Assignment 3

1. Create a CloudFront distribution for your website and explain each step with great technical detail.

CloudFront is a service provided by aws and is used to speed up a distribution from any static and/or dynamic web page. What it does, is that it connects to a worldwide network of data centers called “edge locations” and through it delivers your content.

To create the CloudFront distribution we use the following command:

```
[CV9FCYQ4XQ:~ mxm0822$ aws cloudfront create-distribution --origin-domain-name mi
chelle.cetystijuana.com.s3-website-us-east-1.amazonaws.com --default-root-object
 index.html
```

First we indicate the service we want to use from aws, in this case we want to create a cloudfront, so we use “cloudfront” to access that service. The second parameter is the functionality required to create it. We must indicate the origin were we want to make the distribution (—origin-domain-name), which in this case is my page bucket (s3 bucket url) and the default object (—default-root-object), which is the index from my page.

When running this command, a .json file is shown with the default values of a CloudFront distribution and must be saved.. In this code we can find the status of our distribution, if it’s InProgress or already Deployed. To be able to see more in detail what our configurations are, with our distribution id, we can run the following command:

```
For more details, please visit https://support.apple.com/kb/HT208050.
[CV9FCYQ4XQ:~ mxm0822$ aws cloudfront get-distribution --id E3UU89DV6WLC6M
{
```

In the code above, with “get-distribution” we are obtaining the information stored in the distribution indicated by the id provided with the parameter “—id”.

With my DomainName generated, we can confirm that this was done successfully: michelle.cetystijuana.com.s3-website-us-east-1.amazonaws.com

2. Update your DNS Record to route to the CloudFront end point and explain each step with great technical detail.

To do this update, in the record.json file previously created, we must make the following changes:

- In the “Action” parameter, we change it to UPSERT since we already crated the DNS record. If this isn’t changed, it will create a new one and can get tricky. UPSERT creates one if it doesn’t exists, but since it does, it just updates the object.

```
{
  "Comment": "CREATE record ",
  "Changes": [{
  "Action": "UPSERT",
                "ResourceRecordSet": {
                            "Name": "michelle.cetystijuana.com",
                            "Type": "CNAME",
                            "TTL": 3,
                          "ResourceRecords": [{ "Value":  "michelle.cetystijuana.com.s3-website-us-east-1.amazonaws.com"}]
}}]
}
```

- In the “ResourceRecords” parameter, we changed the value to route it to the CloudFront end point and not the s3 bucket. From our json generated when creating the distribution, we obtain our DomainName element, in this case my domain name is: d2659ndpawa46v.cloudfront.net

With the following command, we change the settings in the record. This will redirect our page, allowing users to view it from our CloudFront distribution created. Doing this, we can now use our DomainName to access the website.

```
[CV9FCYQ4XQ:~ mxm0822$ aws route53 change-resource-record-sets --hosted-zone-id Z]
03346142C3RKH191036Y --change-batch file:///Users/mxm0822/Documents/School/6tose
mestre/CloudComputing/record.json
{
```

In this command we use the parameter “—hosted-zone-id”, we can obtained it by running *aws route53 list-hosted-zones.* In the parameter “—change-batch” we indicate the path where our .json file, in which the modified DNS record is stored. This gives us the following response:

```
[CV9FCYQ4XQ:~ mxm0822$ aws route53 change-resource-record-sets --hosted-zone-id Z]
03346142C3RKH191036Y --change-batch file:///Users/mxm0822/Documents/School/6tose
mestre/CloudComputing/record.json
{
    "ChangeInfo": {
        "Id": "/change/C01326381M8KHRWJ7LHW7",
        "Status": "PENDING",
        "SubmittedAt": "2023-02-16T03:41:32.681000+00:00",
        "Comment": "CREATE record "
    }
}
```

An important thing to notice here is the Status is “PENDING” again, since it (Route 53) needs to apply all changes and spread them in the DNS servers. It can take some time, when is finishes Status will change to “INSYNC”.

Now we can confirm that my subdomain now points to my CloudFront distribution domain, if this was done successfully we can access: michelle.cetystijuana.com and check it.

If when accessing it we see a 403 error, it’s because the record is set to access only with the url provided by it, to avoid this, we created a certificate. We must change this in our distribution configuration, we can obtain it with the following command:

```
[CV9FCYQ4XQ:~ mxm0822$ aws cloudfront get-distribution --id E3UU89DV6WLC6M
```

Here the id was generated when we created the CloudFront distribution. Here we copied the code that was in the section “DistributionConfig” in a new .json file, where we are going to change our needed values.

First, we must obtain our “CallerReference” by running the following command:

```
CV9FCYQ4XQ:~ mxm0822$ aws cloudfront get-distribution --id E3UU89DV6WLC6M | grep
 Caller
            "CallerReference": "cli-1676081828-77597",
```

In this case, my CallerReference is cli-1676081828-77597 and it must be changed in the new .json file we are working in.

```
"CallerReference": "cli-1676081828-77597",
```

As mentioned before, we want to add our subdomain to the list of domains that can redirect our CloudFront distribution. To do this, we must add it as an “Alias” in the current .json file, as shown in the picture below.

```
"Aliases": {
    "Quantity": 1,
    "Items": [
        "michelle.cetystijuana.com"
    ]
},
```

We also modified our certificate as the picture below, same certificate was used for all students:

```
"ViewerCertificate": {
    "CloudFrontDefaultCertificate": false,
    "ACMCertificateArn": "arn:aws:acm:us-east-1:292274580527:certificate/e961006c-fa16-48ce-aeae-1f093f83db26",
    "Certificate": "arn:aws:acm:us-east-1:292274580527:certificate/e961006c-fa16-48ce-aeae-1f093f83db26",
    "SSLSupportMethod": "vip",
    "MinimumProtocolVersion": "TLSv1",
    "CertificateSource": "acm"
},
```

Now having this changes, we proceed to update them and save it with the following command:

```
CV9FCYQ4XQ:~ mxm0822$ aws cloudfront update-distribution --id E3UU89DV6WLC6M --d
istribution-config file:///Users/mxm0822/Documents/School/6tosemestre/CloudCompu
ting/dist_config_acm_Michelle.json --if-match E99QG987FJDKH
```

First we must indicate the service, here we are making an update of a distribution so we use “update-distribution”, then our distribution id and then the path where our new configuration with the certificate is stored. An important thing to notice here is the “—if-math” parameter, this is used to assure that the version we want to update is the correct one and not a passed one. We do this by indicating our ETag.

To obtain our ETag we run the following command:

```
CV9FCYQ4XQ:~ mxm0822$ aws cloudfront get-distribution --id E3UU89DV6WLC6M | grep ETag
    "ETag": "E99QG987FJDKH",
```

3. Explain what it means to minify website resources (html, js, css) and the advantages and disadvantages of this process.

It refers to the process of minification, which means to reduce all unused code and characters from the original code of a website and should be done without affecting or sacrificing any functionality. Used to reduce load times and bandwidth usage of websites.

Unnecessary code could be line breaks, white space characters, comments, etc. One way to reduce it, is concatenating those unused files to save memory. One important thing to have in consideration, is that this process is carried out before the final deployment, meaning once coding is completed.

Advantages

- It improves site speed and accessibility, which makes it better for the users experience.
- Beneficial for users who have a limited plan and would like to save on their bandwidth usage while navigating through a page.
- Minified files provide the same functionality while also reducing bandwidth of the networks requests.
- One mayor advantage is that after removing unnecessary code, which allows to minimize the time by not downloading unwanted data, that is just going to use memory without a purpose.

Disadvantages

- If it’s not done right, code can get hard to comprehend for other people who didn’t write it and adjustments in the future can be very complicated.
    - Important to keep the original full-sized sources and have the fundamental
- Minifying resources such as html, css and js files, has less impact when comparing to other optimizations, such as optimizing images.
- Since all commented code is removed, doing this practice can lead to many errors that become complicated bugs to fix.

4. Write a python script (and explain each step with great technical detail) to:

First, we must have previously installed boto3. In our script, we now need to import it as a library to be able to use aws services, in this case we will use DynamoDB.

a.Create or update a record in the Students DynamoDB table based on the id.

According to documentation, first we initialize the DynamoDB client.

```
import boto3

dynamodb =  boto3.resource("dynamodb")
```

First, to create a record in the “Students” table, we use the *.put_item* function in which we indicate the values of each argument. We must indicate two parameters, the name of the table where we want to create the record and the information of the item we want to create. Item only accepts the type of data dictionary, in which we must indicate and match with the DynamoDb column names. We indicate the type of data and the value itself. Here strings are being used, which is why we use ‘S’.

```
record = dynamodb.put_item(
    TableName='Students',
    Item={
    'id':{'S':'28241'},
    'full_name':{'S':'Michelle Andrea Muñiz Lopez'},
    'personal_website':{'S':'michelle.cetystijuana.com'}
}
)
```

To do the update, we can use the *.update_item* function. Since is an update, we have different parameters that we must indicate. As before, we start by indicating the table were we want to make the changes, in the Key parameter we have UpdateExpression, which is the attribute we want to update. In this case we are changing the id and the action that we want to do it to set it. ExpressionAtributeNames is where we define the reference of the change we want to make, we are updating “#id” for the column “id”. In ExpressionAttributeValues, we indicate the new value of the attribute and we make the new reference as well with ‘:idnum’, which is the name of the new entry.

```
record1  = dynamodb.update_item(
    TableName='Students',
    Key={
    'id':{'S':'28241'},
    },
    UpdateExpression='set #id= :idnum',
    ExpressionAttributeNames={
    '#id':'id',
    },
    ExpressionAttributeValues={
    ':idnum':{'S':'33580'}
    }
)
```

b.Delete a record in the Students DynamoDB table based on the id

To delete a record already created, we use the function *delete_item.* As well as when we created the record, we must indicate the TableName were is the record we want to delete and the Key. For Key we just need to indicate the primary key of the table, in this cases it is the id of the student, so we just pass the id of the student we want to erase.

```
dynamodb.delete_item(
    TableName = 'Students',
    Key={
    'id':{'S':'28241'}
    }
)
```

c.Find a record in the Students DynamoDB table by id.

To find a record by the primary key, in this case id, we use the function *.get_item.* As before, we indicate the name of the table were we want to search for the record. In Key, we pass the id of the student for whom we want to find the record. In response you have a dictionary with all the information regarding that record.

```
record=dynamodb.get_item(
    TableName='Students',
    Key={
    'id':{'S':'28241'}
    }
)
```

5. Explain the difference between Growth mindset and Fixed mindset according to Carol Dweck.

First of all, a mindset helps an individual shape their personality and is what will drive you to reach your highest potencial. Mindset is something that you learn depending on your personal context and what your daily life looks like, it can be learned from direct family such as parents, or from outsiders like friends, teachers and even the media. With it you form and define what success, failure and effort is.

The fixed mindset is where intelligence, personality and abilities are unchangeable and that you can't improve them. The belief that the set of skill you are born with are innate. There is always the fear that you are never enough to achieve your goals, which holds you back from reaching you full potential. Challenges are avoided at all costs for the same reason of fear and not believing in their abilities. Every time there is an obstacle they have to face, they don't put the effort in to overcome it, instead they go the easy route.

On the contrary, the growth mindset is where obstacles are a motivation, here you believe in your abilities and seek to overcome every challenge. There is no fear of making mistakes or failing, instead feedback is used to improve and develop your talents. You believe that with much effort and perseverance, natural qualities you were born with can be improved, while also enjoying the process of growing as a person and becoming more productive.

The primary difference, as mentioned before is in the beliefs of growth and improvement, where in the fixed mindset their is no room for that, what you were born

with is what you are stuck with for the rest of your life; while in the growth mindset their is always something to get better at and keep learning to achieve your goals and reach success.