**Quiz Master**: Project Report

**Group name:** NULL Pointers

**Group**:

*Parth Patel*

*Dev Patel*

*Shailen Sutradhar*

*Gautam Santhanu Thampy*

**Due Date**: May 12th, 2025

**Class**: CMPE272 Sec 03

**Prof:** Andrew Bond

# Table of Contents

[Table of Contents 2](#_Toc198027051)

[Introduction 4](#_Toc198027052)

[Purpose 4](#_Toc198027053)

[Intended Audience 4](#_Toc198027054)

[Technical stack 4](#_Toc198027055)

[Architecture & High-Level Design 6](#_Toc198027056)

[Definitions and acronyms1 6](#_Toc198027057)

[Project risks 7](#_Toc198027058)

[Development process 7](#_Toc198027059)

[Milestones & Time schedule (Agile Methodology) 8](#_Toc198027060)

[Demo 10](#_Toc198027061)

[Arriving at the home page 10](#_Toc198027062)

[Logging in using email 10](#_Toc198027063)

[Create Quiz 13](#_Toc198027064)

[Attempting Quiz 14](#_Toc198027065)

[My Quizzes tab 16](#_Toc198027066)

[Technical Walkthrough 17](#_Toc198027067)

[Code Repository 17](#_Toc198027068)

[Testing using Vitest 17](#_Toc198027069)

[AWS API Gateway 18](#_Toc198027070)

[AWS S3 19](#_Toc198027071)

[AWS Lambda Functions 19](#_Toc198027072)

[AWS DynamoDB 22](#_Toc198027073)

[Firebase Authentication 23](#_Toc198027074)

[Deployment and CI/CD (Dev) 23](#_Toc198027075)

[Cloud Deployment 23](#_Toc198027076)

[CI/CD (Continuous Integration Continuous Deployment): 25](#_Toc198027077)

[Organization 28](#_Toc198027078)

[Project group 28](#_Toc198027079)

[Communication 29](#_Toc198027080)

[GitHub: github.com/ParthPatel00/QuizMaster 29](#_Toc198027081)

[Website: https://quizmaster.dedyn.io/ 29](#_Toc198027082)

[Citations 29](#_Toc198027083)

# Introduction

## Purpose

QuizMaster is designed to be an application that enables users to assess their knowledge and proficiency on a specific topic of their choice. Its core functionality is to generate quizzes based on user-provided content.

Users can upload a PDF document containing content for which they would like to test their knowledge in. QuizMaster will generate a quiz tailored to this content and provide it to the user in a simple, user-friendly interface. Upon completion, users can receive instant grading, feedback on correct and incorrect answers, and the ability to retake the quiz as many times as they like.

Furthermore, when logged in/signed up, QuizMaster will also allow users to save quizzes for future retakes.

## Intended Audience

The QuizMaster tool is designed to empower the executives and students across schools, universities, enterprises and a portion of general audience.

* Students can upload lecture slides and textbooks to generate tests helping them prepare for exams and be able to test their knowledge on a topic.
* Enterprises can integrate the tool in their employee recruitment, training and assessment.
* Universities can use QuizMaster to automate composing examination questions and reducing the risk of cheating with the ability to generate unique questions for each test.
* General audience can utilize the tool to generate quizzes from books, documents and research papers.

## Technical stack

**Frontend**: React, Typescript, Tailwindcss

**Backend**: AWS Lambda, API Gateway

**File storage**: Amazon S3

**Database**: AWS DynamoDB

**LLM**: Google Gemini

**Authorization**: Firebase

**CI/CD**: GitHub Actions, Code Build

**Version control**: GitHub, Git

Reasoning for selected tech stack:

**React**:

* Component based code for front-end makes it more readable and code is reusable for future uses
* Responsive and faster to load
* Better documentation and resources to create a user-friendly interface

**AWS S3 Bucket**

* For file storage of PDFs

**AWS DynamoDB**

* Used as a document store of JSON files received from Gemini

**AWS Lambda**:

* Affordable cost due to its serverless function and only queries made to the function are charged
* Easier integration with other AWS services
* Faster

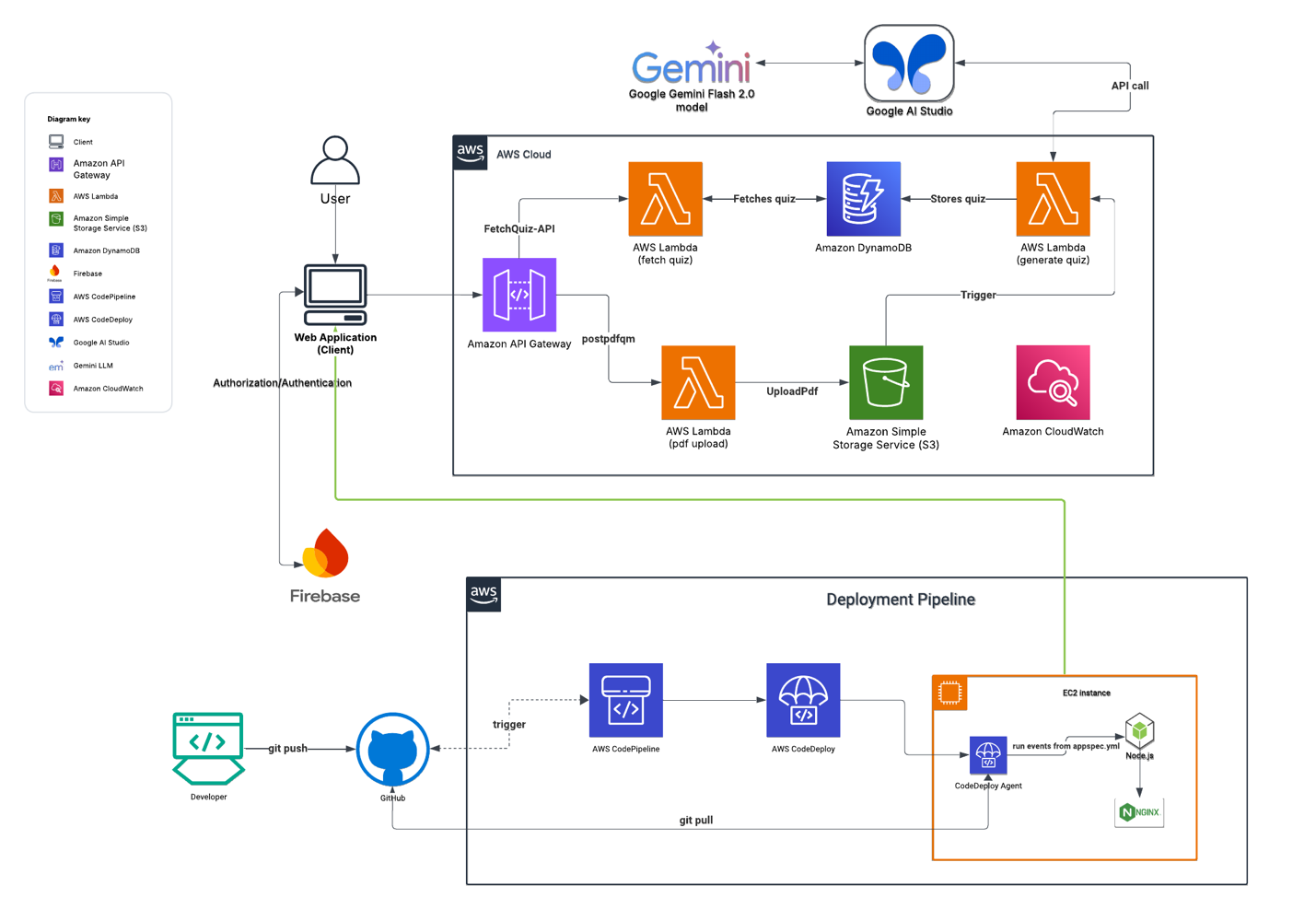
**AWS API Gateway**:

* Secure deployment of Rest APIs
* Scalable to handle large number traffic loads
* Seamless integration with AWS Lambda

**Code Build:**

* Offers free 100 build minutes/month
* Allows automated software release through multiple environments

## Architecture & High-Level Design



Link to [Lucid Chart](https://lucid.app/lucidchart/688ae160-42d1-405e-a50b-52bd6cf6056e/edit?viewport_loc=222%2C412%2C1866%2C930%2C0_0&invitationId=inv_4212941e-a821-4b6b-b1a9-7bd71192c72f)

## Definitions and acronyms[1](#_Citations)

**AWS**: Amazon Web Services. A cloud computing platform that offers a variety of services, including storage, computing and content delivery.

**EC2**: Amazon Elastic Compute Cloud. A web service that let users run application in the Amazon Web Services cloud.

**S3**: Amazon Simple Storage Service. A cloud-based object storage service that allows users to store and retrieve data.

**LLM**: Large Language Model. A type of artificial intelligence that can understand and generate human language.

**API Gateway**: Application Programming Interface Gateway. A software tool that manages API calls between clients and backend services.

**DynamoDB**: Amazon DynamoDB. A cloud-based database service.

**CI/CD**: Continuous Integration and Continuous Deployment/Delivery. A software development practice that uses automation to speed up and improve the software development process.

**Git**: Global Information Tracker. A free, open-source version control system.

# Project risks

**Accessibility**: The application must be accessible by the users all time. For server or some other failure, the user might not be able to access the website at all. Besides, the saved data might be inaccessible for the user.

**Responsiveness**: The webpage might not be responsive across devices with different screen sizes which might lead to broken layout, unreadable texts and a poor user interface.

**File** **Handling**: The application will not function if the user uploads a file too big for the LLM or a file of inappropriate type.

**Performance**: It might take long for the quiz to be generated which will lead to inefficiency. Sometimes, failure to generate quizzes can also happen.

**Security**: The authentication can fail, which will lead to unauthorized access. There can be breaches in stored quiz data leading to privacy and security concerns.

**Incorrect or Unethical Quiz**: The AI model might generate unrealistic or inappropriate questions and incorrect answers. It might also generate questions out of the given material.

**Scalability**: A high number of users trying to access the webpage can slow down the application and cause it to crash.

# Development process

1. Requirements gathering
   1. Coming up with features
   2. Designing the website
   3. Choosing an LLM
2. Creating the frontend
   1. Creating a home page
   2. Creating a navbar
   3. Creating a quiz page
   4. Creating the My Quizzes page
   5. Creating authorization
   6. Integrating frontend buttons with AWS Gateway endpoints.
   7. Testing
3. Creating an organization on AWS for team collaboration
4. Creating storage (S3 and DynamoDB)
5. Setting up API gateway and Lambda functions
   1. Setting up
   2. Implementing
   3. Testing
6. Setting up Gemini account
7. Enabling Code Deploy and EC2 for deployment
   1. Provisioning
   2. Testing

# Milestones & Time schedule (Agile Methodology)

|  |  |  |
| --- | --- | --- |
| Tasks to Accomplish | Total weeks | Complete by |
| Requirements gathering  Research which LLM model to use, and setup an account with it (ChatGPT/Gemini/Deep Seek etc.)  Creating an AWS organization account for collaboration | 1 | Feb 10th |
| Develop a functional home page.  Creating an S3 bucket  Setting up a Dynamo database | 2 | Feb 24th |
| Milestone #1  Team meeting + Testing + Next steps | | |
| Create the Quiz frontend component  Implement the pdfUploadToS3 Lambda function  Create NoSQL mock data for DynamoDB  Test connection to AWS lambda  Test trigger events to S3  Test responsiveness of frontend | 2 | March 10th |
| Milestone #2  Team meeting + Testing + Next steps | | |
| Setup Firebase and backend for authorization  Do prompt engineering with LLM of choice.  Test authorization with Gmail, and new email  Test example PDFs with prompts | 1 | March 17th |
| Milestone #3  Team meeting + Testing + Next steps | | |
| Implement the Lambda function responsible for populating DynamoDB  Implement backend for fetching data from DynamoDB  Test populate dynamo lambda function  Test fetching data from dynamo lambda function | 2 | March 31st |
| Milestone #4  Team meeting + Testing + Next steps | | |
| Provisioning EC2 and Code Build for deployment  End-to-end testing | 2 | April 14th |
| Milestone #5  Team meeting + Final Testing + Presentation planning | | |
| Creating a presentation  Assigning slides to each member  Practice presenting | Remaining time: ~ 1 Month | May 12th |
| PRESENT PROJECT IN CLASS | | |

# Demo

## Arriving at the home page

The application starts at the QuizMaster homepage (<https://quizmaster.dedyn.io/>). The homepage features:

* A navbar at the top with a “Sign up/Login” button.
* A central area providing a brief introduction to the features of the app.
* A blue “Create Quiz” button that allows users to start creating their AI-generated quiz.

A screenshot of a quiz master

AI-generated content may be incorrect.

## Logging in using email

A key feature of this application is the ability for users to return to quizzes they've previously created through the “My Quizzes” tab. However, accessing this feature requires signing in.

* **Sign In or Sign Up**: The user can click the “Sign up/Login” button to sign in or create a new account.

A screenshot of a computer

AI-generated content may be incorrect.

**Option 1: Sign in with Google**

* If the user selects “Sign in with Google,” they’ll be prompted to choose from their Google account emails.
* Upon signing in, the user will be redirected to a personalized homepage with an updated navbar featuring:
  + The “My Quizzes” tab.
  + Their Google account picture.
  + A “Logout” button.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a quiz master

AI-generated content may be incorrect.

**Option 2: Sign Up Using a New Email and Password**

* If the user chooses to sign up with a new email and password, they’ll still be able to save quizzes, but the homepage layout will be slightly different.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a quiz master

AI-generated content may be incorrect.

## Create Quiz

When the user clicks on the “Create Quiz” button, they are prompted to:

* Enter a title for their quiz.
* Upload a PDF file.

**File Constraints**:

* The file must be in PDF format.
* The file size must be less than 5MB.

These constraints ensure ease of parsing and optimize token usage for the LLM model (Gemini). Once both inputs are completed, the user can click the “Generate Quiz” button.

**Progress Indicators**:  
While the quiz is being generated, a status bar is displayed, which goes through the following steps:

* “Connecting to AWS…”
* “Uploading your file…”
* “Fetching Generated Quiz…”

Once all steps are completed, the user will be directed to the quiz page.

A screenshot of a quiz

AI-generated content may be incorrect.

## Attempting Quiz

The quiz page consists of up to 5 questions (the current limit). Each question may be:

* True/False, or
* Multiple choice with 4 options, where one option is correct.

The user must select a single option for each question before submitting the quiz.

**Post-Submission Feedback**:  
After submitting the quiz, the user will see their score. The feedback includes:

* Correct answers highlighted in green.
* Incorrect answers highlighted in red.

This feedback allows the user to learn from their mistakes and either retake the quiz immediately or later.

* The “Retake Quiz” button will reset all answers.
* The “Generate a New Quiz” button will take the user back to the homepage.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a quiz

AI-generated content may be incorrect.

## My Quizzes tab

The “My Quizzes” tab allows the user to see all quizzes they’ve created.

* Clicking on “View Quiz” will take the user to the quiz page again, where they can view the details of the selected quiz.

A blue and white rectangle

AI-generated content may be incorrect.

# Technical Walkthrough

## Code Repository

All code related to the frontend and client-side backend is stored on GitHub ([QuizMaster Repository](https://github.com/ParthPatel00/QuizMaster)). The majority of the code resides in the frontend folder, where components are split into different subfolders for better readability.

## Testing using Vitest

Testing of APIs was conducted using Vitest library:

There are a total of 7 test cases. They are as follows:

1. Renders the Create Quiz button
2. Shows error when trying to upload a file too big
3. Shows error when trying pass an empty name
4. Shows error when unable to fetch the quiz from Gemini
5. Fetches Quiz using GET operation from DynamoDB and renders correctly on QuizPage.
6. Performs the fetch Quiz operation when “My Quizzes” button is clicked
7. Performs exponential polling from DynamoDB instead of failing on the first try.

The test cases are run using the CLI command “npx vitest” which runs the test cases as shown below

A screenshot of a computer program

AI-generated content may be incorrect.

## AWS API Gateway

Two API Gateways have been created:

1. **postpdfqm Gateway** (REST Protocol):
   * Responsible for accepting the quiz name and PDF file, creating a unique destination location in an S3 bucket, and uploading the file.
   * Only handles POST operations.
2. **fetchQuiz-API Gateway** (HTTP Protocol):
   * Responsible for fetching the quiz from DynamoDB once created.
   * Also used to fetch quizzes in the "My Quizzes" section.
   * Only handles GET operations.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

## AWS S3

The S3 bucket serves as the landing zone for PDF files. Once a unique destination name is generated, the files are uploaded.

* The bucket can store up to 5GB of data under the free tier.
* Each file is stored with a unique name in the format:  
  <name\_of\_file>\_\_<quiz\_name>\_\_<datetime>\_\_<email\_if\_provided>.pdf

A screenshot of a computer

AI-generated content may be incorrect.

## AWS Lambda Functions

A screenshot of a computer

AI-generated content may be incorrect.

There are four Lambda functions, each with a specific role:

1. **Generate-presigned-url**
   * Accepts the quiz name and file via the API Gateway and uploads the file into the S3 bucket.

A screenshot of a computer

AI-generated content may be incorrect.

1. **Process\_pdf\_lambda**
   * Triggered once the file is uploaded to S3.
   * Parses the PDF, sends it along with a prompt to Gemini, and receives a JSON containing questions and answers.
   * The JSON response is then stored in DynamoDB.

A screenshot of a computer

AI-generated content may be incorrect.

1. **fetchQuiz**
   * While the previous functions execute, the client-side performs exponential polling on this function to check if the quiz is available in DynamoDB.
   * If the quiz is found, it fetches and displays the data.
   * If not found after 10 tries, a failure message is shown.

A screenshot of a computer

AI-generated content may be incorrect.

1. **fetchUserQuizzes**
   * Triggered when the user clicks "My Quizzes." It fetches quizzes generated by the logged-in user via a DynamoDB query.
   * Accepts the user’s email and returns a list of quizzes for display on the client side.

A screenshot of a computer

AI-generated content may be incorrect.

## AWS DynamoDB

A DynamoDB table named QuizSystem is used, with an index on quiz\_id.  
Table attributes include:

* quiz\_id
* document\_name
* error\_message
* generated\_at
* metadata
* questions
* quiz\_name
* source\_document
* status
* user\_id

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

## Firebase Authentication

Firebase Authentication is used to allow users to log in via email or Google account. A Firebase project has been set up with the necessary configurations, including sign-in providers and allowed domains.A screenshot of a computer

AI-generated content may be incorrect.

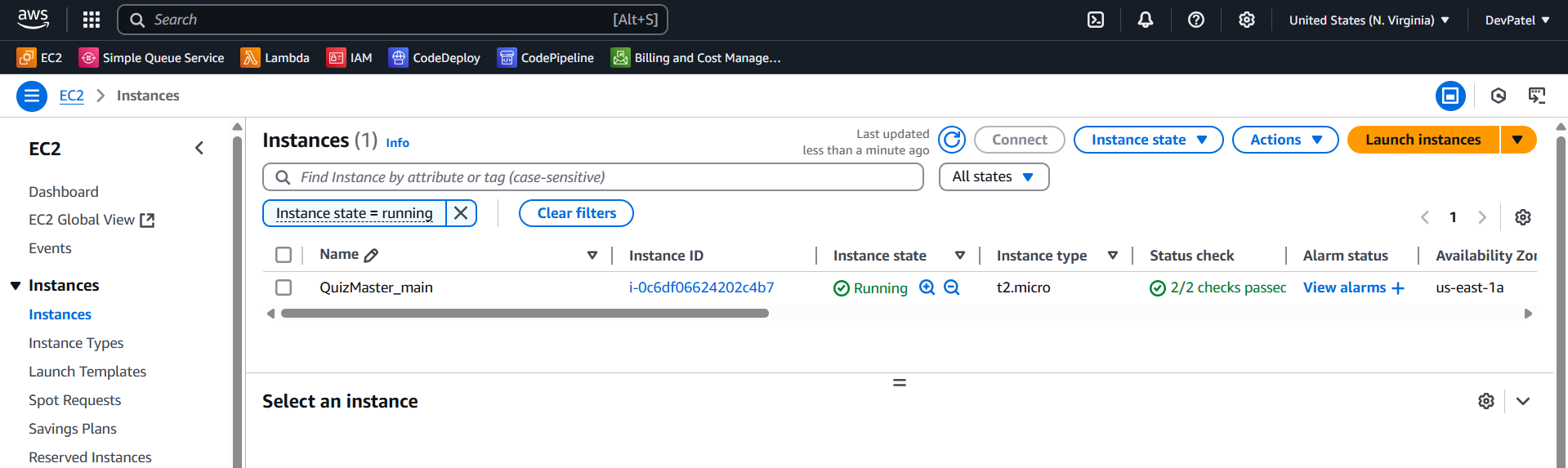
## Deployment and CI/CD (Dev)

Cloud Deployment

For demo purposes and to avoid incurring unnecessary charges the application was deployed plainly on AWS EC2 using Nginx for reverse proxy to mask the Public IP of the EC2 instance and setup SSL security with OpenSSL.

EC2 Instance was created with Ubuntu OS, t2.micro instance type, and Free tier eligible 30 GB storage, the Instance was setup in us-east-1 (N. Virginia) region for faster access and low cost. Security Group assigned to it had ports 80 and 443 were exposed publicly for HTTP and HTTPS requests.

AWS EC2

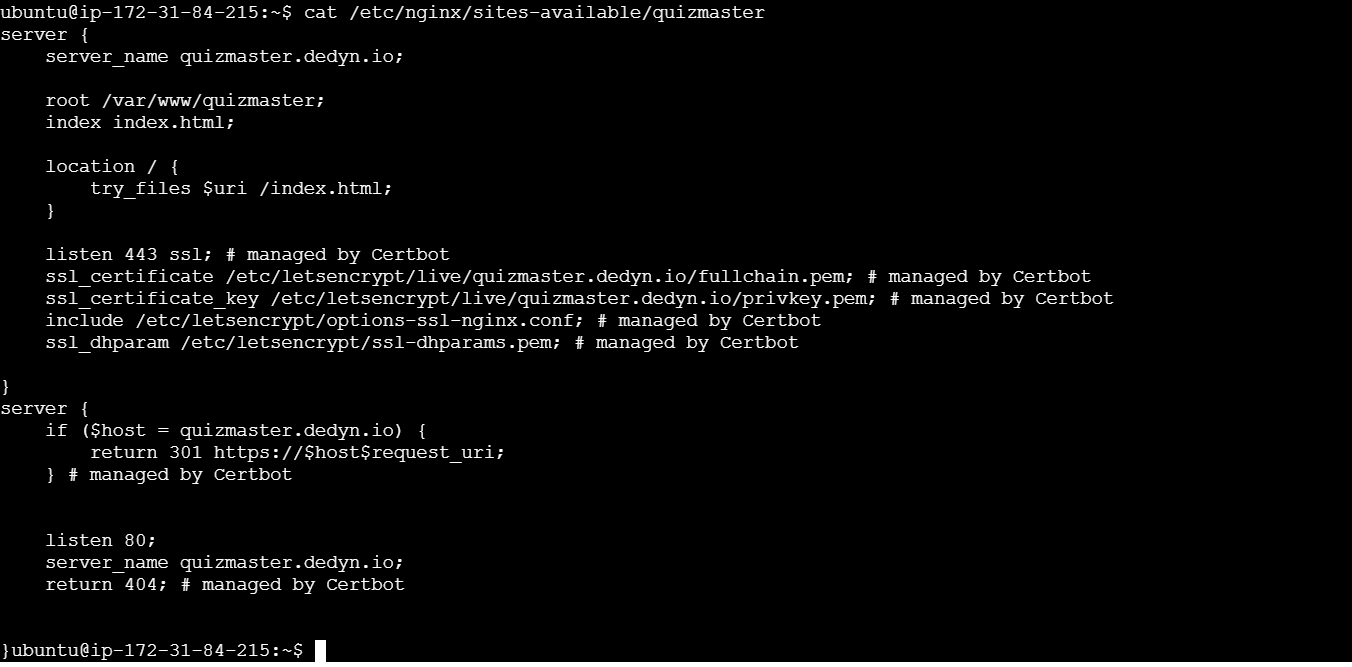


The domain quizmaster.dedyn.io was acquired from the desec.io DNS provider and the DNS record with CNAME A was updated to the public IP of the EC2 instance.

<https://desec.io/domains/quizmaster.dedyn.io>

The GitHub Repo was cloned into the EC2 Instance using AWS EC2 Instance Connect and after installing necessary dependencies using npm and installing the libraries required including node JS, OpenSSL, etcetera.

After testing out the frontend on Public IP using **npm run dev** the build for the application was created using **npm run build** that built the vite app and a dist folder was created contents of which were then copied into a folder at path var/www/quizmaster and then Nginx config file was edited to host this folder for frontend instead of the default Nginx frontend.

Once that was done the Nginx was routing the domain to public IP and the application was visible on the domain. After that SSL was set up by installing certbot and then linking the newly obtained SSL certificate to the quizmaster.dedyn.io domain.

sudo apt update

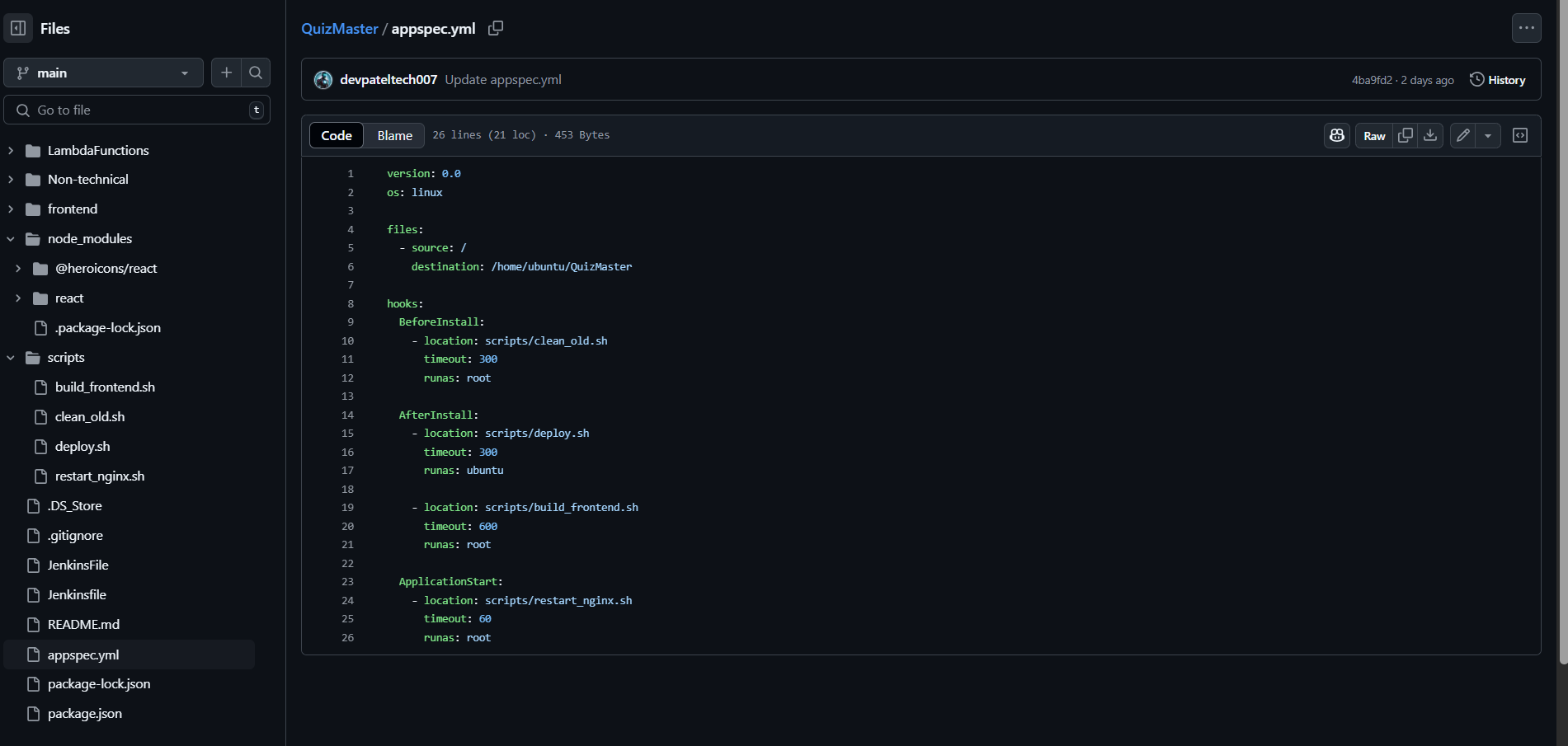
sudo apt install certbot python3-certbot-nginx -y

sudo certbot --nginx -d quizmaster.dedyn.io

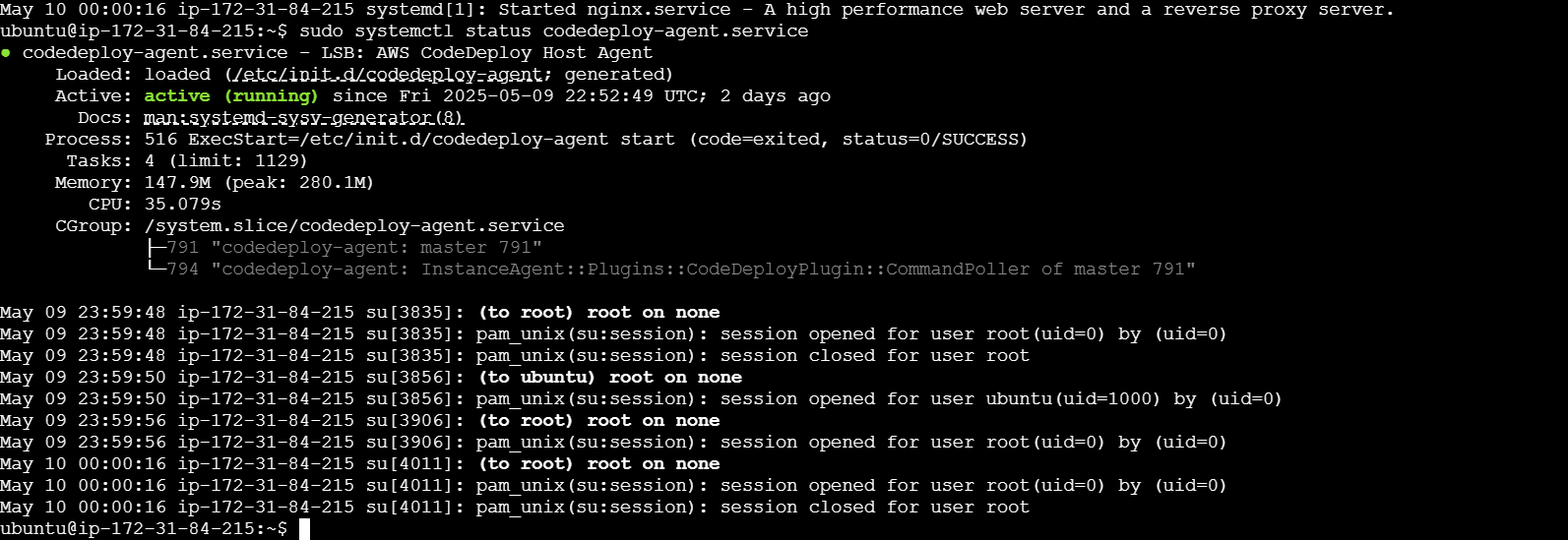
### CI/CD (Continuous Integration Continuous Deployment):

AWS CodePipeline was used to set up a CI/CD pipeline with source as GitHub repo and GitHub account owner had to authorize AWS CodePipeline access to GitHub hooks for listening to Push requests which then triggers the CI/CD pipeline to kickoff.

**appspec.yml** file was added to the root directory of GitHub Repo for AWS CodeDeploy to pull the stages of deployment with the events and execute them chronologically.



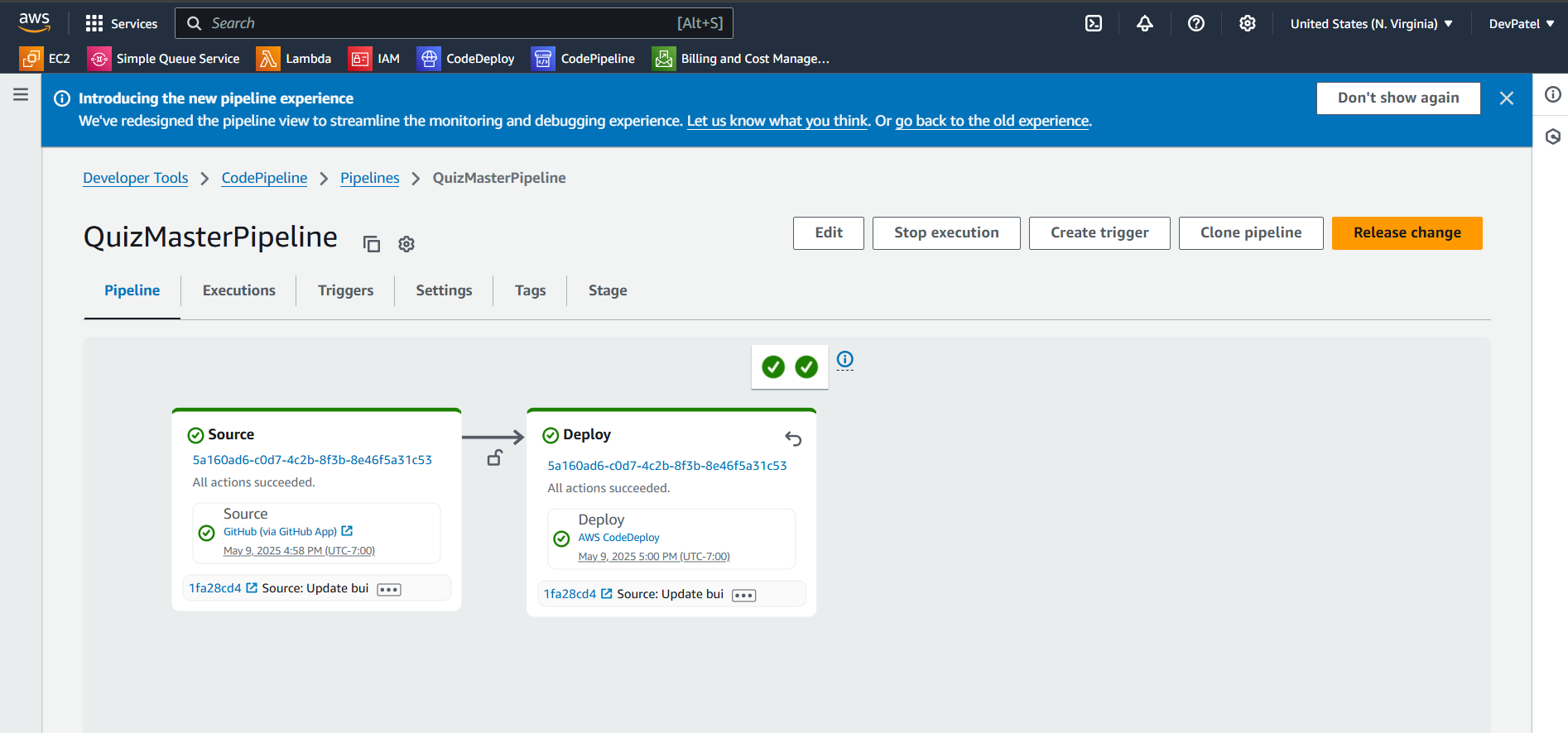
CodeDeploy Agent was installed on EC2 which is running as a service to execute the CodeDeploy commands.

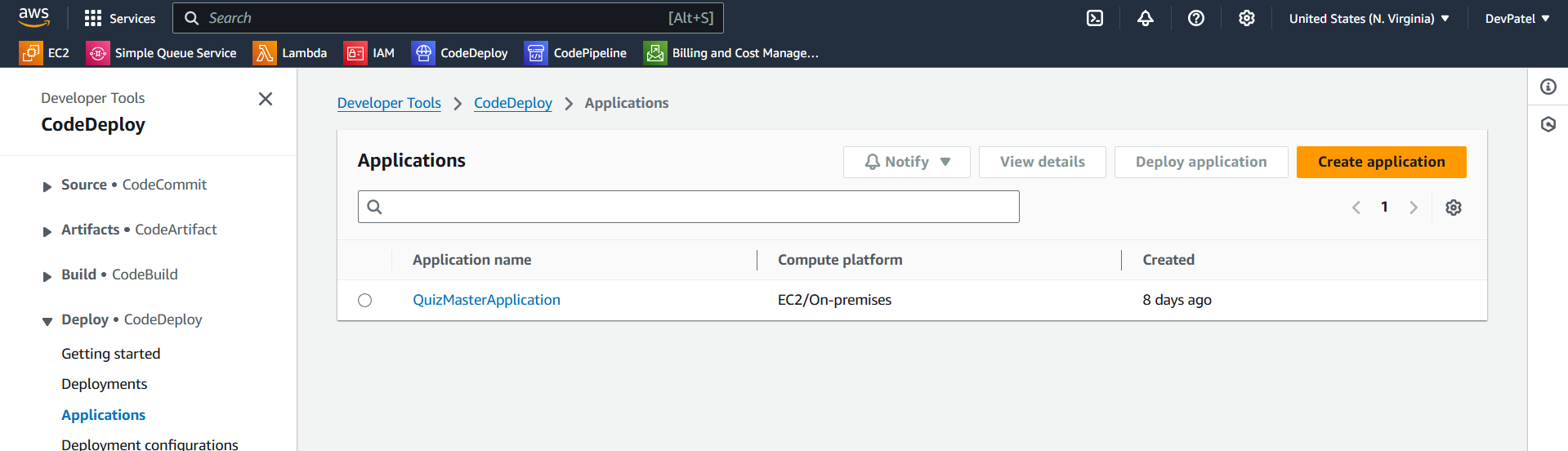


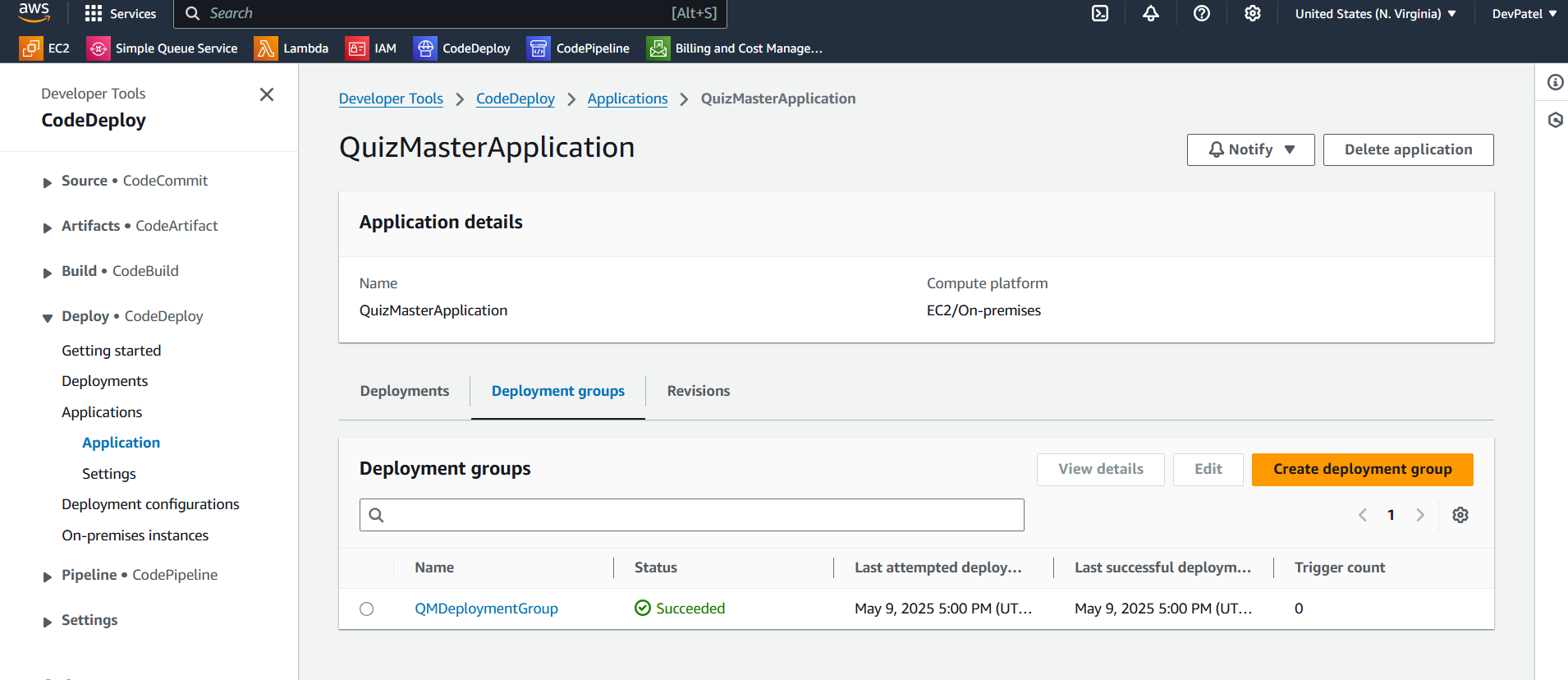
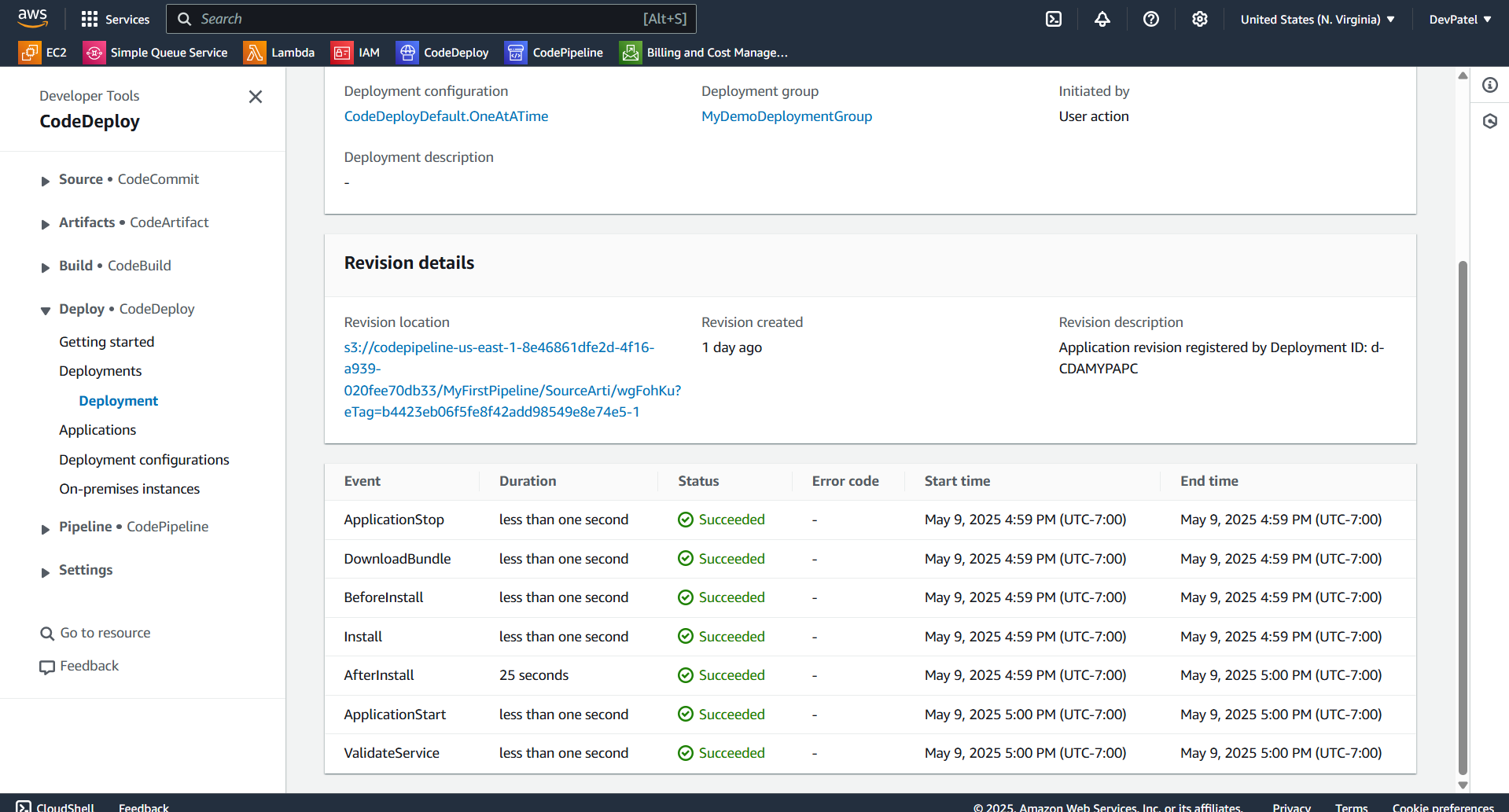
There are 3 scripts under the /**scripts** folder in the repo root directory which contains bash scripts to remove the old files from the folder, pulls new changes into the EC2 and then do npm install and npm run build to build the frontend, replace it into Nginx hosted files and then restart Nginx. The site keeps serving the old front end till the pipeline has completed all the stages.

The CI/CD Pipeline works as follows:

1. A commit pushed to the GitHub Repo.
2. AWS CodePipeline triggered.
3. AWS CodeDeploy pulls the GitHub Commits and deploys them on EC2 using the CodeDeploy Agent installed on the EC2 Instance.





# Organization

## Project group

|  |  |
| --- | --- |
| Name | Tasks |
| Parth Patel | * Frontend development * Client-side backend develop * FetchQuiz & FetchUserQuizzes Lamda development * Authentication/Authorization development * Testing |
| Shailen Sutradhar | * Requirements Gathering * Generate\_presigned\_url Lambda development * Provisioning resources |
| Dev Patel | * Cloud Deployment * Continuous Integration /Continuous Deployment pipeline * EC2 Provisioning * Cost Optimization |
| Gautam Santhanu Thampy | * Process\_pdf Lambda development * Prompt engineering |

# Communication

## GitHub: [github.com/ParthPatel00/QuizMaster](https://github.com/ParthPatel00/QuizMaster)

## Website: <https://quizmaster.dedyn.io/>

# Citations

* 1. <https://docs.aws.amazon.com/glossary/latest/reference/glos-chap.html#:~:text=Amazon%20Web%20Services%20(AWS),is%2Dcloud%2Dcomputing%2F%20>.