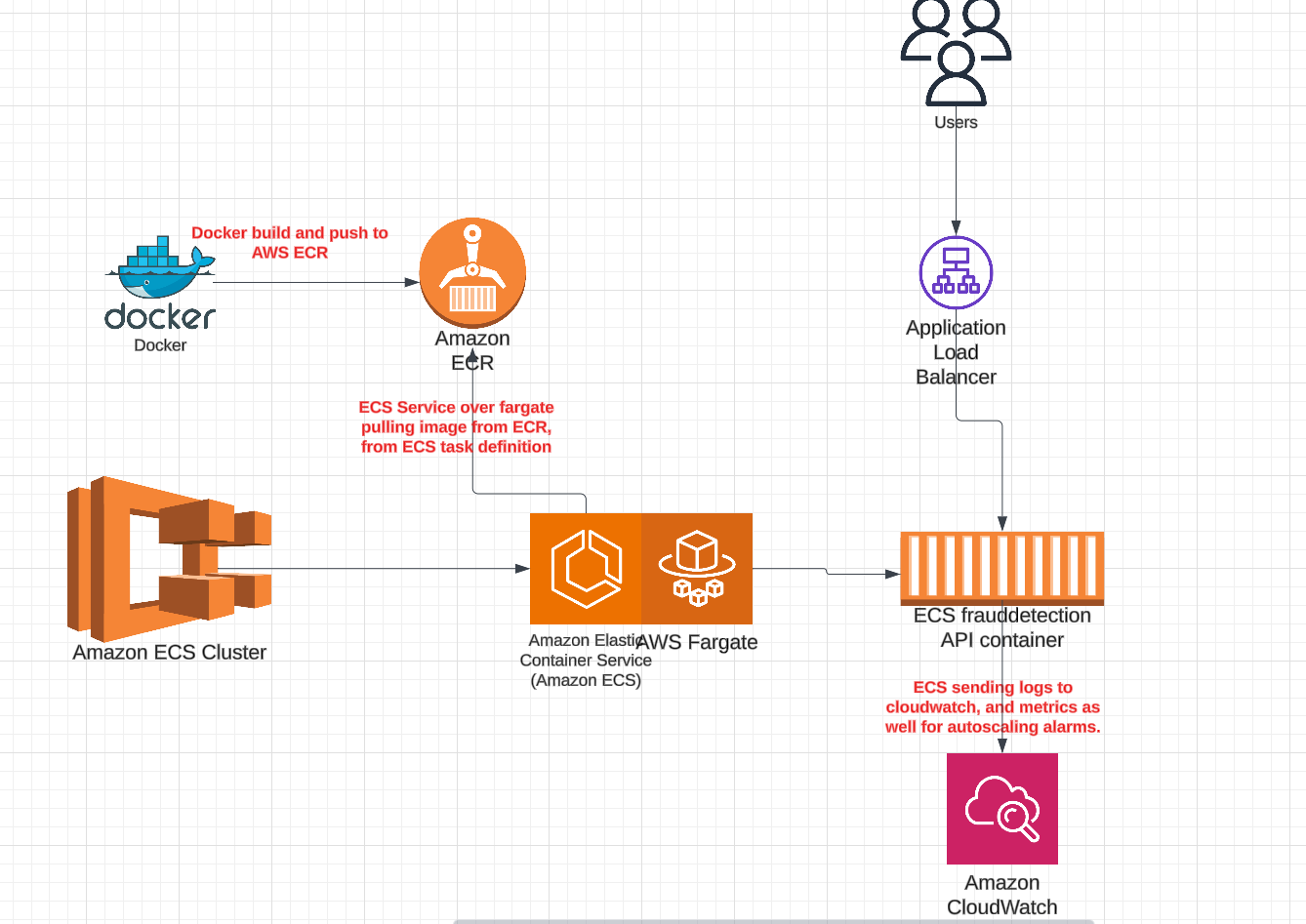
**Fraud Detection API:**

For the fraud detection API, since the request was to Serve as a container orchestration service without managing servers or Kubernetes clusters, ***ECS over fargate has been chosen*** to host the fraud detection API. ECS will integrate easily with Cloudwatch and Cloudtrail, and allow auto scaling as well.

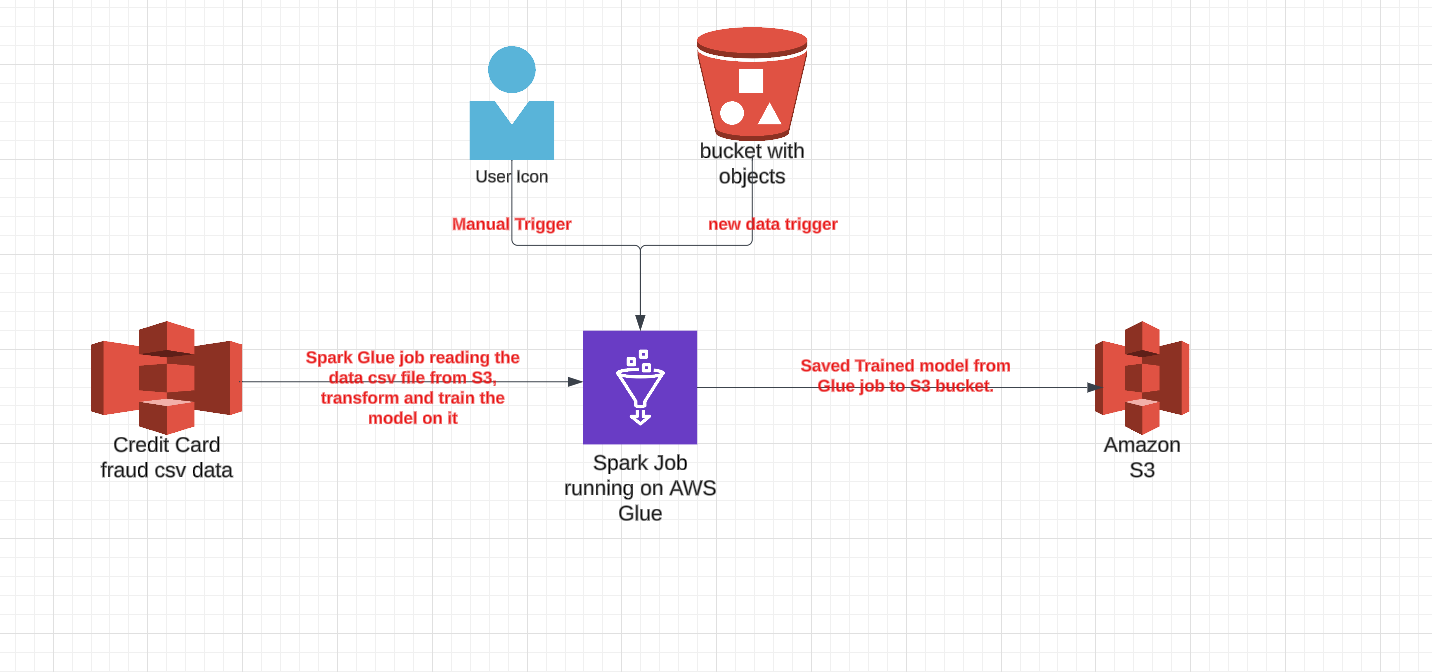
A docker image will be built from the docker file, pushed to **ECR**. A ECS task definition will be created based on that image, and ECS service will be built on top of it, exposed by an **ALB behind a target group**.

Cloudwatch will be used for logging, as well as sending metrics and have a cloudwatch alarm over this metric to use ECS autoscaling.

*Note: An api gateway might be useful to serve api requests, but an ALB exposing the ECS service should be enough for the simple use case for cost optimization.*



**Fraud Detection Pipeline:**



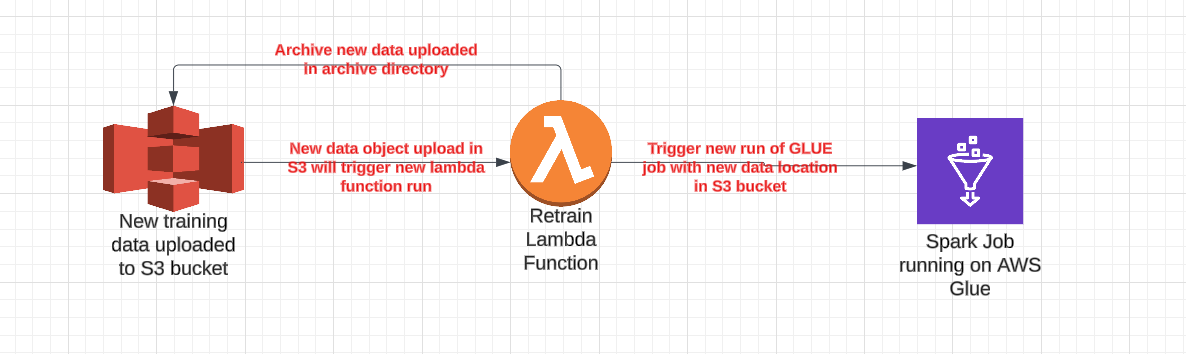
The data pipeline architecture is simpler and straightforward. Since the requirement was to Integrate with other AWS services, is fully managed, Charges only for usage, and Supports future development; **AWS Glue ticks all the boxes.**

All that is really needed is to adapt the code to use the GLUE platform. **S3 bucket** will serve as input for the pipeline, and **another S3** will be used as storage for the resulting model. Pipeline will be served on demand whenever it is triggered whether manually directly through Glue, or when the refresh\_function, which is going to be explained next in this document, whenever it is triggered itself by new data upload in S3.

**Retrain Function:**

Same as data pipeline, the retrain function is even simpler: A simple servless job is needed that needs to be triggered whenever a new dataset is available to retrain the model.

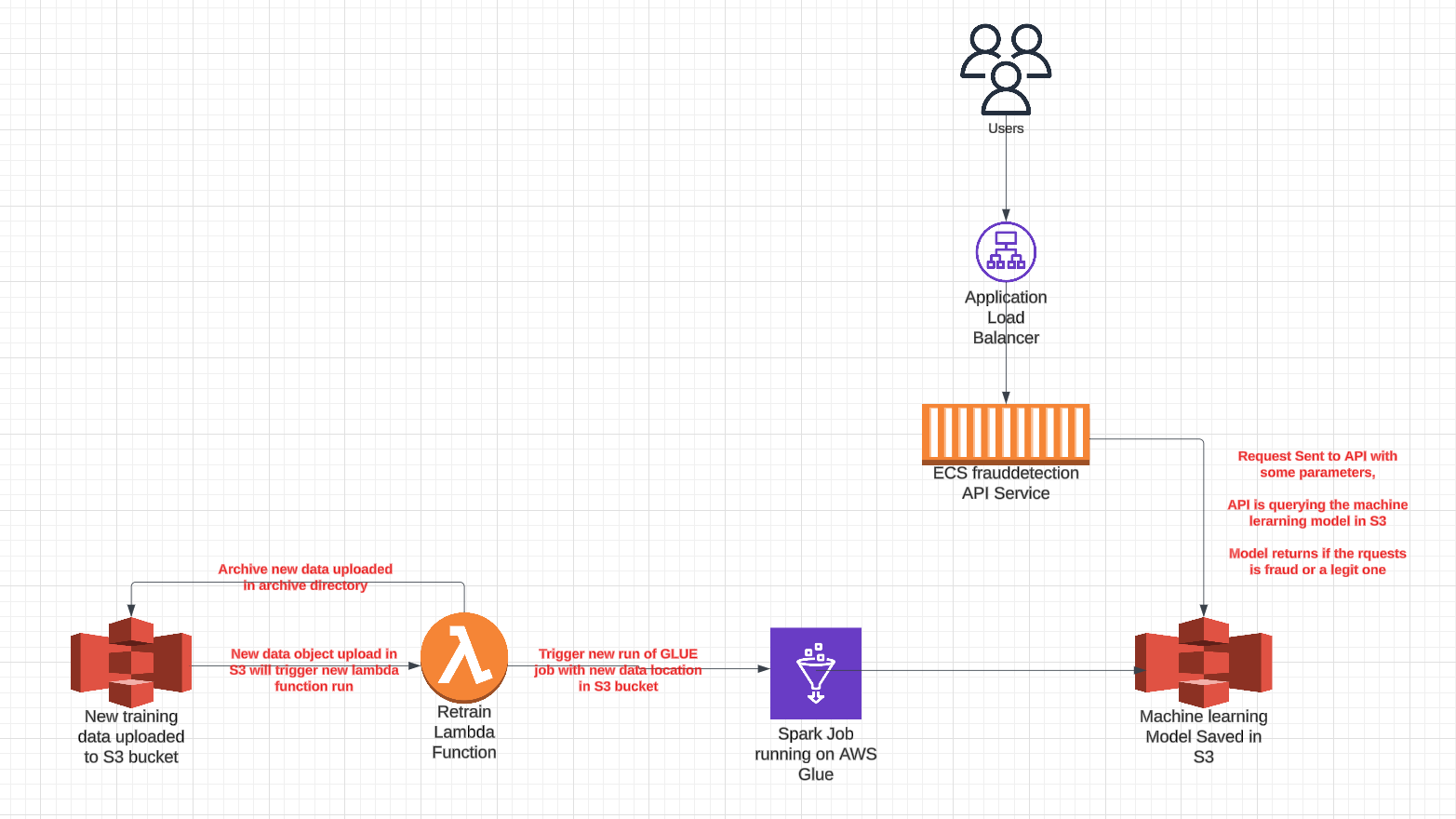
In this case we will use the cloud native solutions in AWS: A lambda function will provide the serverless solution, and using S3 trigger to lambda function to trigger the job whenever a new dataset is uploaded; The lambda function itself will trigger the glue job to train the model and upload it to S3, as explained in previous paragraph.



**Note to be explained here**:

* It was asked to watch a database/directory: Always watching a directory or database means it has to be running all the time, instead of working on demand. However, choosing the current architecture of S3 trigger will allow it to work serverless on demand whenever a new object is uploaded.
* The Machine learning pipeline is not built to learn in progress: It means it takes a dataset, and train the model as new on them. That means, new dataset, if it doesn’t include old dataset in it, machine learning model will built brand new model, without taking in consideration previous runs: Hence, uploading new dataset has to be done in a way to append it to old dataset as well. Otherwise, codebase and architecture has to change to have a step where old already-built pipeline is being loaded and built upon the new analysis.

**General Architecture**



**Process as follows:**

1. Trigger from S3 when new dataset is uploaded
2. Lambda function is triggered to trigger itself a Glue job to train the model with uploaded S3 data.
3. A model trained is saved to S3 bucket.
4. Requests coming from users will be directed to API and query the machine learning model in S3, and return if it is a fraud request or not.

**Important Note**:

* It was required to run module tests for each service; however the tests are not present, hence the only test screenshot provided is the test\_api.py script.