**PROJECT – 1**

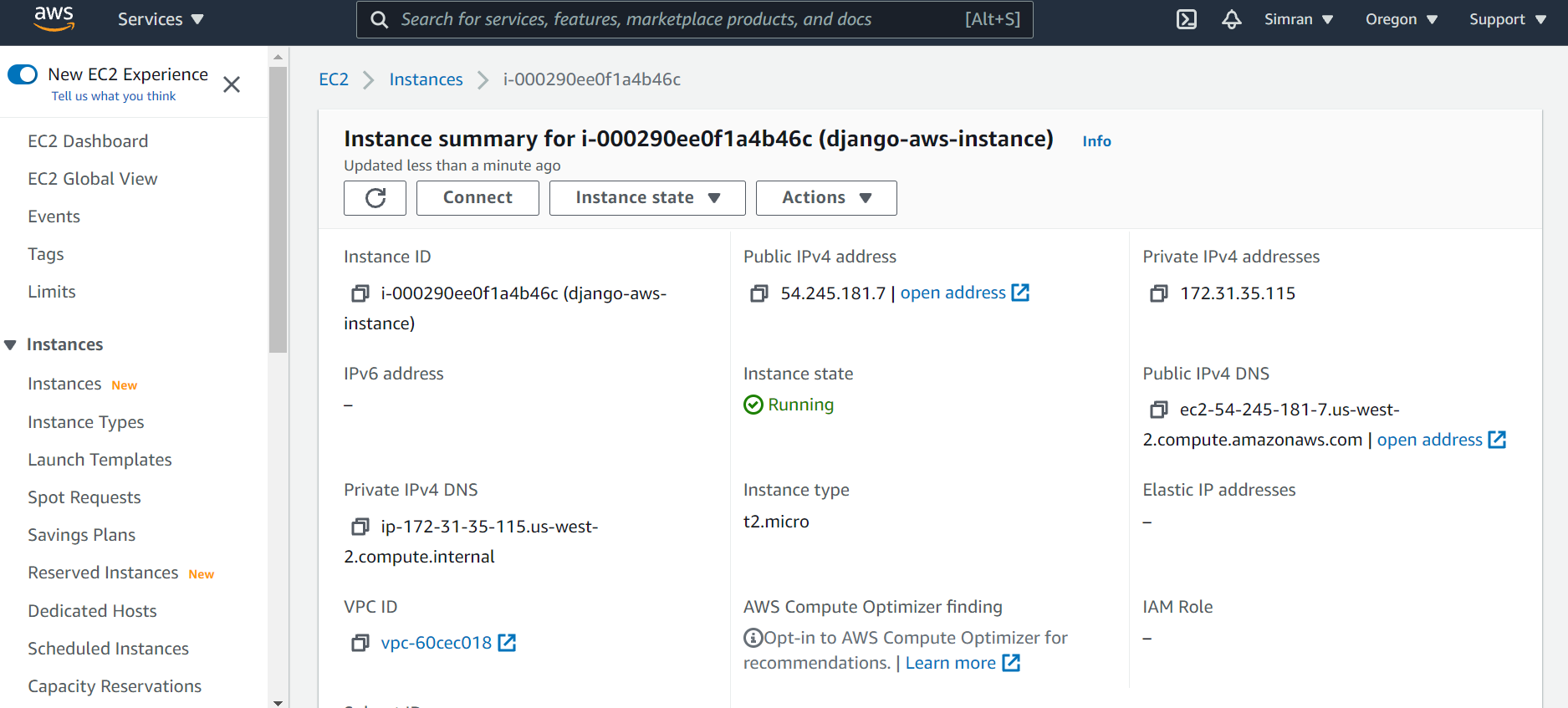
Name: Simran Tanvir Memon

Student Id: 015950610

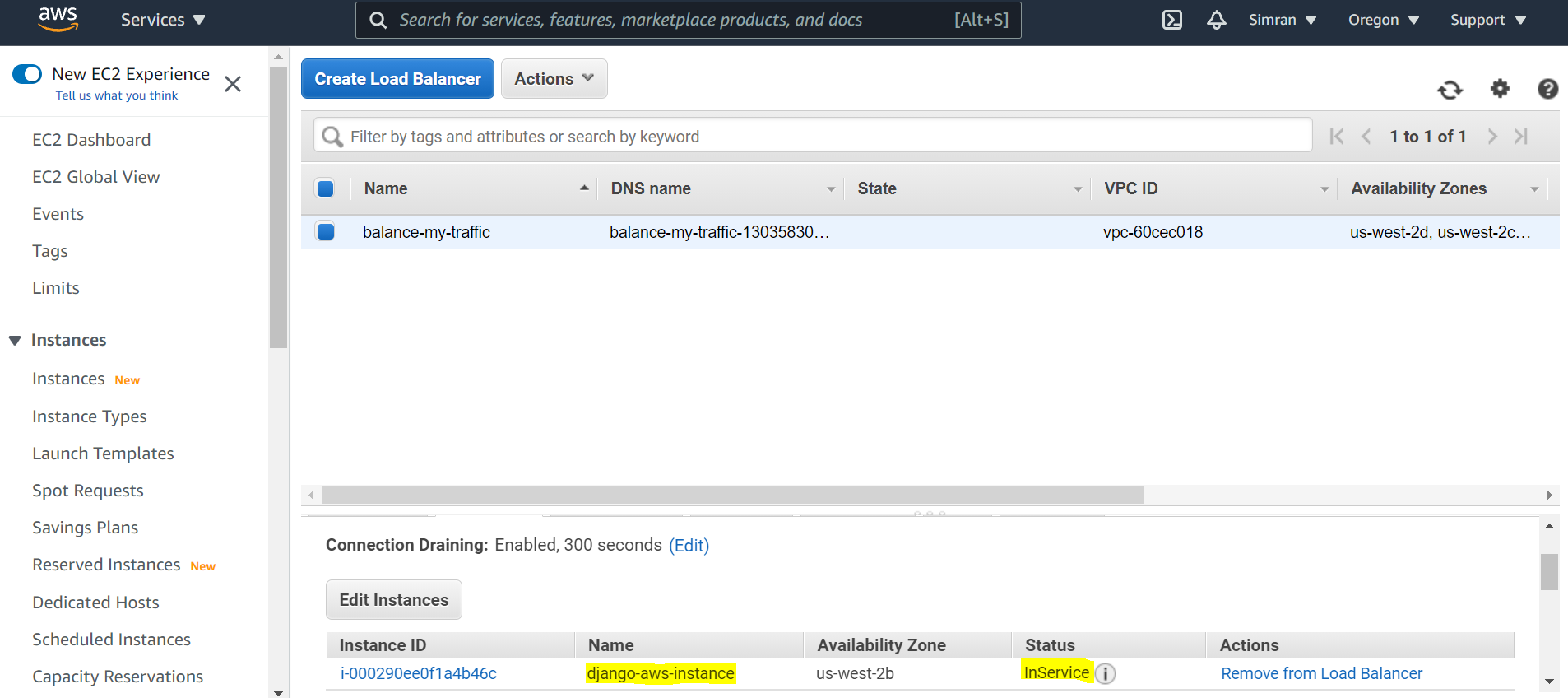
Github link: <https://github.com/simran-memon/cloudproject1>

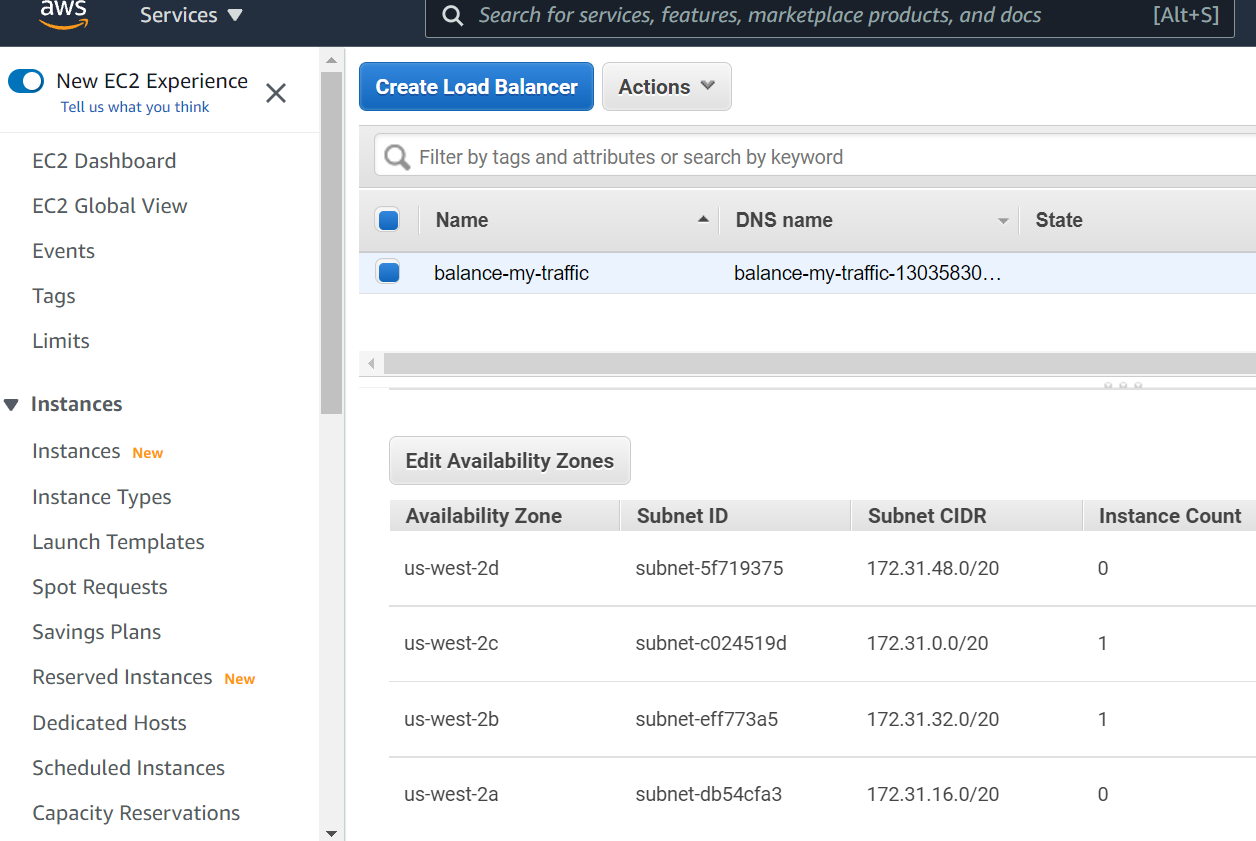
Domain name: simranmemon.link

* **Screenshots of various AWS resources created as a part of solution:**
* **AWS EC2: “django-aws-instance”**

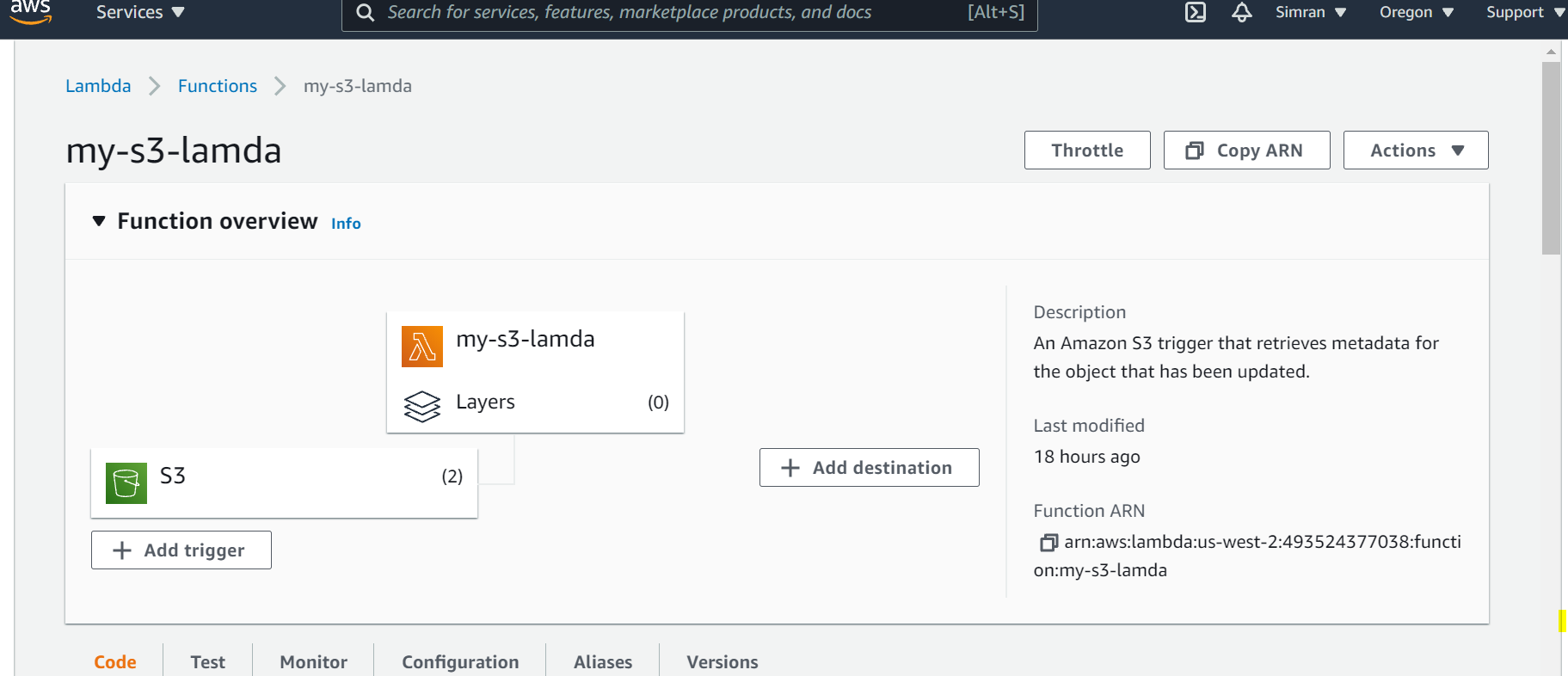


* **AWS ELB: “balance-my-traffic”**

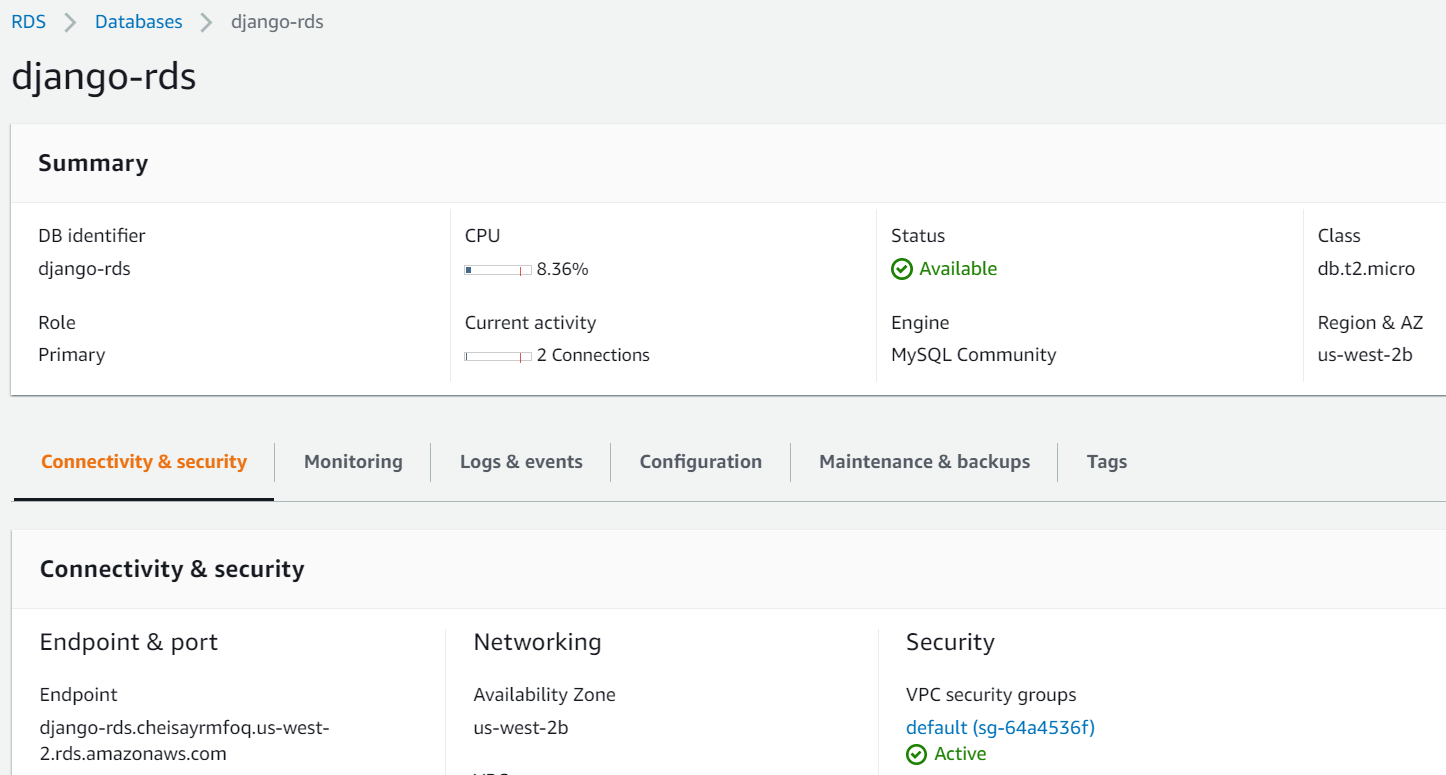




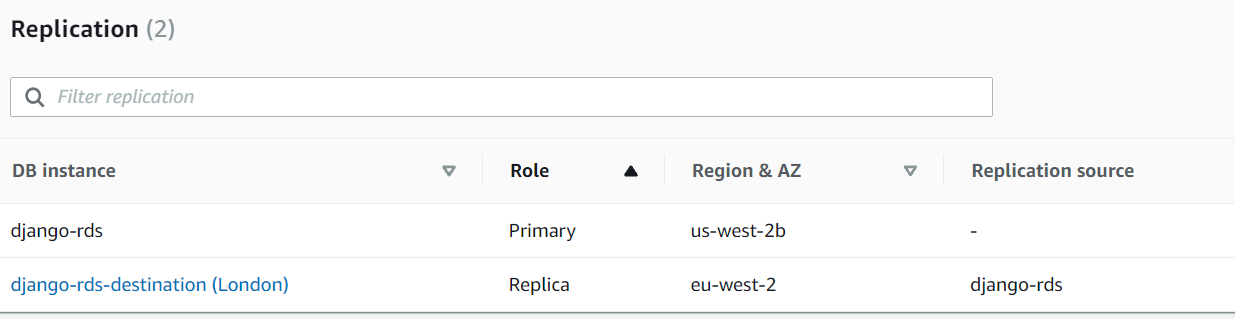
* **AWS LAMBDA:** function name: **“my-s3-lambda”**



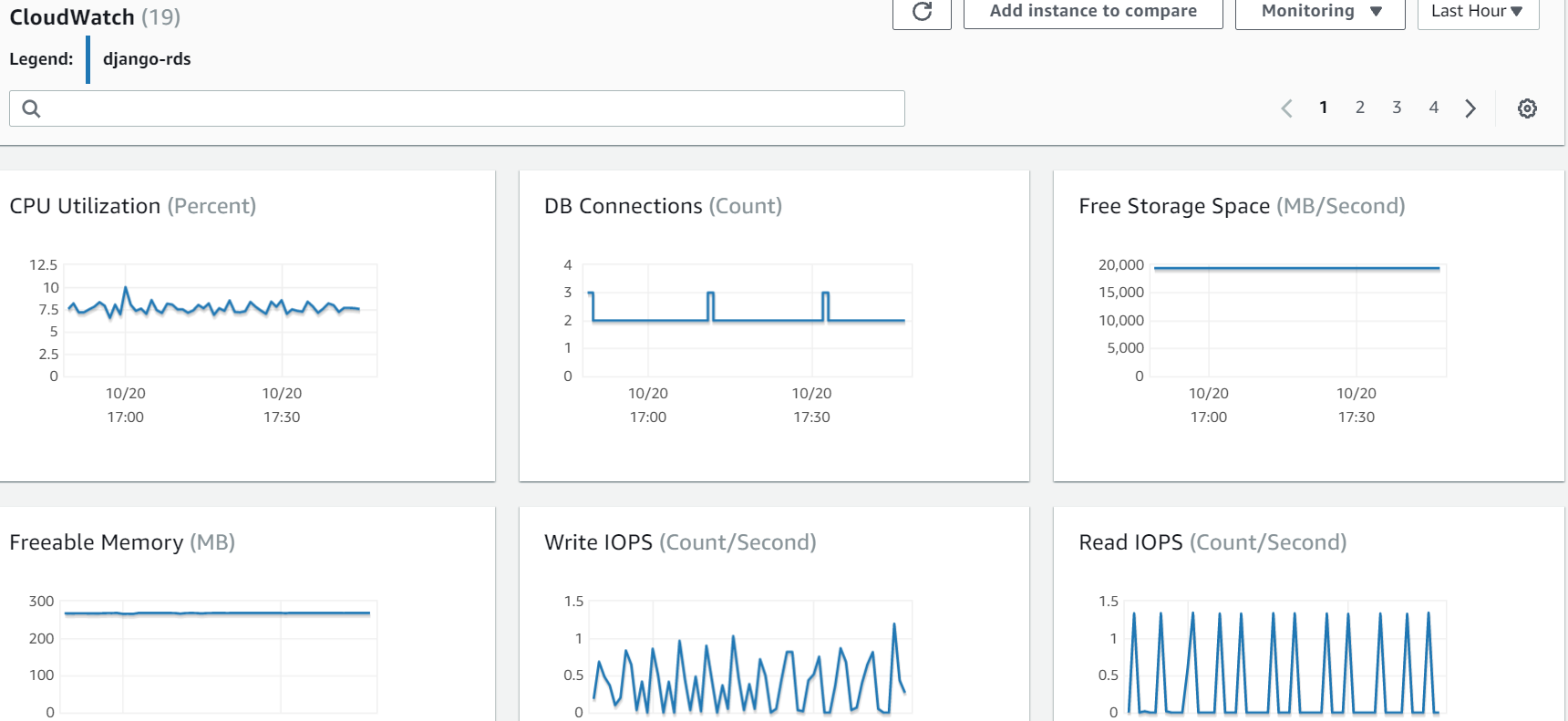
* **AWS RDS: “django-rds”**



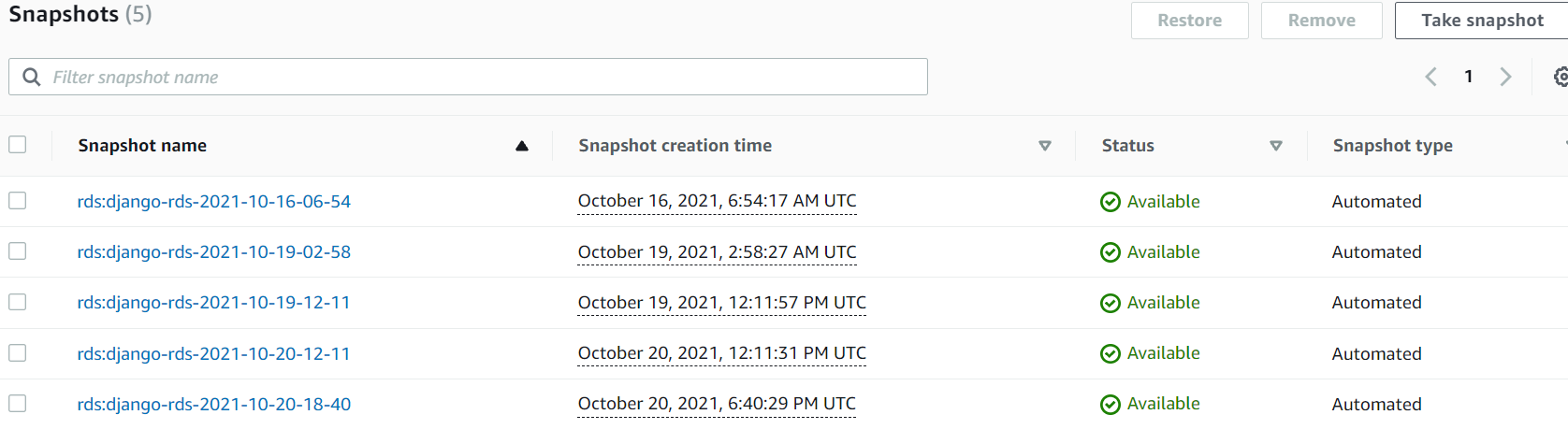
To create a multi-AZ and multi-region architecture, for enhanced availability and durability a replica of primary db instance is created in a different AWS Region.



CloudWatch for RDS helps in monitoring metrics of the db instance.

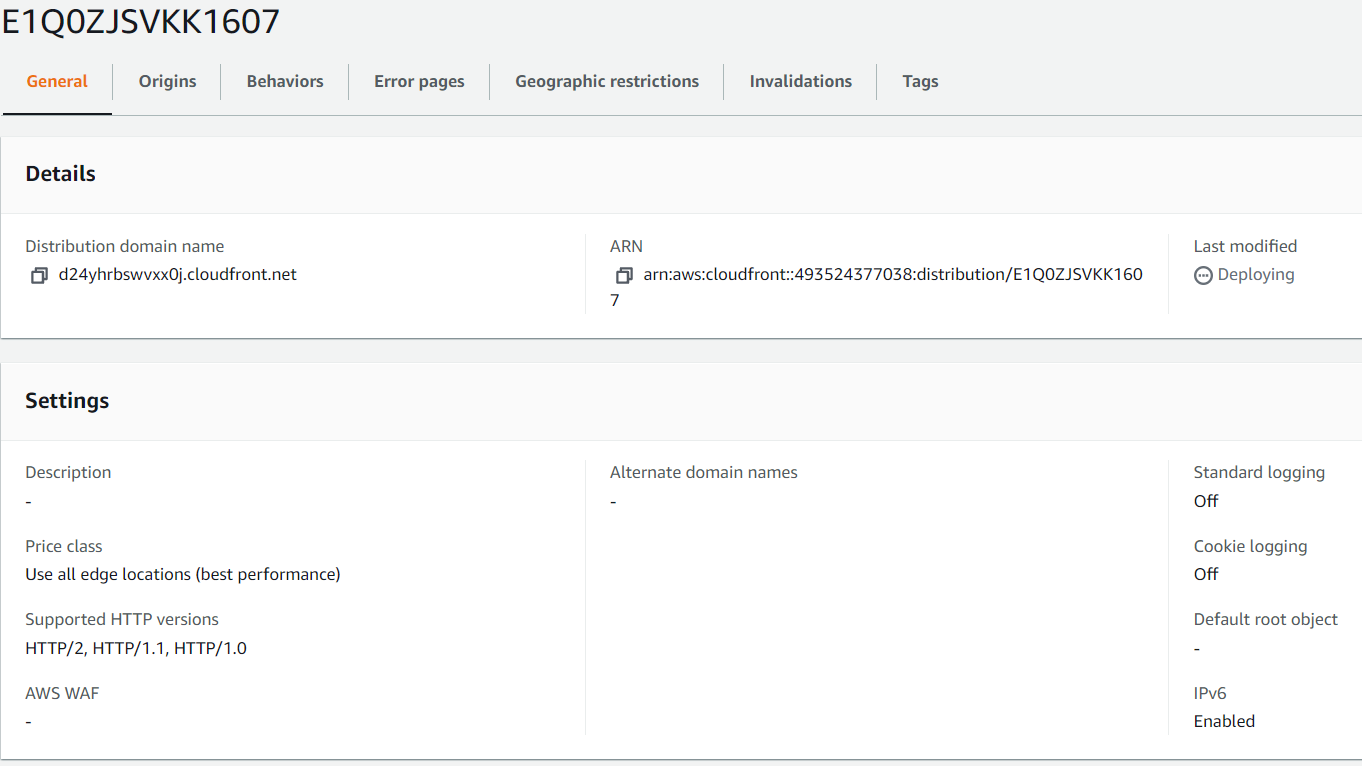


Storage volume snapshot of db instance and backup of entire database.

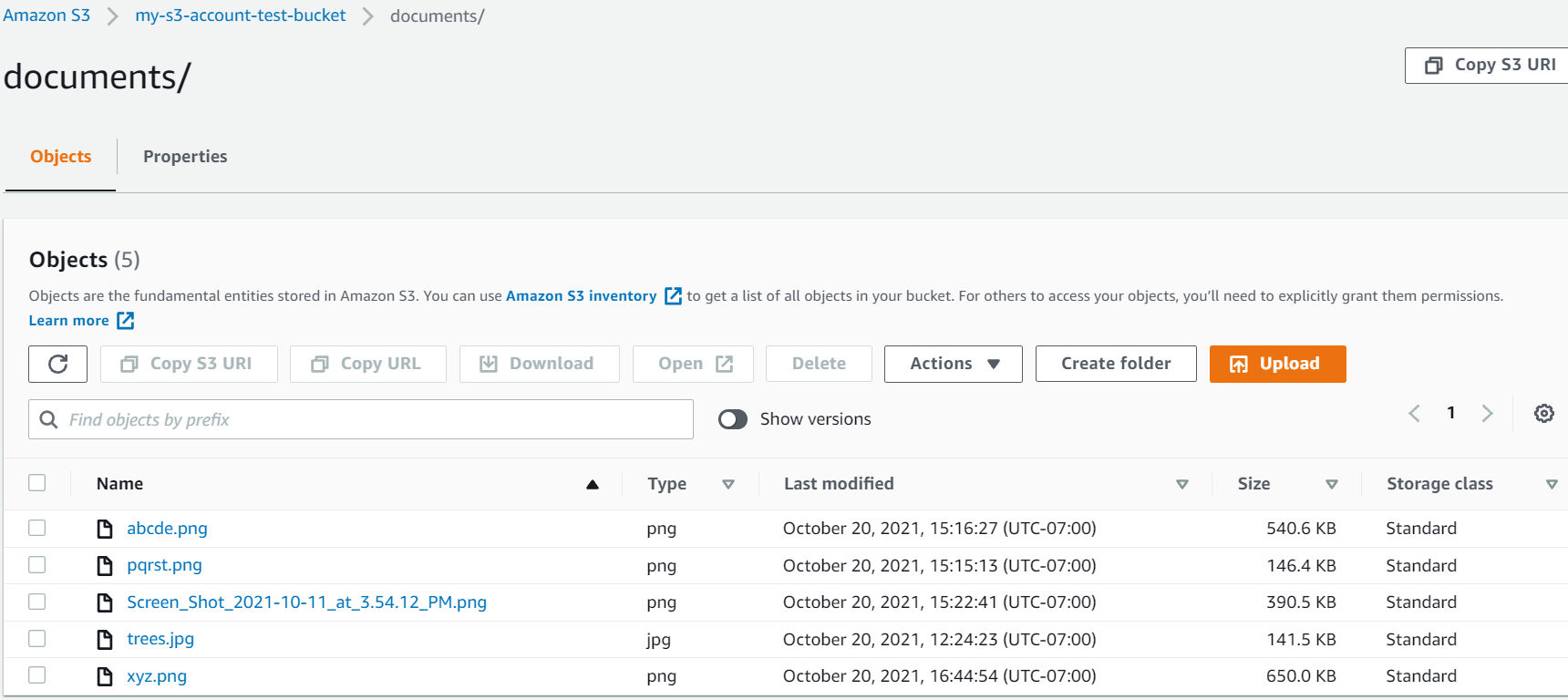


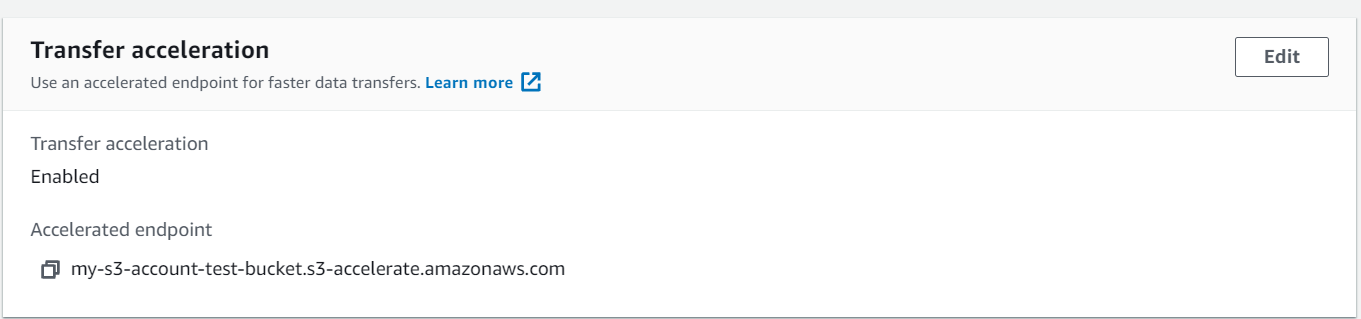
* **AWS CloudFront:**

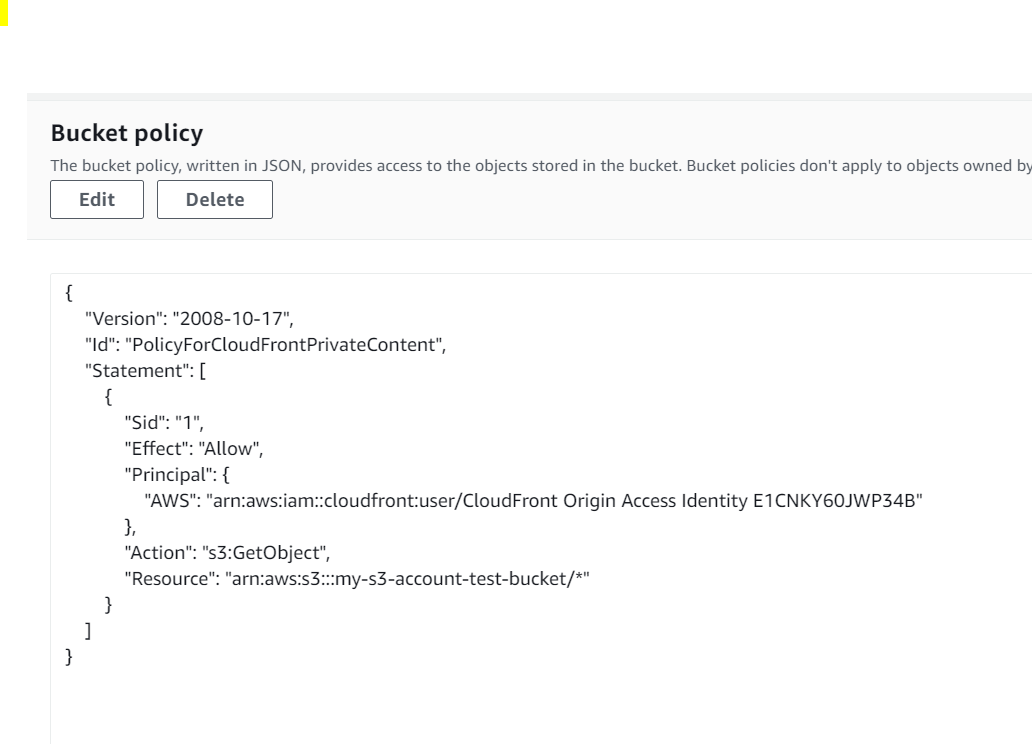
distribution\_domain\_name: https://d24yhrbswvxx0j.cloudfront.net



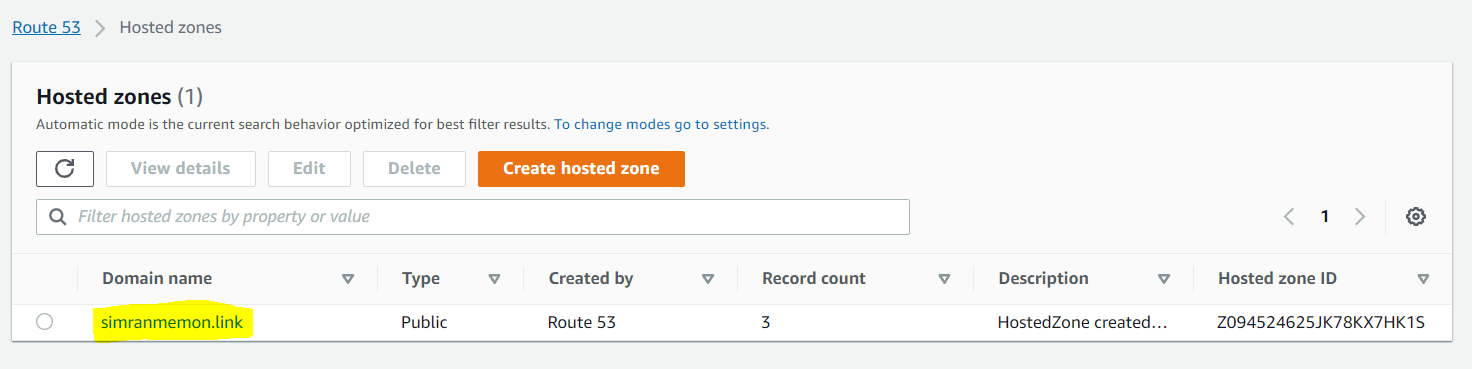
* **AWS S3:** bucket\_name:”my-s3-account-test-bucket” and also the S3 Transfer Acceleration is enabled for 500x performance of the application.



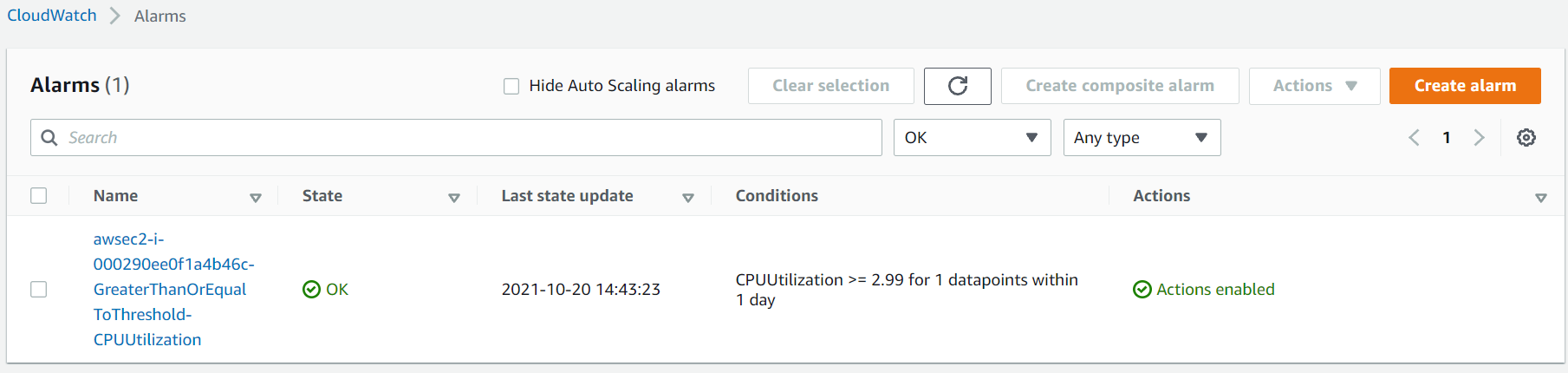




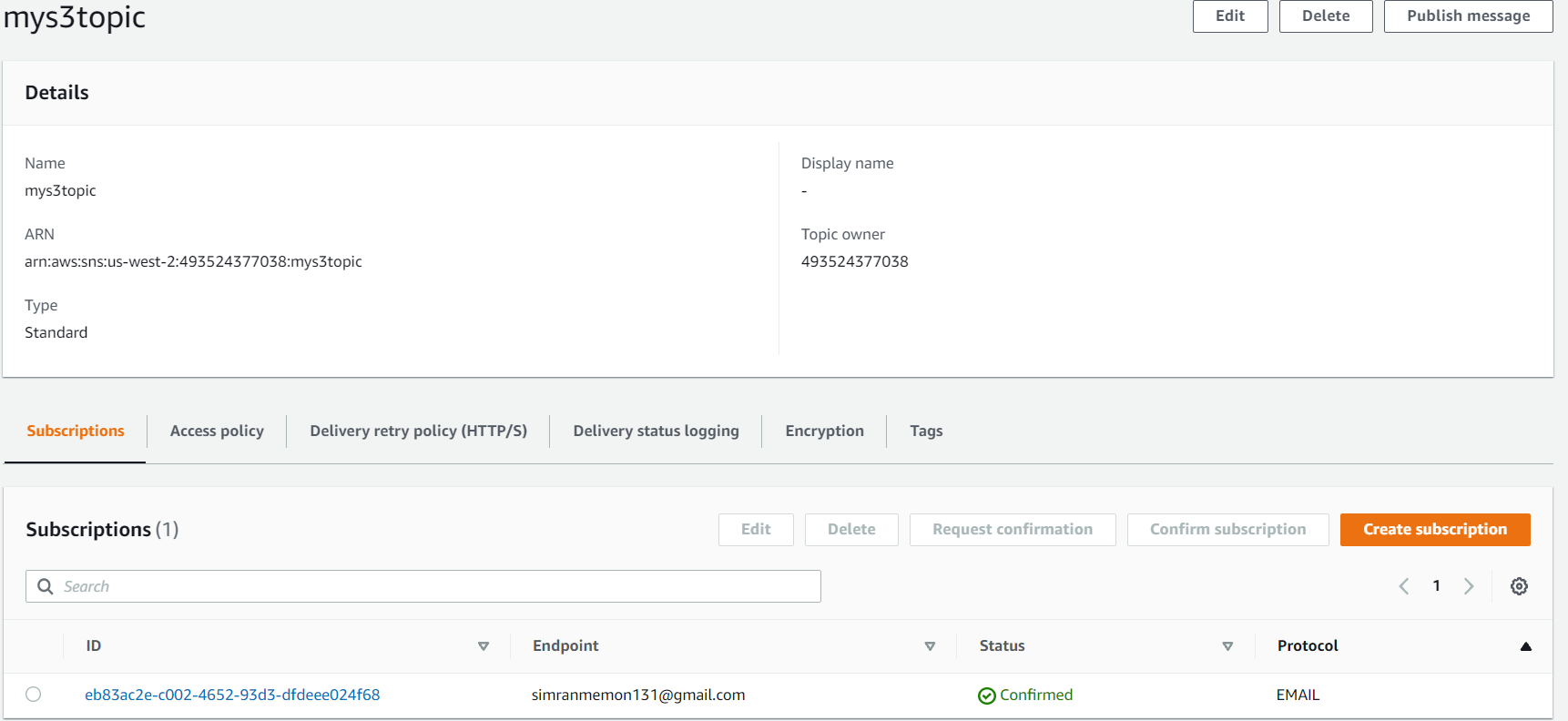
* **AWS Route 53 :** “simranmemon.link”



* **AWS CloudWatch:** Metrics monitoring and alarms are set so that we can be notified if our application enters into an ‘unhealthy’ state.



* **AWS SNS:** topic: ”mys3topic" and subscription: “[eb83ac2e-c002-4652-93d3-dfdeee024f68](https://us-west-2.console.aws.amazon.com/sns/v3/home?region=us-west-2#/subscription/arn:aws:sns:us-west-2:493524377038:mys3topic:eb83ac2e-c002-4652-93d3-dfdeee024f68)”



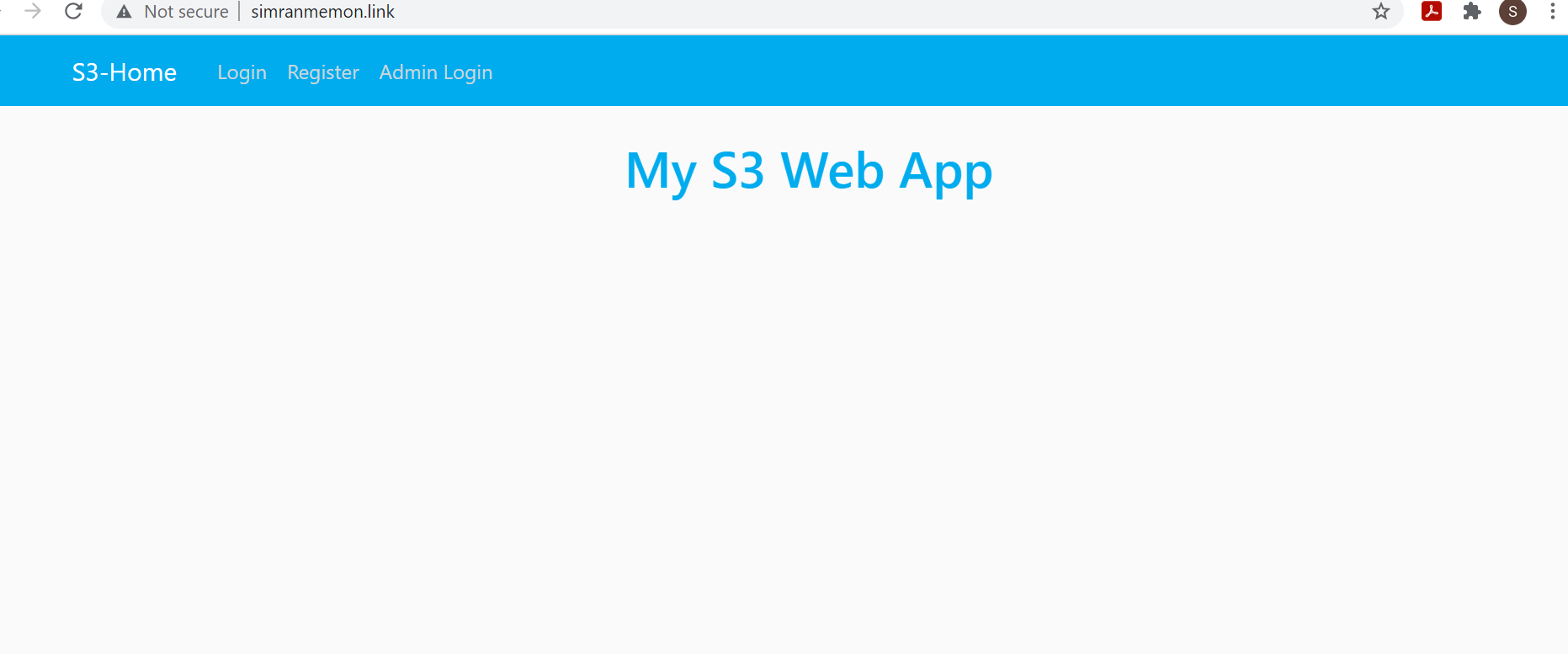
* **Screenshots of WebApp UI functioning:**

1. **User Registration:**

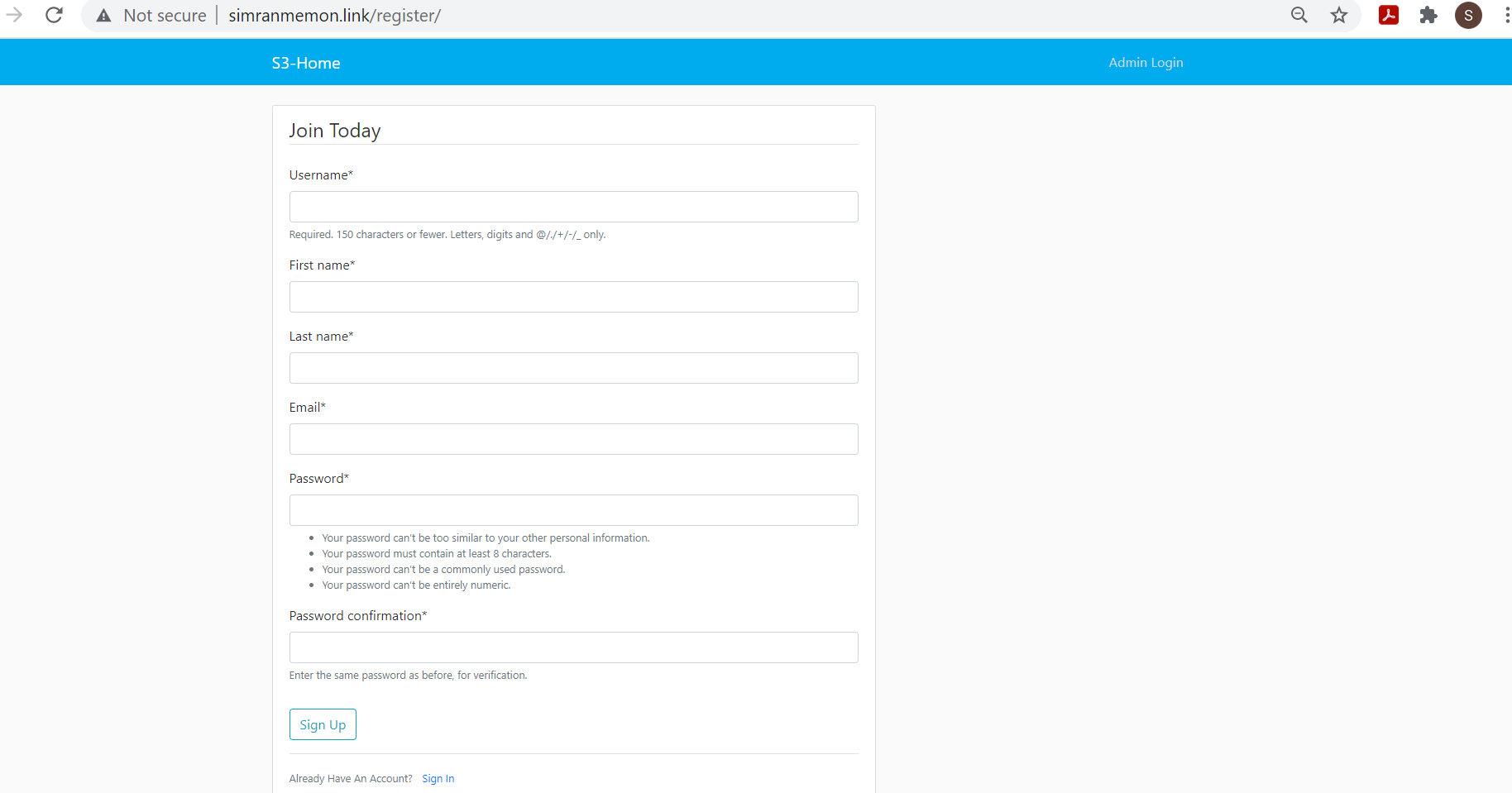
* This feature allows the user to enter his first name, last name, emailid, username and password. After registration, the user will be granted access to use the webapp and will be asked to login with the same registered credentials.

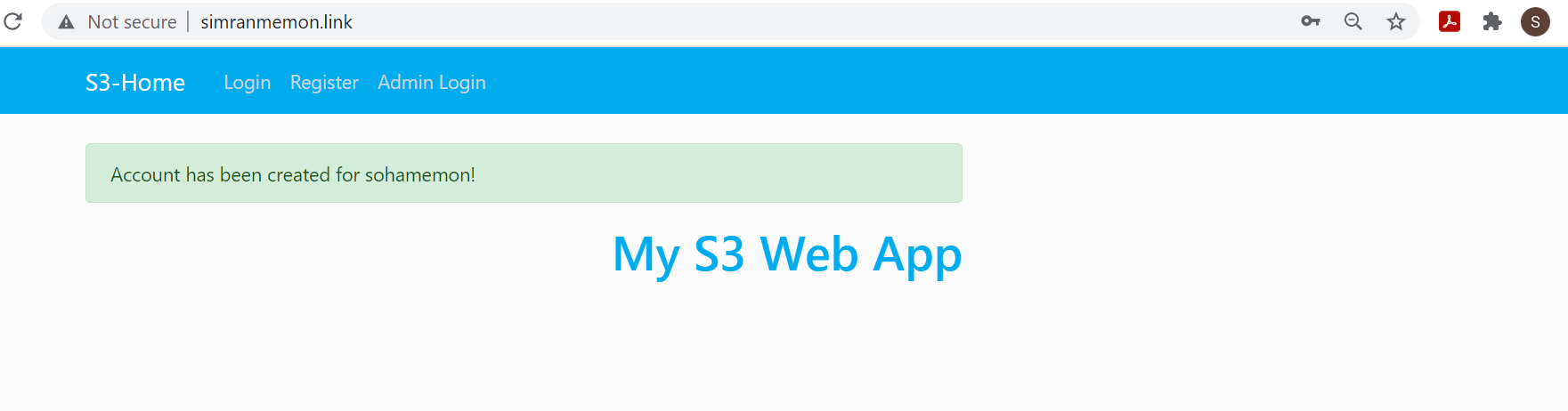
PFA screenshots for the same:

* Home page of the app



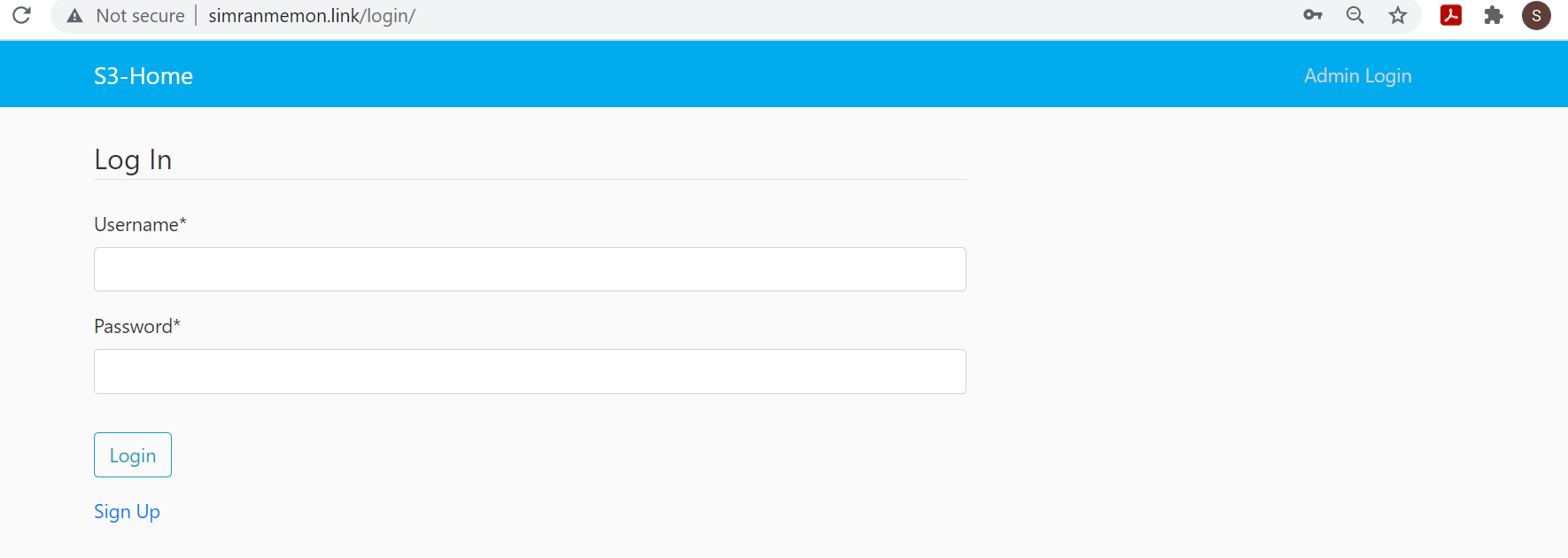
* User sign up page:



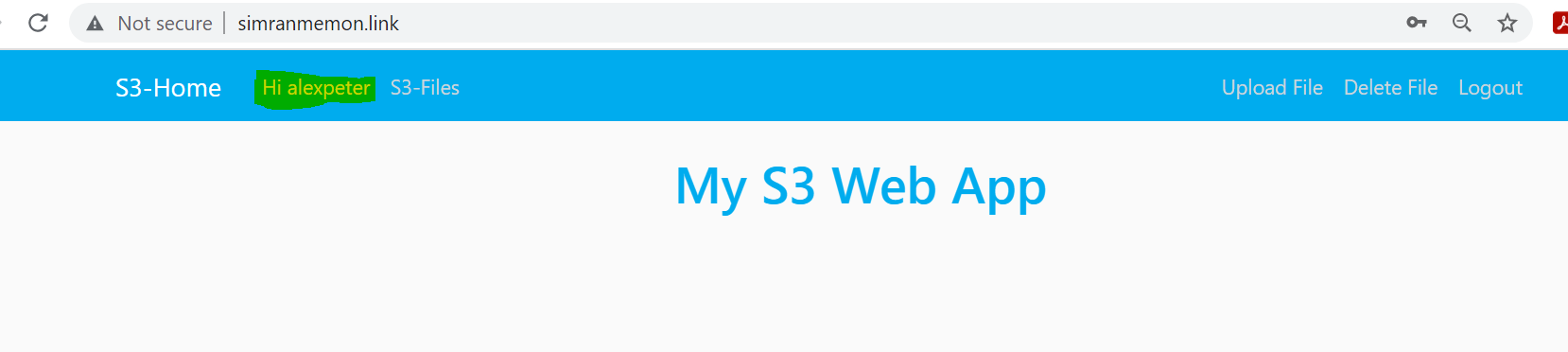


1. **Custom Login:**

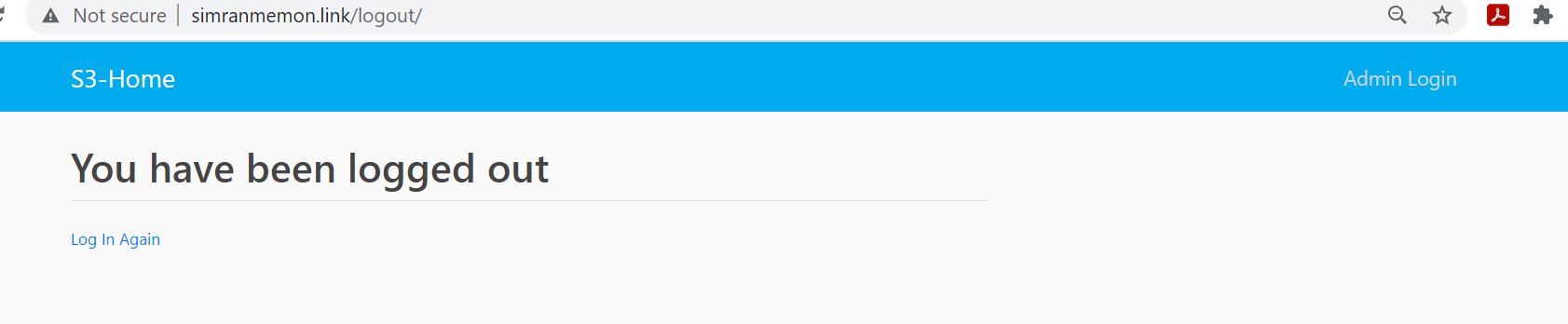
* After completing the registration process, the user will be able to login will his credentials. The user will then be provided access to the webapp with the ability to view the files that he has uploaded, upload new files, update already existing files and delete any file of choice.



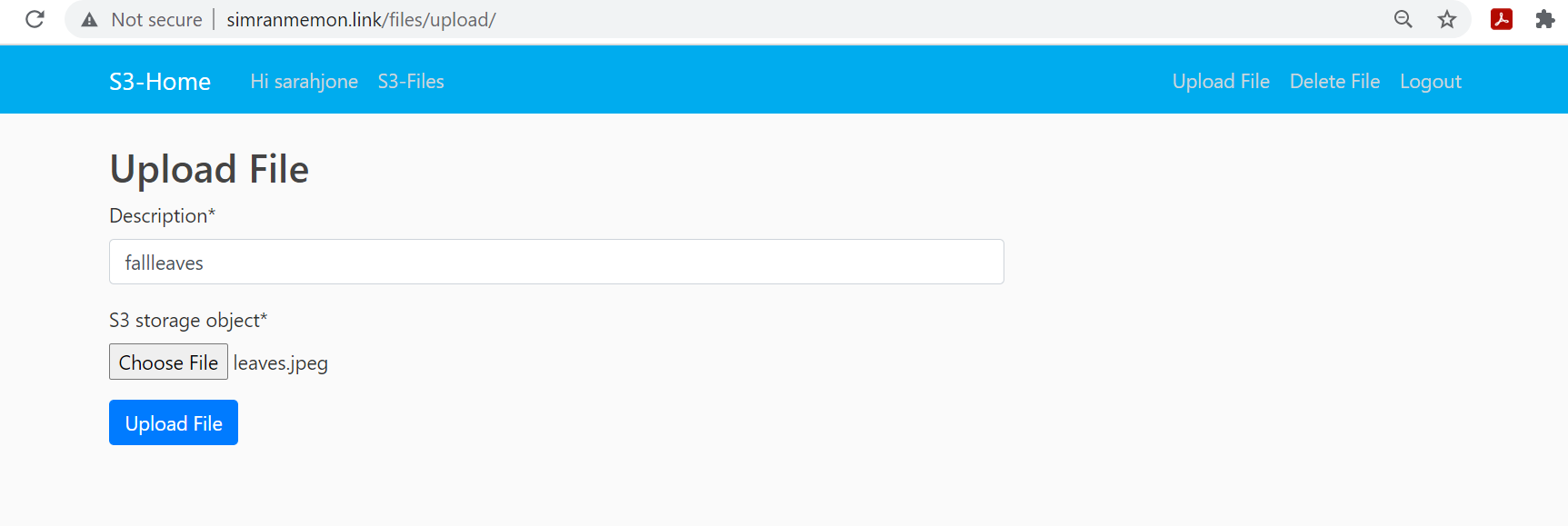
Successful user login:



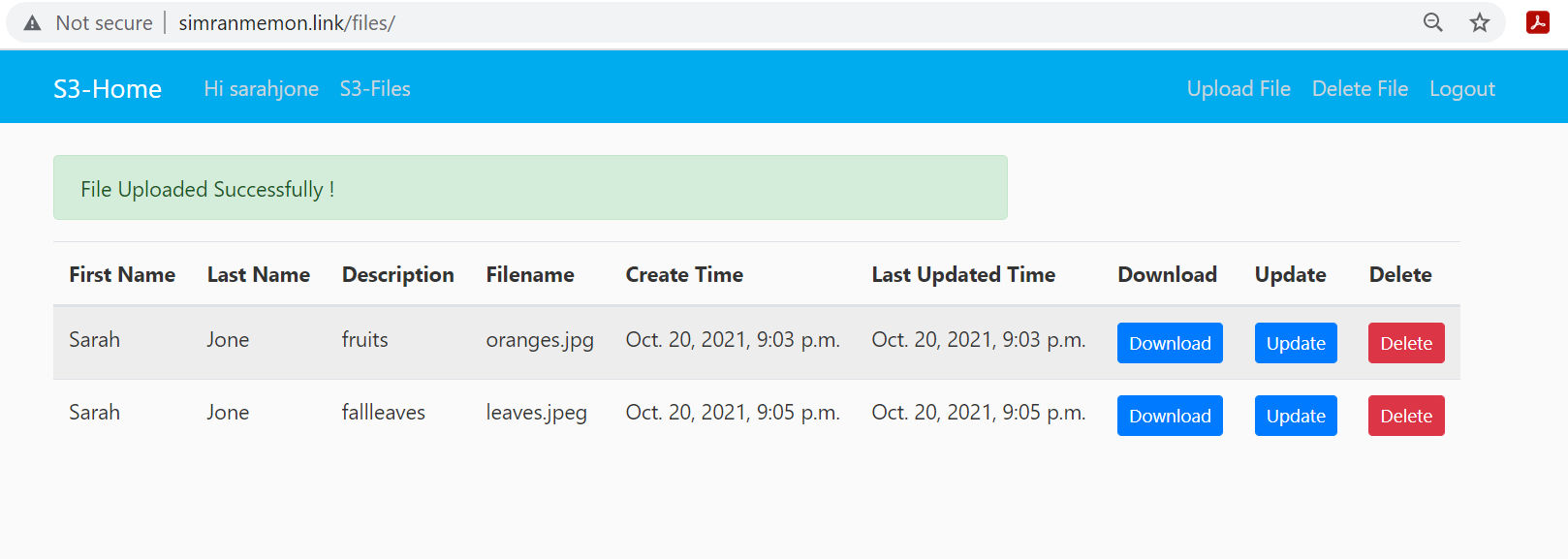
User will also be able to successfully logout of the app:



1. **File Upload:** Through web UI, the file can be uploaded by registered users of max size 10mb. Along with a short description of the file.

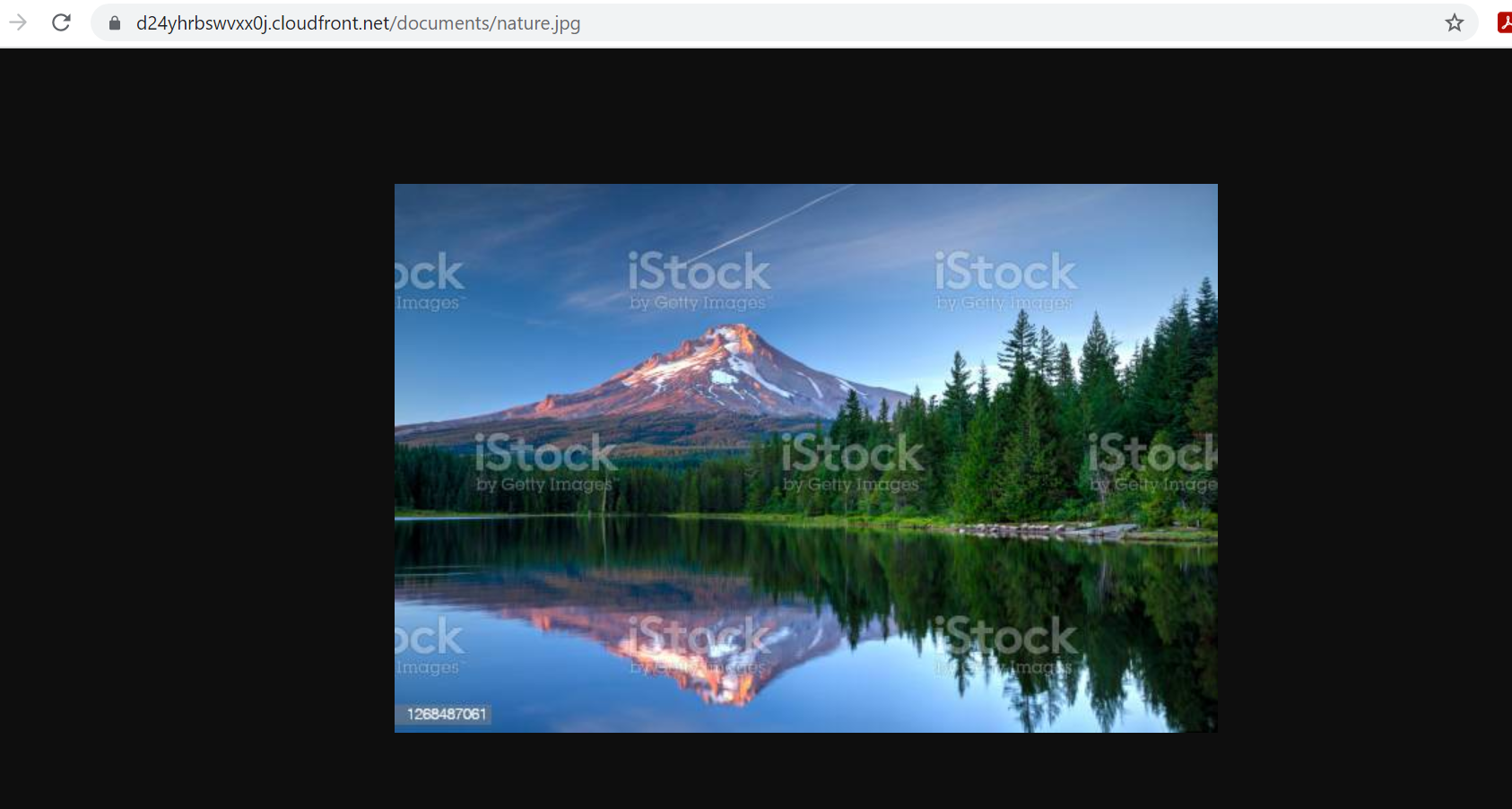


file uploaded by user successfully along with the user and file details:



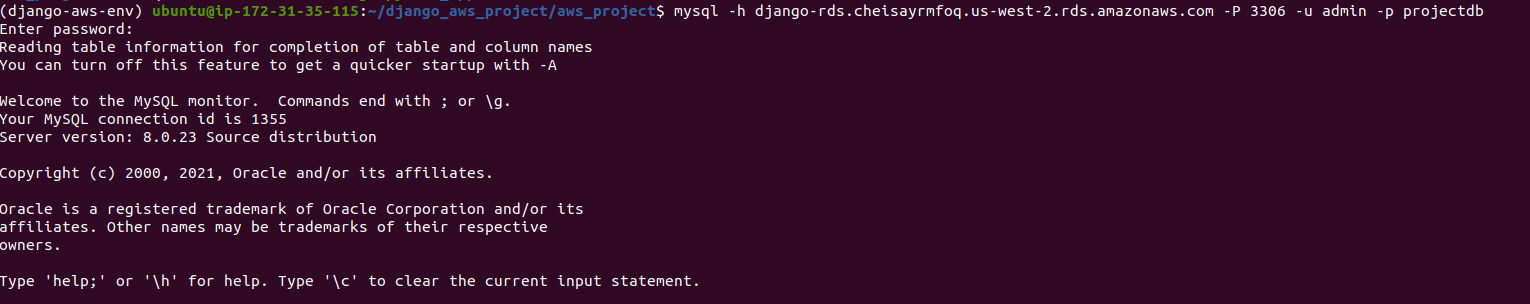
1. **File Download: (check the S3-Files tab for this feature)**Thisfeature will provide the user with an ability to scroll though already uploaded list of files and to download the file through CloudFront service. It will also display details of every file upload done such as : user’s first name, user’s last name, file upload time, file updated time, file description.

file can be downloaded by “save image as”, and the download feature leverages CloudFront.

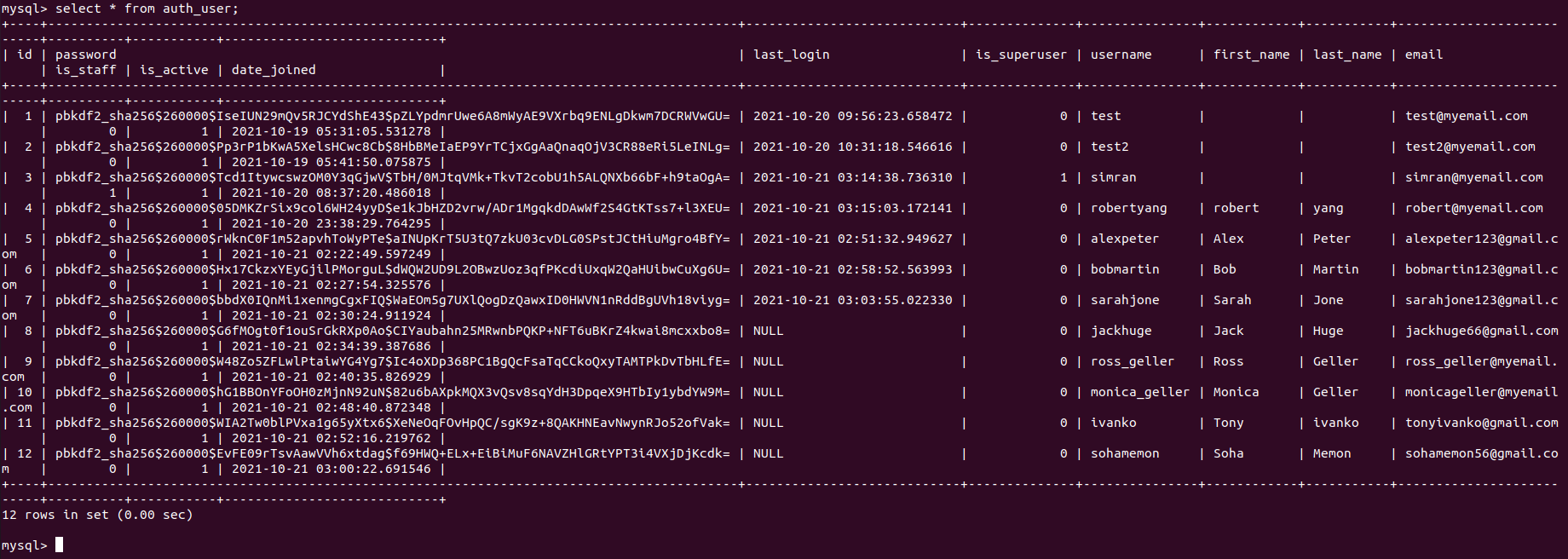


1. **Database Updates:** The database will track details of every file upload done by every user. It will track details regarding file such as: user’s first name, user’s last name, file upload time, file updated time, file description. The user credentials will also be saved to the database.

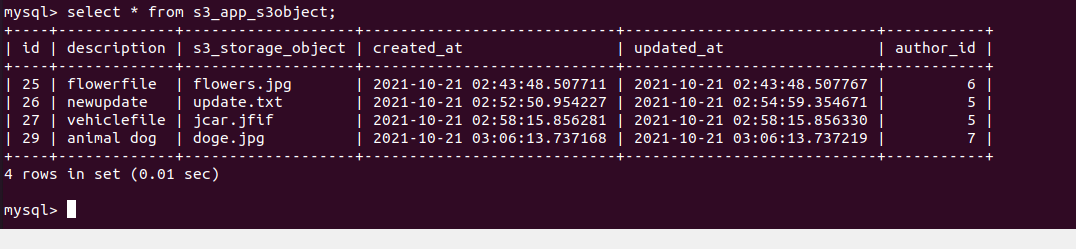
* For connection: mysql -h django-rds.cheisayrmfoq.us-west-2.rds.amazonaws.com -P 3306 -u admin -p projectdb



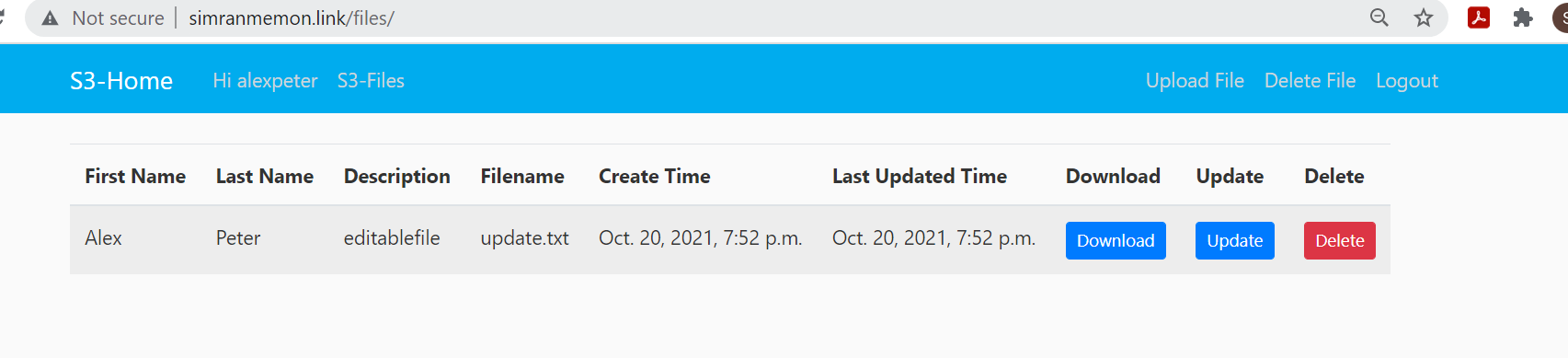
* The table: auth\_user consists of details of the user’s and the admin.

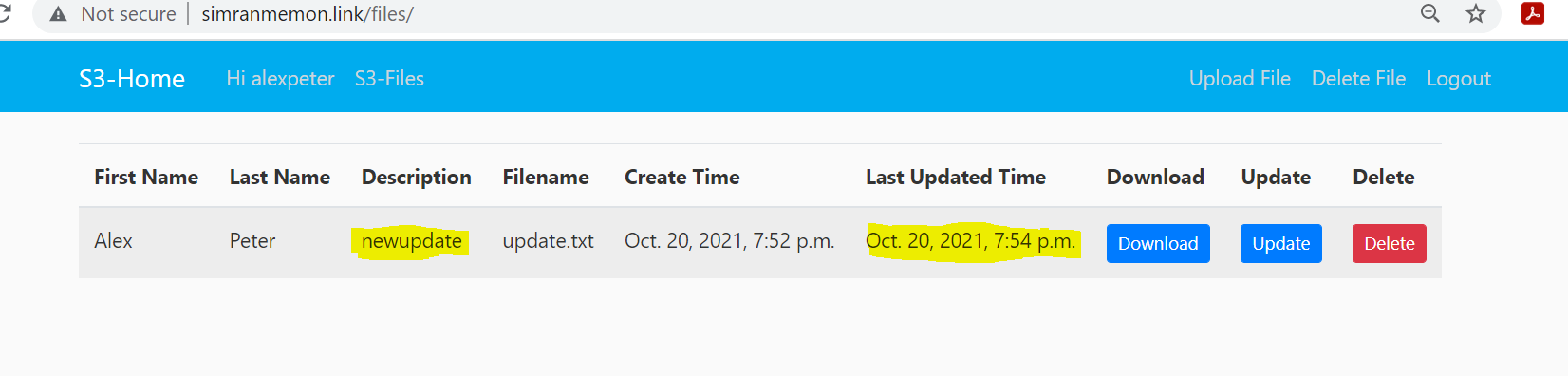


The author\_id column is the foreign key, mapped to id column from table: auth\_user. This way the file details are tracked for every user.

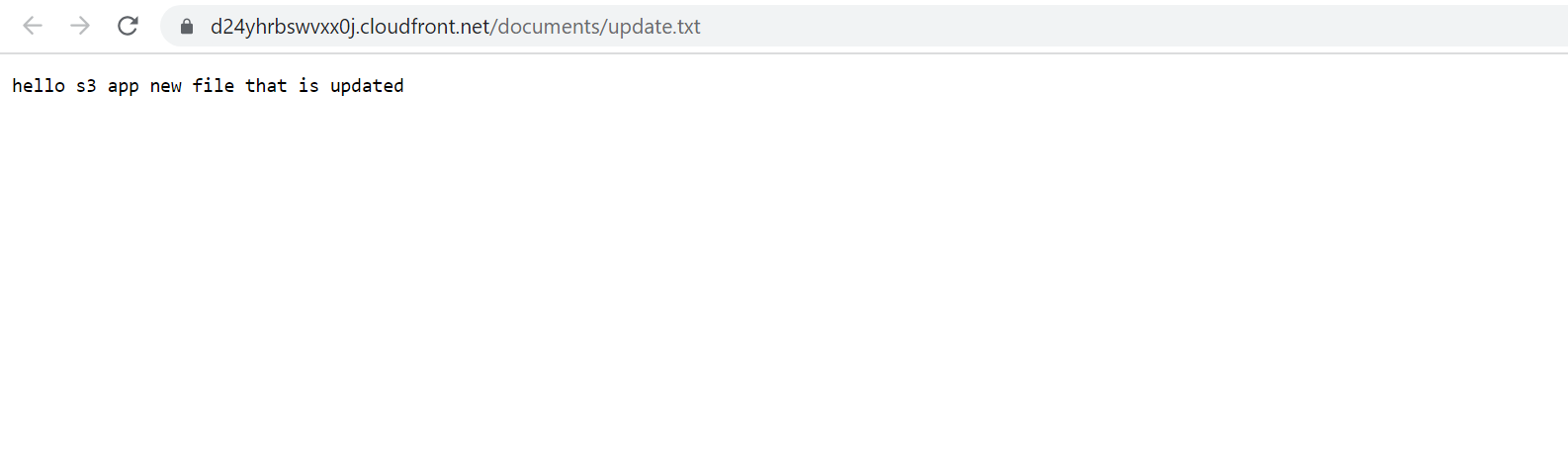


1. **File Edit:** A file can be downloaded via CloudFront and then can be edited and written back to S3. The update feature on the webapp will help achieve this.

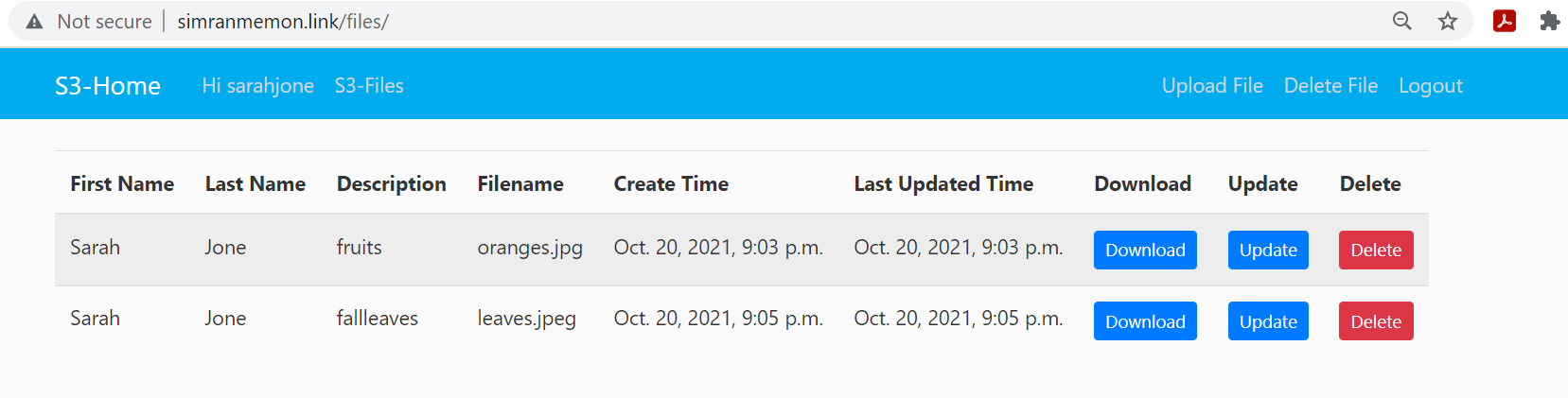




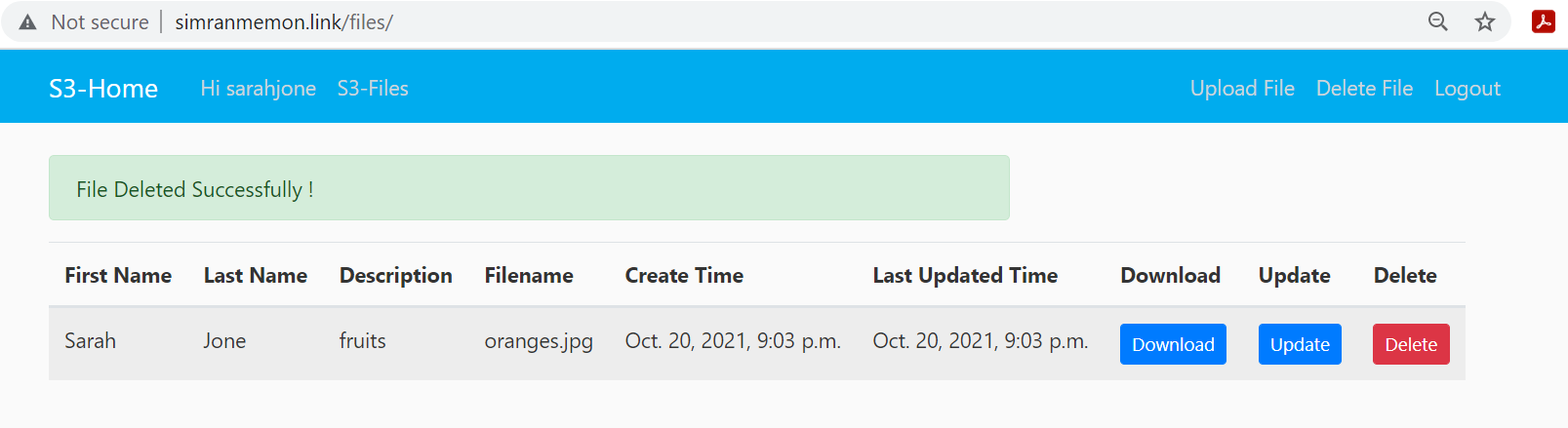
File updated successfully:



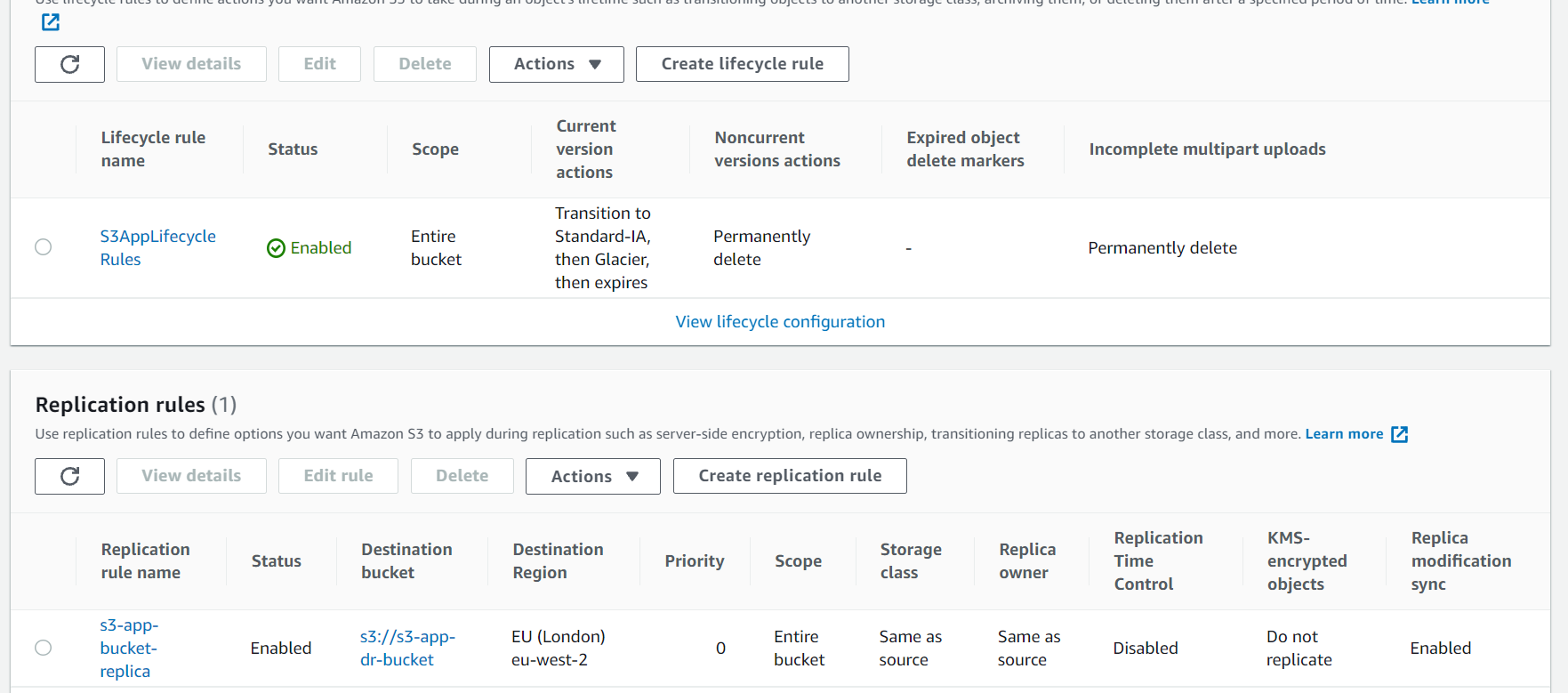
1. **File Delete:** Everyuser will be able to delete his own file only. He can upload files to s3 bucket and delete only his own specific files. He won’t be allowed to delete files of other users as the files uploaded by other users are not accessible to him. Once the delete action is triggered, an email will be sent to Admin that the file has been deleted.



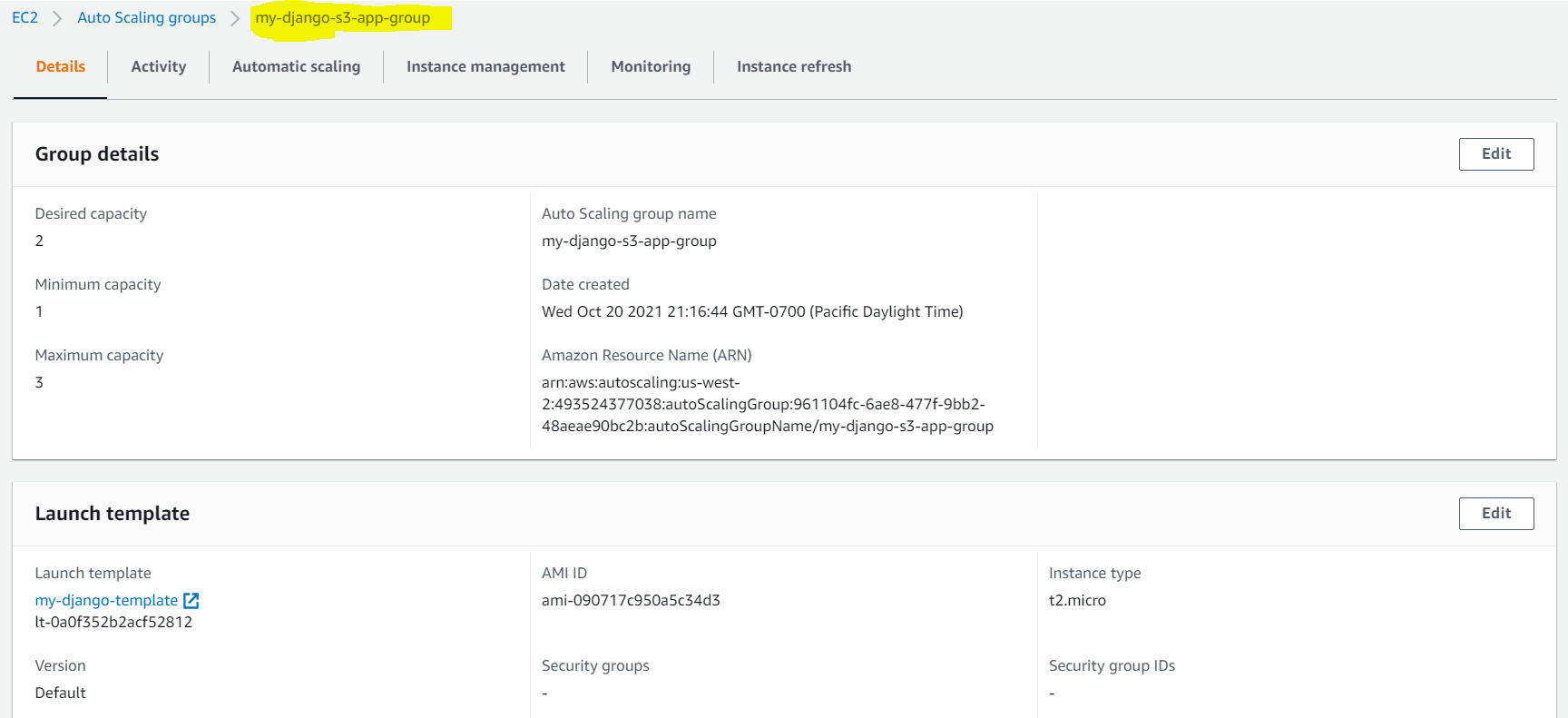
After deletion:

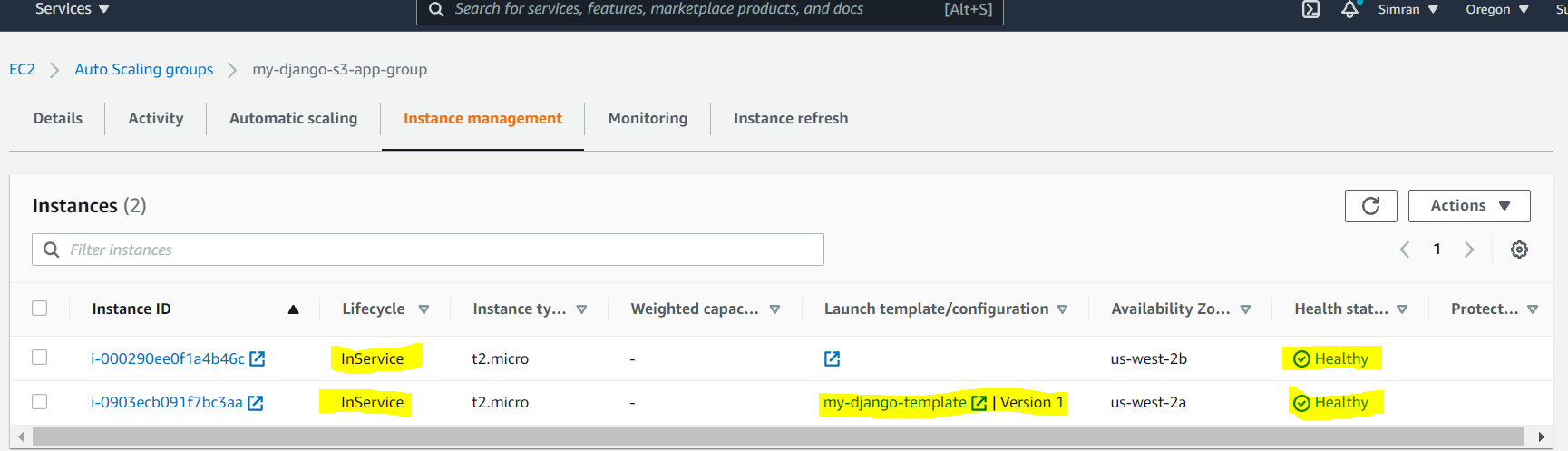


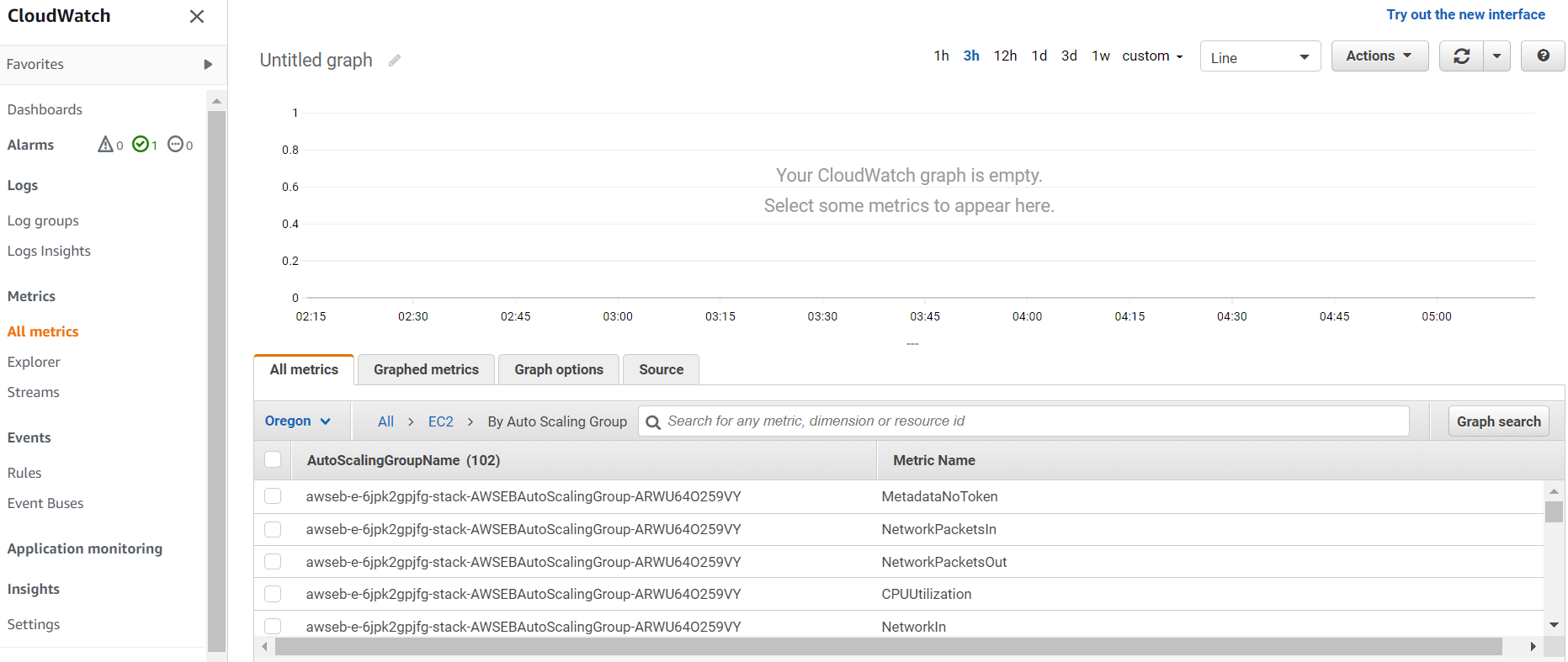
1. **DR Measures:** Data from sourcebucket – “my-s3-account-test-bucket”is replicated in destination bucket – “s3-app-dr-bucket” using **S3 Cross region replication (S3 CRR).** This is done for maximum durability and availability of data even during times of failures.



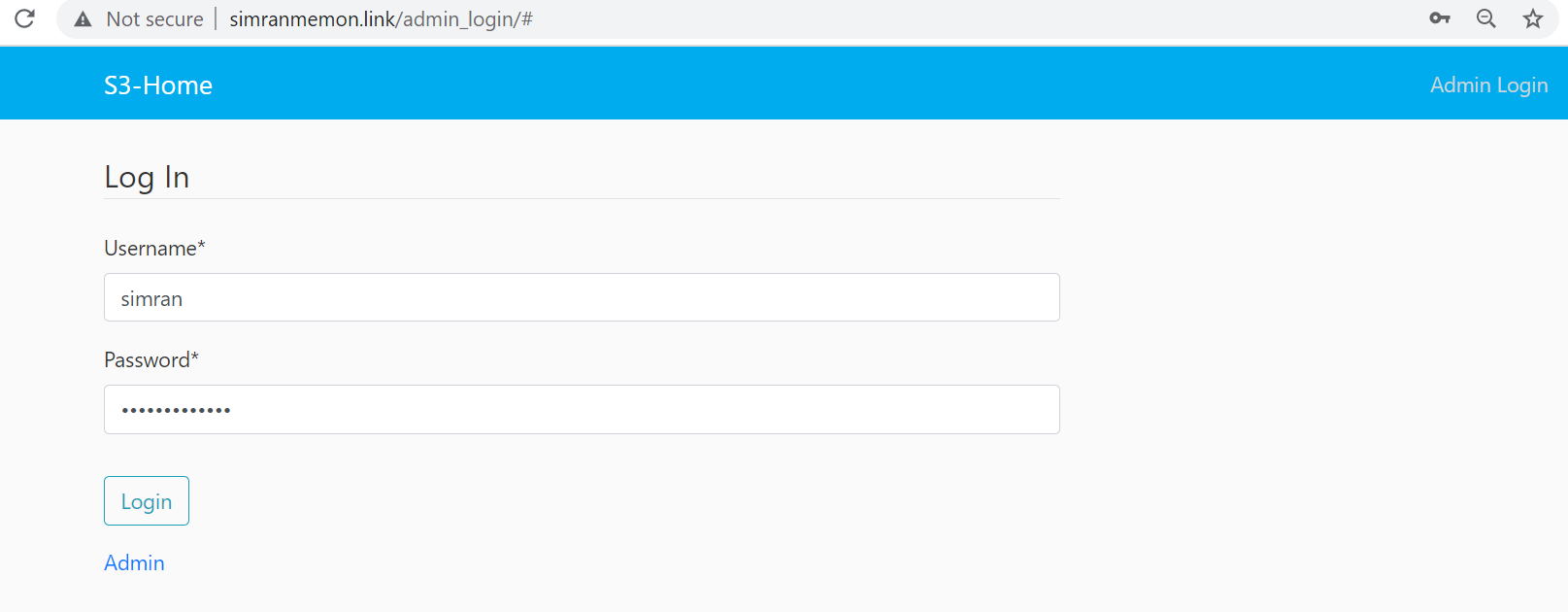
1. **Autoscaling Group:** Auto scaling Group is created to help ensure that the application always has the right amount of capacity to handle the traffic demand. It also helps in detecting unhealthy instances and terminates them if required. It is configured in multi-Availability Zones for better durability and performance.



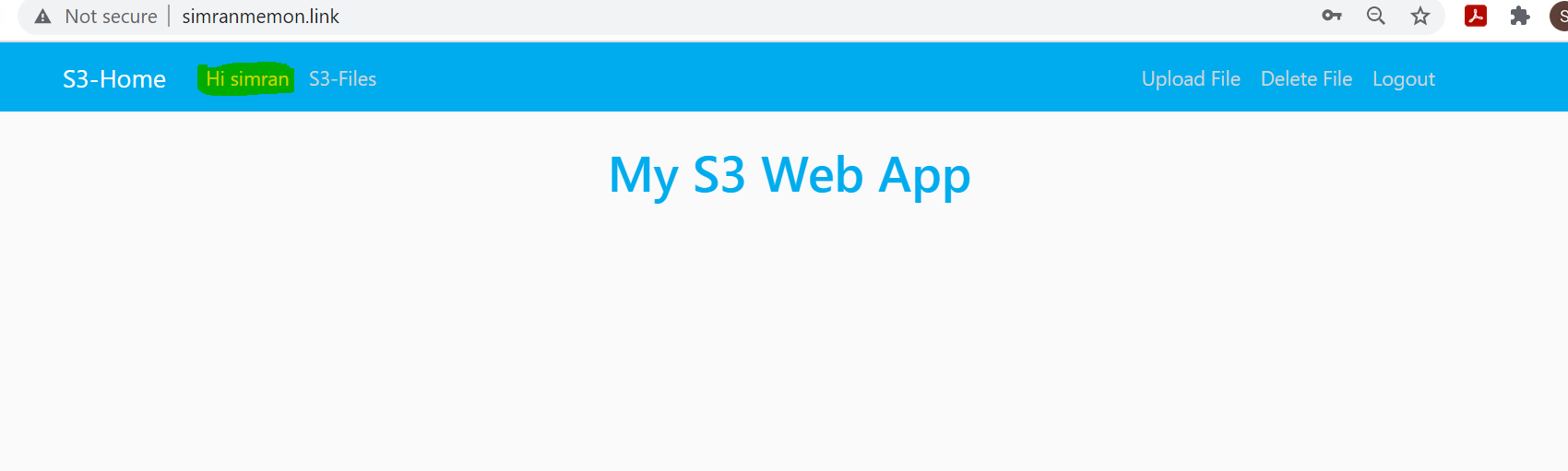




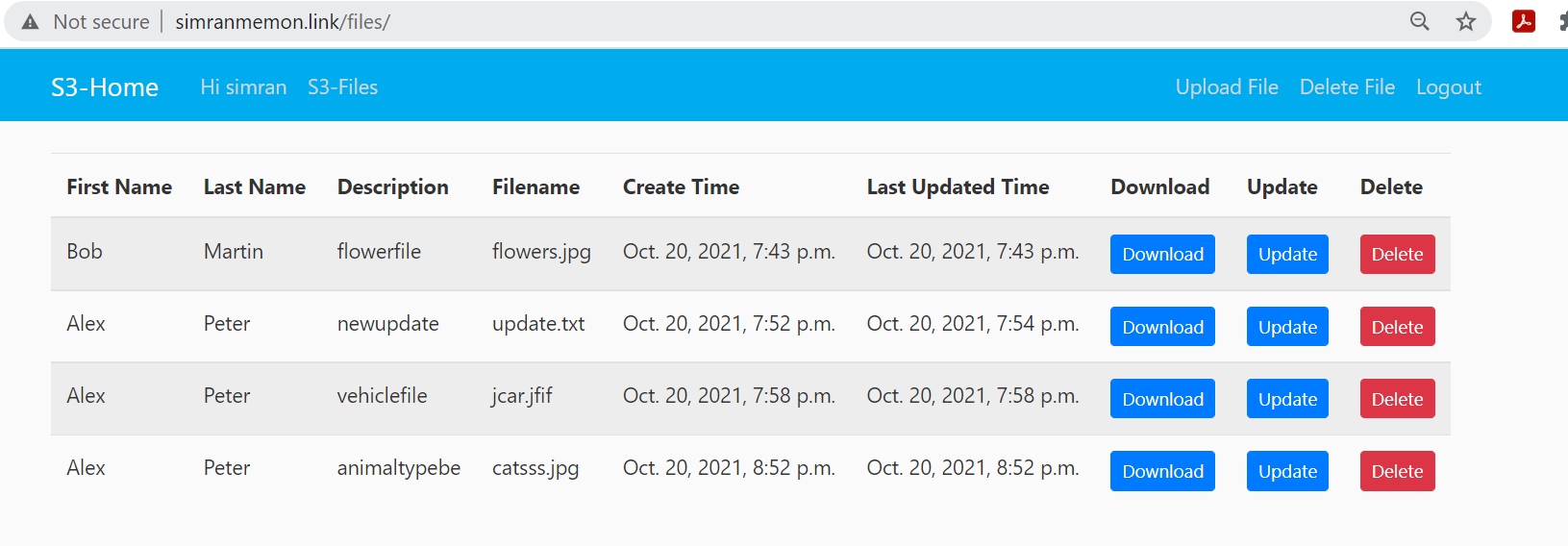
1. **Admin Panel:** An admin is created which will have access to all the uploaded files by all the users, with an ability to delete files of any user. Also, whenever there is a delete performed on a file, a lambda event is triggered and it sends an email via AWS SNS service to the admin only.

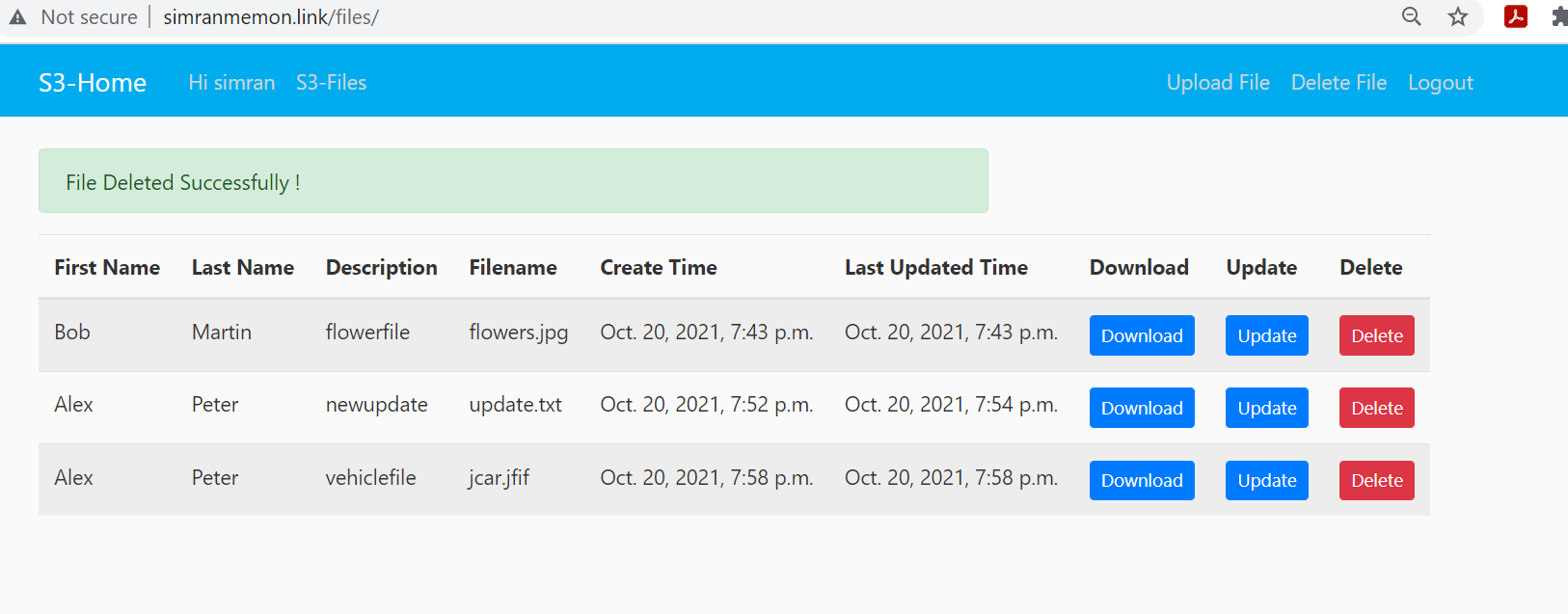


Admin login successful:

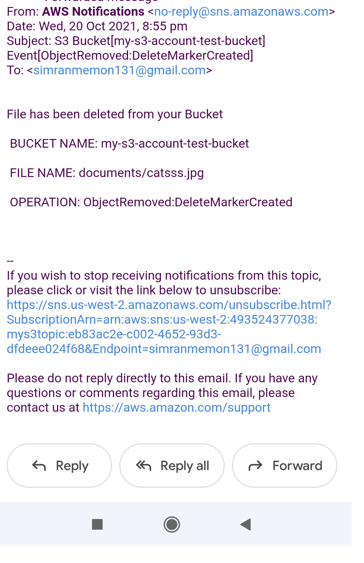


Admin has access to all the files and can delete any file:

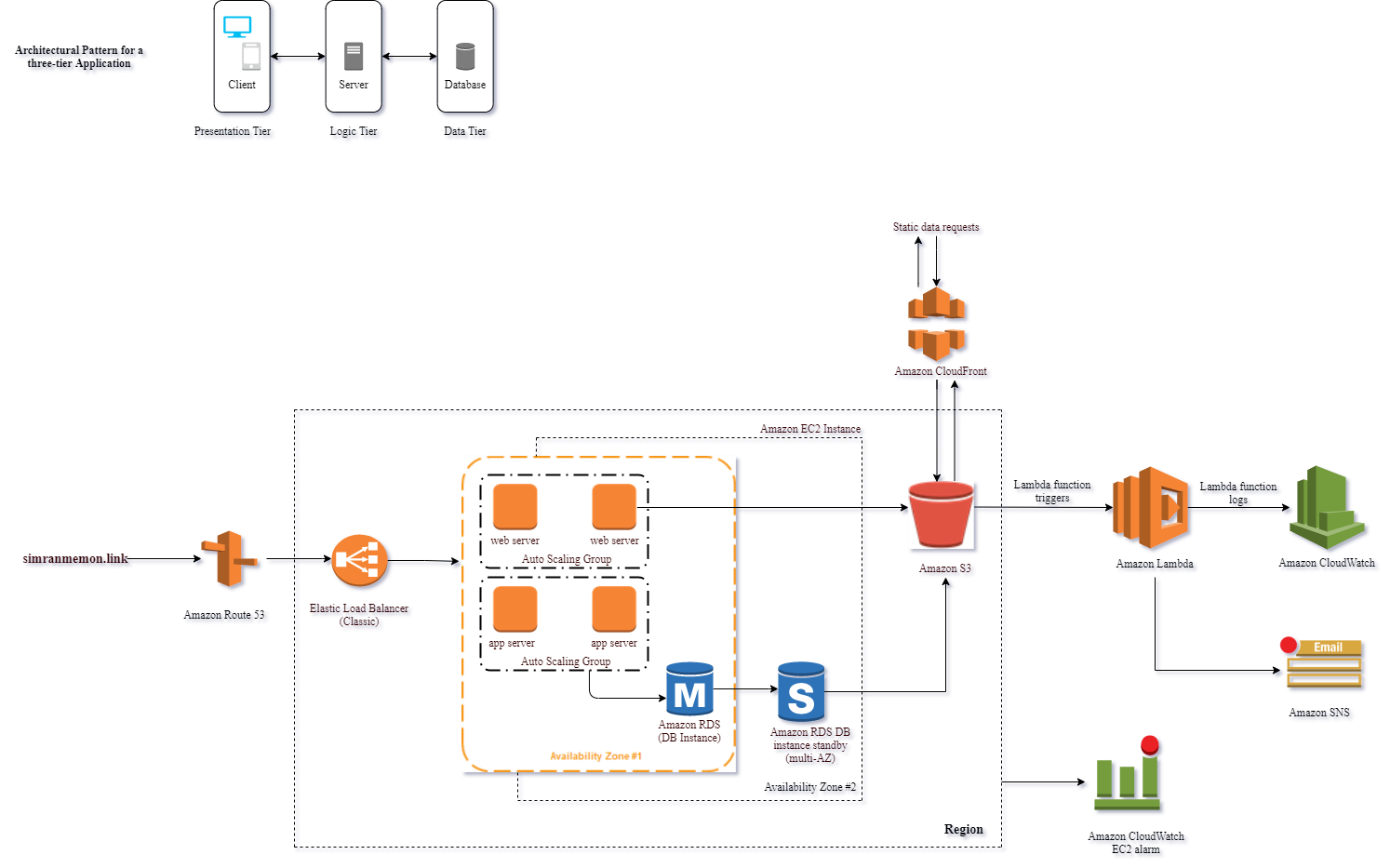




Message received on admin’s email\_id that the file has been deleted:



* **Architectural Diagram of the Solution:**



* **References:**

1.  <https://docs.djangoproject.com/en/3.2/>

2. <https://www.youtube.com/watch?v=UmljXZIypDc&list=PL-osiE80TeTtoQCKZ03TU5fNfx2UY6U4p&ab_channel=CoreySchafer>

3. <https://medium.com/saarthi-ai/ec2apachedjango-838e3f6014ab>

4. <https://docs.aws.amazon.com/autoscaling/ec2/userguide/GettingStartedTutorial.html#gs-walkthrough-summary>

5. <https://docs.aws.amazon.com/lambda/latest/dg/lambda-python.html>

6. <https://simpleisbetterthancomplex.com/series/2017/09/04/a-complete-beginners-guide-to-django-part-1.html>

7.<https://django-storages.readthedocs.io/en/latest/backends/amazon-S3.html>

8. <https://medium.com/analytics-vidhya/how-to-access-aws-s3-using-boto3-python-sdk-e5fbd3d276bd>