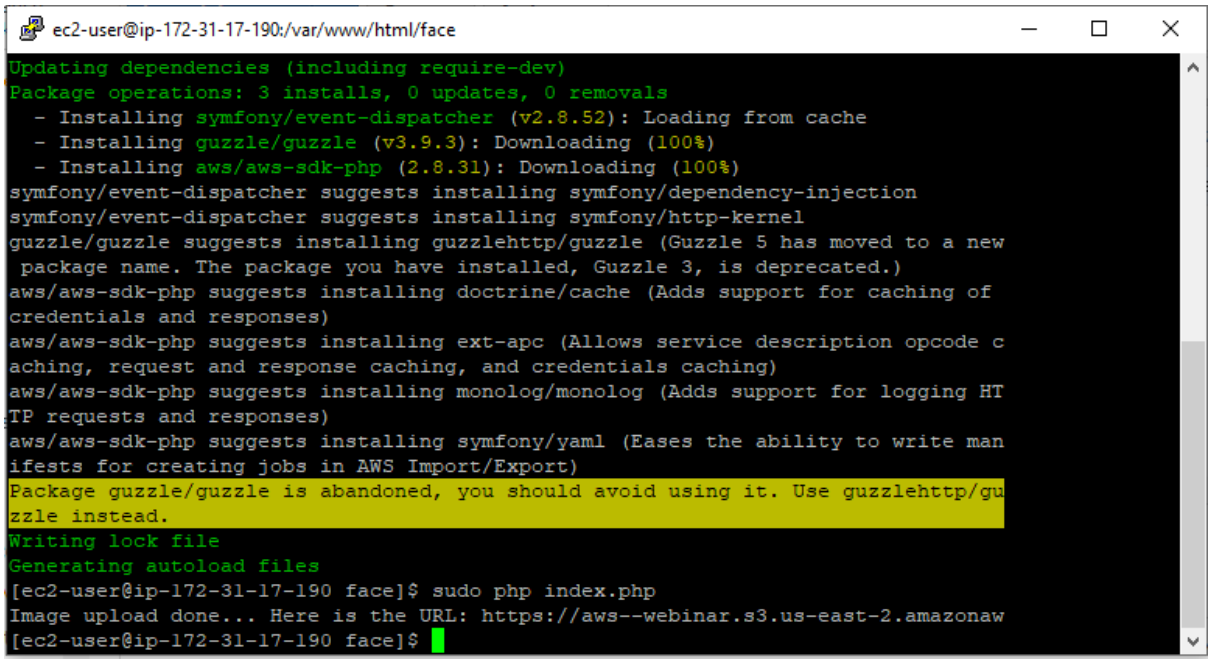


d) Text In Image:

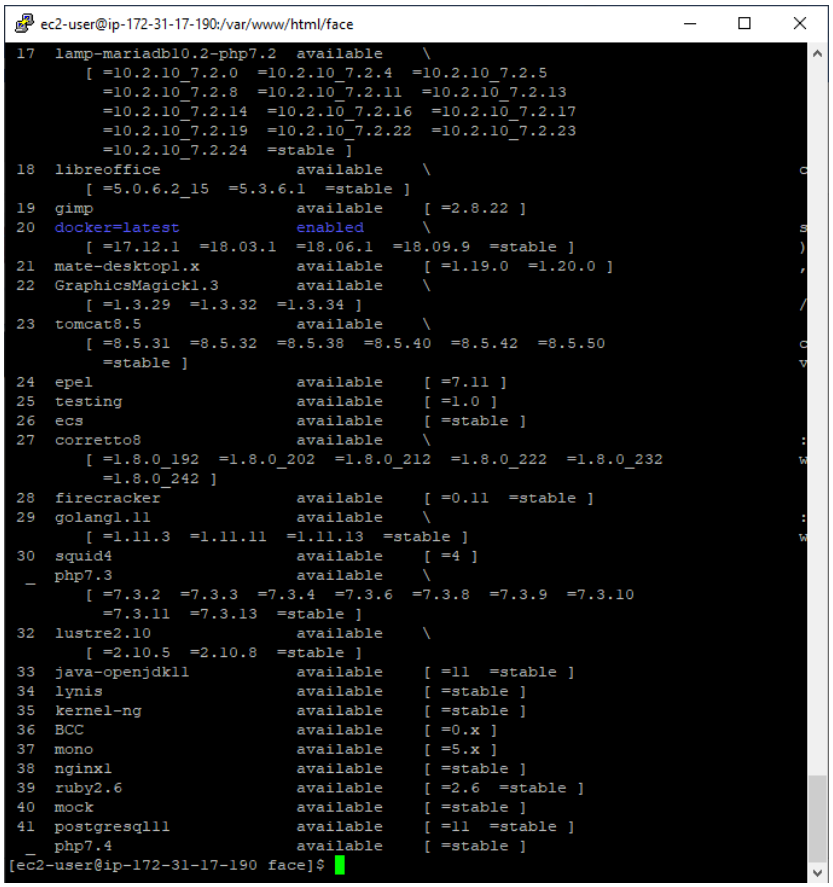


EC2 & S3 Configuration Screenshots:

a) Installing AWS-SDK:



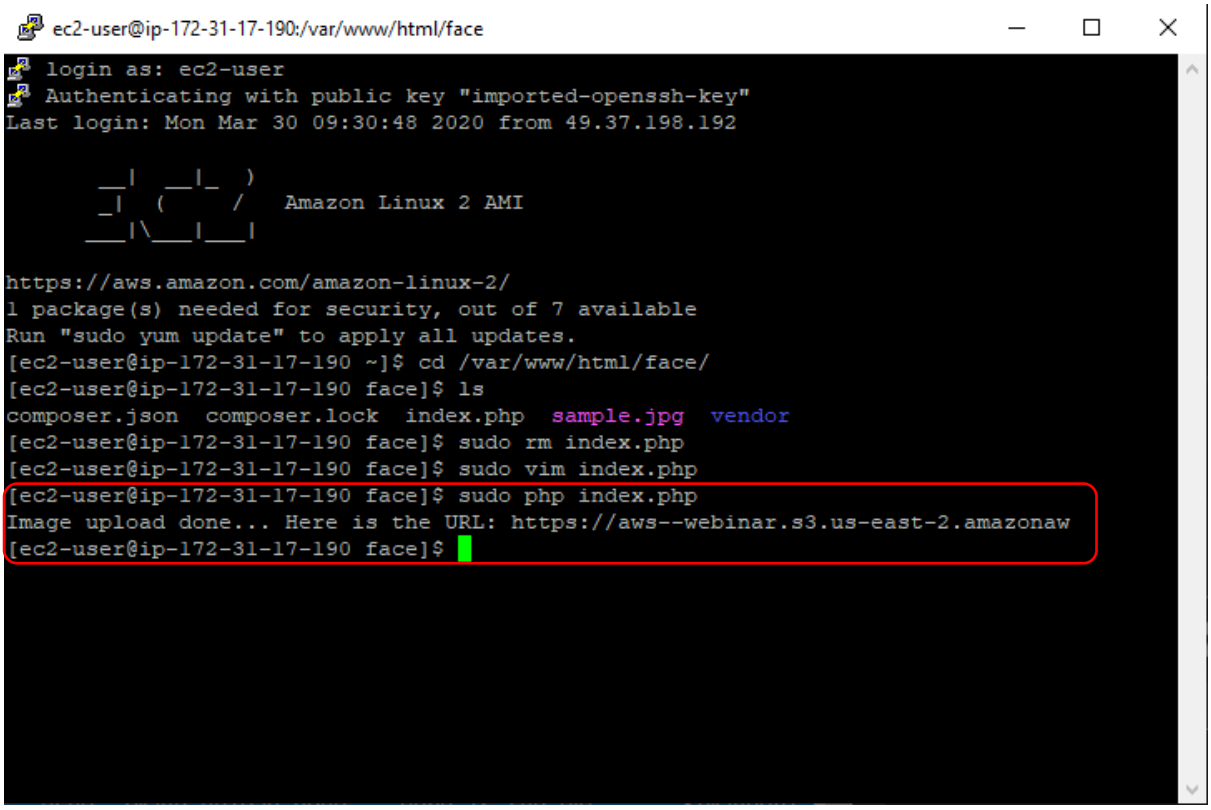
b) Installing PHP:



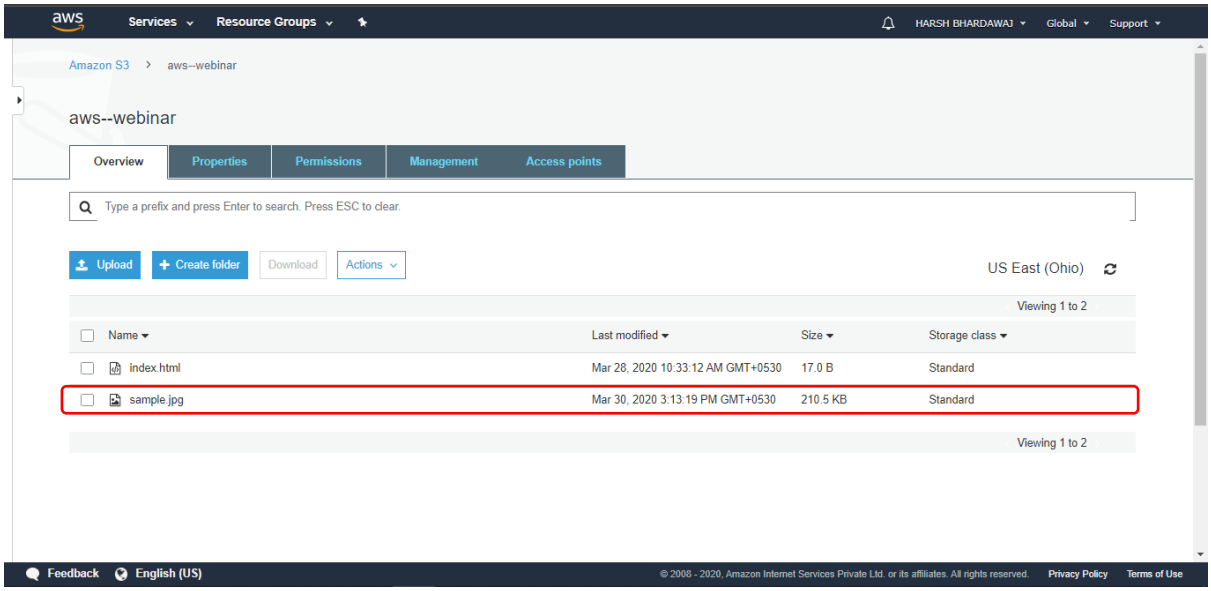
c) Index.php File Code:

d) Image Upload to S3:

Step i: Image Upload Successful:



Step ii: Uploaded Image in S3:

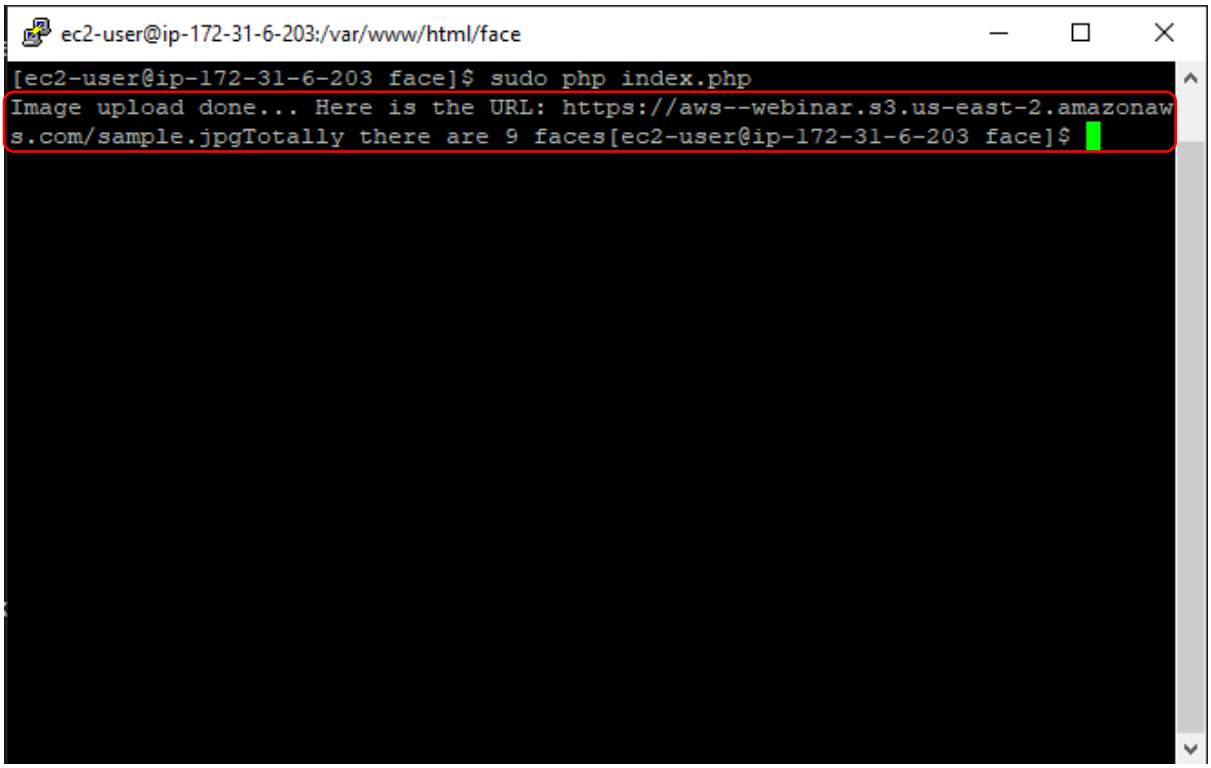


EC2 & Rekognition:

a) Face Detect Screenshot:

Step i: Feed in image:

Step ii: Successfully detected all the 9 faces from the uploaded image:



Conclusion:

Thus, using the Amazon Web Services we built the face recognition system based on the architecture as shown above. The Face detection is done and all the results are depicted in the report.