WEBAPP PROMOTING CLEAN ENERGY DEPLOYED ON AWS Elastic Kubernetes Service (EKS)

1. Contents

Contents

2. Abstract

The main project aim was to build webapp, deploy in cloud and build continuous Integration/continuous deployment around this system. The app provides some information's about clean energy possibilities for homeowners in Ireland, links to companies which can help with it and links for grants from government if technology apply for such grant. Webapp also has solar panels calculator which provide average energy generation according to the size of panels in Ireland, all entries from calculator are stored in database (AWS RDS SQL) for future use and are visible for admins on this site. Once the application was ready, docker container image was built and stored in AWS ECR (Elastic Container Service), another step was to deploy in AWS EKS (Elastic Kubernetes Service) cluster. Last step was to build continuous Integration/continuous deployment system.

3. Acknowledgments

I would like to thank my teachers who gave me opportunity to work on this project. During the project I have learned a lot about all framework needed to finish project like ASP.NET core and AWS Elastic Kubernetes Service, SQL, and other useful info about clean energy technologies. Big thanks to my supervisor Mehran Rafiee who give me golden points which help me a lot in my project.

For ASP.NET core I have used these videos from YouTube channel, and tutorial site:

https://www.youtube.com/watch?v=iKmUMgZj-cE (Industrial and IT Automation, n.d.)

https://www.youtube.com/watch?v=YUPg41kG_kw (BoostMyTool, n.d.)

https://www.w3schools.com/asp/default.asp (W3Schools, n.d.)

For AWS EKS I have used these videos from YouTube channel, and tutorial sites:

https://www.youtube.com/watch?v=LlisKI-gN5w (Sandip Das, n.d.)

https://www.youtube.com/watch?v=nEK7e0QUVio (Sandip Das, n.d.)

https://docs.aws.amazon.com/eks/latest/userguide/getting-started.html (AWS, n.d.)

https://docs.aws.amazon.com/eks/latest/userguide/add-user-role.html (AWS, n.d.)

https://docs.aws.amazon.com/eks/latest/userguide/metrics-server.html (AWS, n.d.)

For Calculator formula used these websites:

https://www.saurenergy.com/solar-energy-blog/here-is-how-you-can-calculate-the-annual-solar-energy-output-of-a-photovoltaic-system (Saur Energy, n.d.)

https://solargis.com/maps-and-gis-data/download/ireland (Solargis, n.d.)

For information about AWS tools used these websites:

https://docs.aws.amazon.com/whitepapers/latest/overview-deployment-options/amazon-elastic-kubernetes-service.html (AWS, n.d.)

https://aws.amazon.com/codecommit/ (AWS, n.d.)

https://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-parameter-store.html (AWS, n.d.)

https://aws.amazon.com/codepipeline/ (AWS, n.d.)

https://aws.amazon.com/codebuild/ (AWS, n.d.)

https://aws.amazon.com/ecr/ (AWS, n.d.)

https://aws.amazon.com/ecr/ (AWS, n.d.)

https://aws.amazon.com/cloudwatch/ (AWS, n.d.)

https://aws.amazon.com/iam/ (AWS, n.d.)

4. Introduction

My Project is online Web Portal wrote in ASP.NET core framework, which contains information about clean energy technologies that can be used by homeowners, links for fitting companies in Ireland and links for grants if they apply for that technology also in Ireland. Is deployed on AWS cloud EKS (AWS Kubernetes) and contain CI/CD (Continuous Integration/Continuous Deployment) system which use AWS CodeCommit, CodePipeline, ParamaterStore, CloudWatch, Elastic Container Repository and CodeBuild. It is Highly Available, Secure, Redundant and a Scalable system, deployed on two availability zones on North Virginia region us-east-1a and us-east-1b.

a. Aims of the project

1. Implement what I have already learned during the Higher Diploma
2. Learn new tools and frameworks needed by current DevOps engineers which I am planning to be in future
3. Be helpful for people looking for clean energy technologies
4. Be my portfolio for future employer

Ad. 1 During my Higher Diploma studies I have learned a lot about the AWS Cloud tools and wasn't difficult to design and implement VPC (virtual private network) for the project as we had modules which covered this technology, also creating networking, subnetting for the system was simple. ASP.NET core was also in one of modules and material help me a lot in this project. On one of assignment, I did AWS ECS (Elastic Container Service) deployment and had some knowledge about Docker and containerization on AWS.

Ad.2. One of main tools I had to learn was ASP.NET layout of the page and how CSS is implemented in such framework, how you use Razor pages and the model cooperating with view of the page. Also seeing how to save, retrieve data from SQL database.

AWS EKS how to deploy and maintain cluster where most of work is done through AWS CLI with kubectl plugin commands.

Ad.3. Hope this web application will be useful and helpful for users, having information of clean energy technologies explained, have links to fitting companies around Ireland and links for possible grants from Government institutions like Sustainable Energy Authority of Ireland.

b. Scope

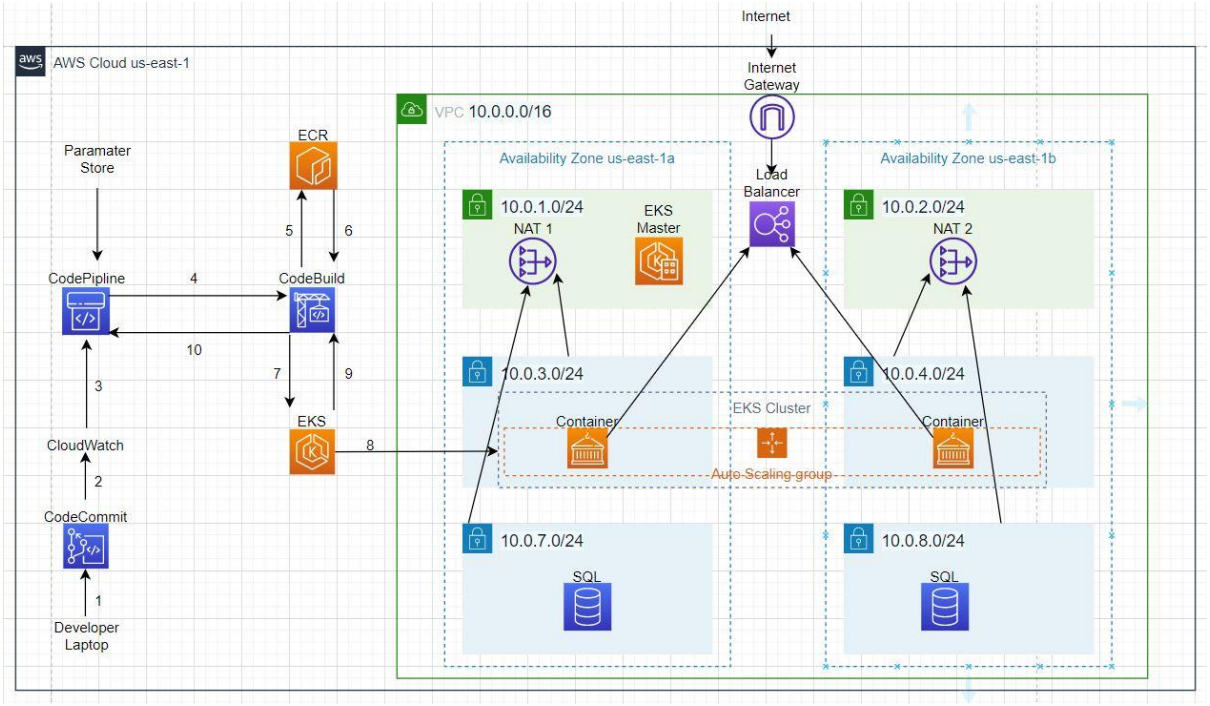
Scope of this project follow:

- Webapp in ASP.NET core
  a. Website form and calculations
  b. Database connection for storing results
  c. Display all records after logging in
- Continuous Integration and Continuous deployment system on AWS by using CodePipeline, CodeBuild, CodeCommit, Parameter store from System Manager, CloudWatch events
- AWS CodeCommit setup, where we store our webapp code

- AWS ECR setup, where we store our webapp docker container
- AWS EKS service setup, orchestrator tool that our containers run on
- Acquiring all necessary clean energy details and formula to make calculations for users
- AWS VPC setup for our infrastructure
- Setup AWS Security groups and IAM roles to achieve highly secure system
- AWS VPC Subnetting to separate frontend and backend services
- AWS Networking to make sure packet traffic flow desired way
- AWS Autoscaling group setup so our webapp scale when needed

5. Architecture

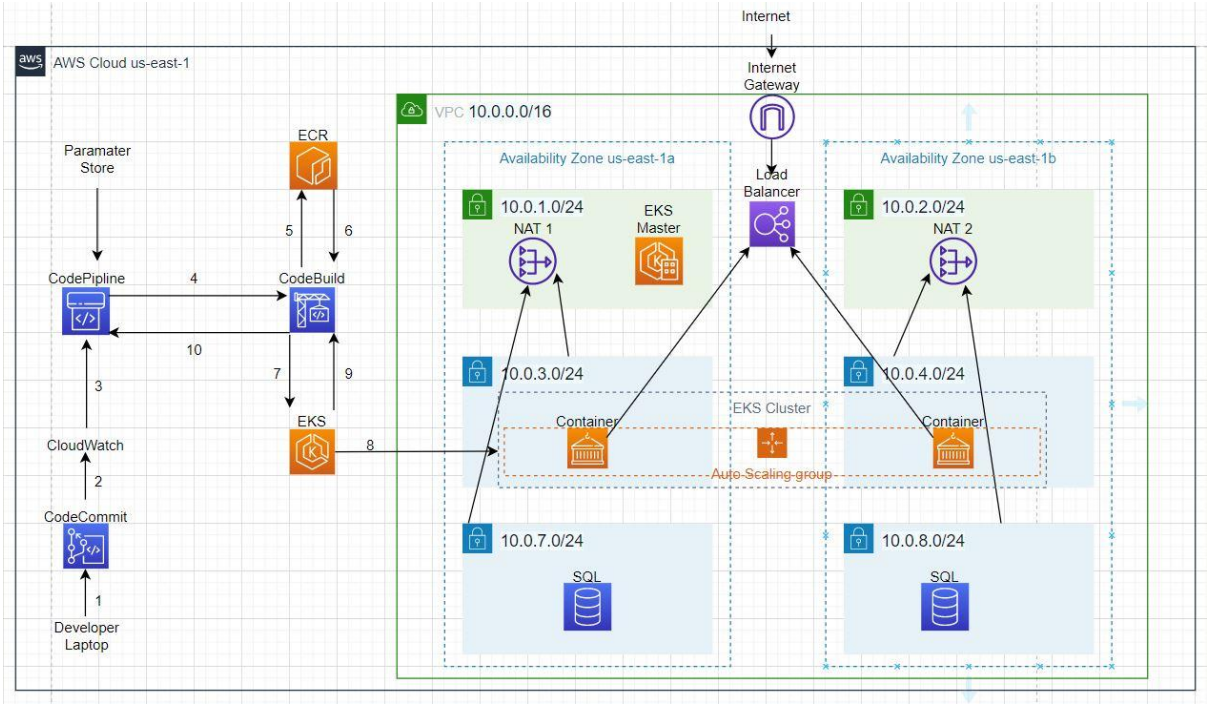
a. Infrastructure Design Specification



EKS cluster is deployed on Private Subnet 1 and 2, backend database in Private Subnet 3 and 4 on EC2 nodes and EKS master is deployed across all subnets. There are 2 NAT gateways one on each availability zone. Containers accept traffic just from the Database security group on port 1433, load balancer security group all traffic and EKS API master security group all traffic. All outbound traffic from containers, nodes and database are going through NAT gateways. There is 2 EC2 nodes deployed across the 2 availability zones were containers run. Autoscaling group is set to 1 minimum node, desired 2 nodes and if spot needed up to 3 nodes can be deployed. EKS cluster pods autoscaling group is called Horizontal Pod Autoscaler and is set to minimum 2 pods and maximum 3 pods, is deployed by watching metrics of CPU if CPU usage is more than 70% than another pod is deployed. For this to work metrics-server must be deployed on EKS cluster which is not by default. SQL Server database on AWS RDS is use with multizone deployment. System is design with 6 pillars of Well-Architected framework for operational excellence, security, reliability, performance efficiency, cost optimization, and sustainability also system is Highly Available by spanning through 2 availability zones. It scales on demand.

b. Continuous Integration and Continuous Deployment

Continuous Integration (CI) is DevOps practice were developers regularly merge their code changes into central repository after which automated builds and test are run. The key goal of continuous integration is to reduce time of software updates. And Continuous deployment is automating deployment, so production happens automatically. For this project to achieve this I have used tool from AWS CodeCommit, CodePipeline, CodeBuild, Elastic Container Repository(ECR), Parameter store from System Manger and CloudWatch events.



When a developer updates the code to central repository(CodeCommit), CloudWatch event is triggered and send information to CodePipeline which starts a CI/CD pipeline, it is verifying changes first on CodeCommit, getting variables from CodePipline project and from parameter store if that pass instruction are send to CodeBuild to start test and build docker container which is send to ECR(Elastic Container Repository), once this stage is done, CodeBuild sends deployment instruction to EKS(Elastic Kubernetes Service), and if all stages pass on CodeBuild, than CodeBuild sends information of finished process to CodePipeline and process is done. Changes done by developer are applied to production in 2-3 min.

CodeBuild require 3 files to run buildspec.yml, deployment.yaml that run with kubectl(Kubernetes CLI) and dockerfile for building docker container:

- Buildspec.yml

“

```
version: 0.2

phases:
  install:
    commands:
      - echo Installing app dependencies...
      - curl -o kubectl https://s3.us-west-2.amazonaws.com/amazon-
eks/1.22.6/2022-03-09/bin/linux/amd64/kubectl
      - chmod +x ./kubectl
      - mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export
PATH=$PATH:$HOME/bin
      - echo 'export PATH=$PATH:$HOME/bin' >> ~/.bashrc
```

```
      - source ~/.bashrc
      - mkdir -p $HOME/.kube
      - aws eks --region us-east-1 update-kubeconfig --name projectEKS
      - echo 'Check kubectl version'
      - kubectl version --short --client
      - echo check connection to claster
      - kubectl get svc
  pre_build:
    commands:
      - echo Logging in to docker...
      - docker login -u $docker_username -p $docker_password
  build:
    commands:
      - echo Build started on `date`
      - echo Building the Docker image...
      - docker build -t $IMAGE_REPO_NAME .
      - docker tag $IMAGE_REPO_NAME:$IMAGE_TAG
$AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com/$IMAGE_REPO_NAME:$IM
AGE_TAG
  post_build:
    commands:
      - echo Build completed on `date`
      - echo Pushing the Docker image to ECR...
      - aws ecr get-login-password --region $AWS_DEFAULT_REGION | docker login
--username AWS --password-stdin
$AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com
      - docker push
$AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com/$IMAGE_REPO_NAME:$IM
AGE_TAG
      - kubectl apply -f eks/deployment.yaml
      - kubectl rollout restart -f eks/deployment.yaml
```

“

- Deployment.yaml

“

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app.kubernetes.io/name: cleanenergy
    app.kubernetes.io/instance: cleanenergy-instance
    app.kubernetes.io/version: "1.0.0"
    app.kubernetes.io/managen-by: kubectl
  name: cleanenergy
spec:
  replicas: 2
  selector:
    matchLabels:
```

```
      app: cleanenergy
  template:
    metadata:
      labels:
        app: cleanenergy
    spec:
      containers:
        - image: 953941695125.dkr.ecr.us-east-
1.amazonaws.com/cleanenergycontainer:latest
          imagePullPolicy: Always
          name: cleanenergy
          resources:
            requests:
              cpu: "250m"
            limits:
              cpu: "500"

          ports:
            - containerPort: 80
```

“

- Dockerfile

“

```
FROM mcr.microsoft.com/dotnet/sdk:6.0 AS build-env
WORKDIR /app

# Copy everything
COPY . ./
# Restore as distinct layers
RUN dotnet restore
# Build and publish a release
RUN dotnet publish -c Release -o out

EXPOSE 80


# Build runtime image
FROM mcr.microsoft.com/dotnet/aspnet:6.0
WORKDIR /app
COPY --from=build-env /app/out .
ENTRYPOINT ["dotnet", "CleanEnergy.dll"]
```

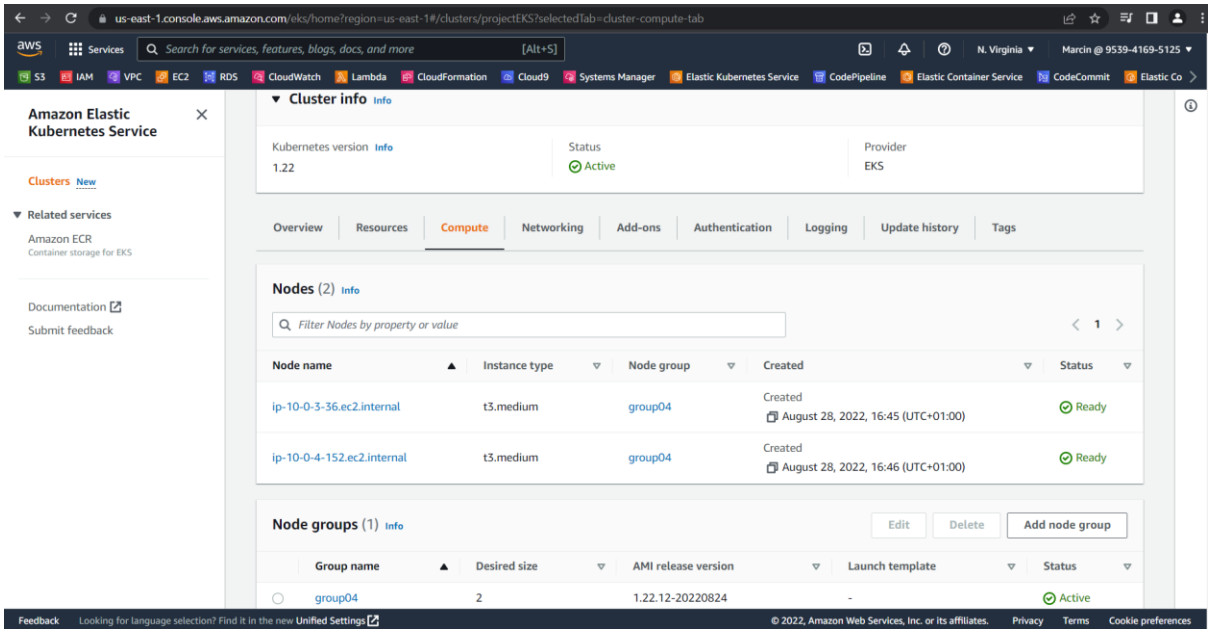
“

c. Environment information’s

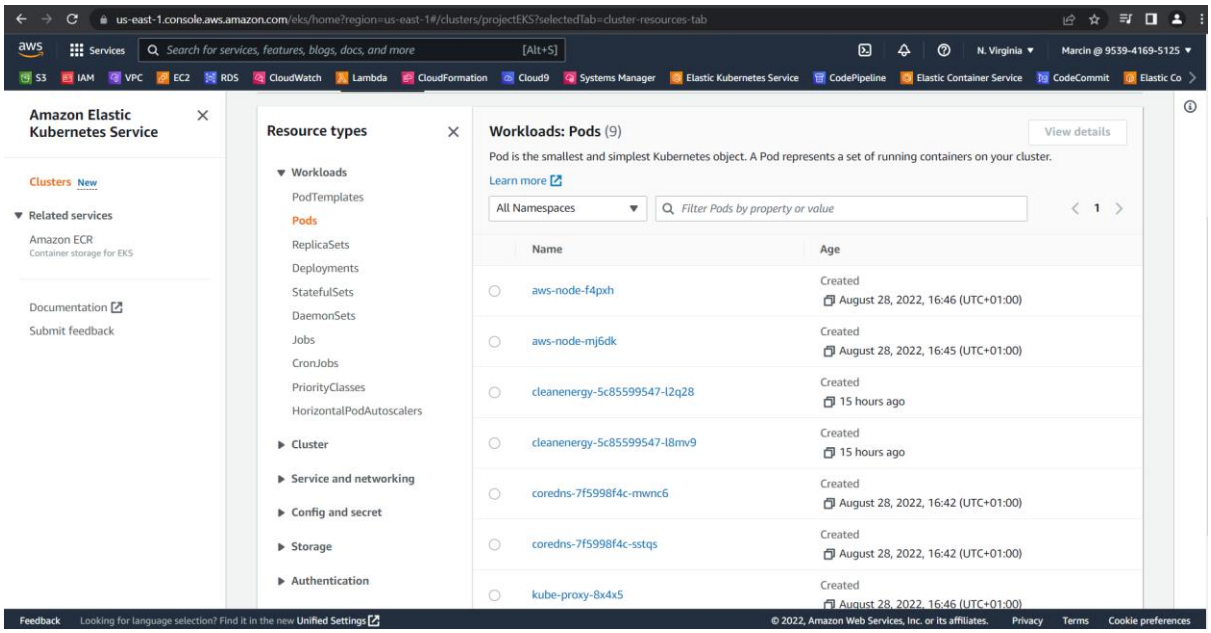
- AWS Elastic Kubernetes Service (EKS)

Amazon Elastic Kubernetes Service is fully managed service where you can run Kubernetes on AWS without needing to install, operate and maintain Kubernetes control plane or nodes. It is integrated with the core AWS services like Auto Scaling Groups, VPC (Virtual Private Network), ECR (Elastic Container Repository), CloudWatch, IAM, load balancer for your containerized applications. As the orchestrator tool is very popular in modern businesses that can automate configuration, management, and coordinate of computer systems, applications, and services. When running applications on EKS you can choose underlying computer power for containers from EC2 or Fargate.

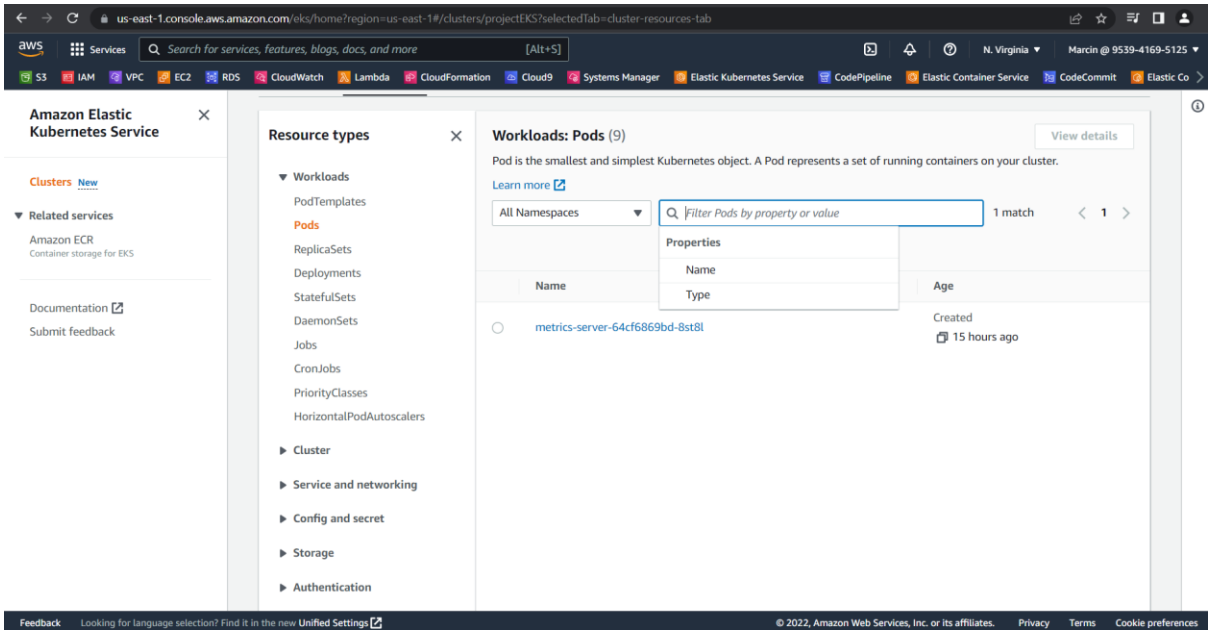
For this project EC2 was chosen as underling computer power for containers:



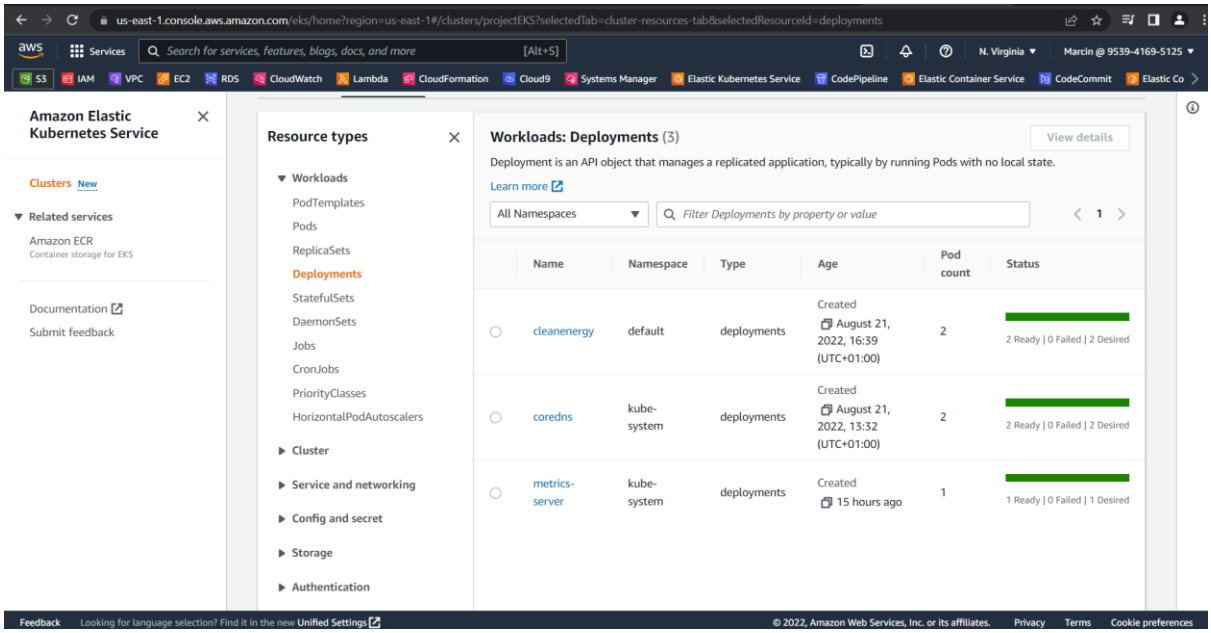
Have 2 pods deployed for clean energy application:



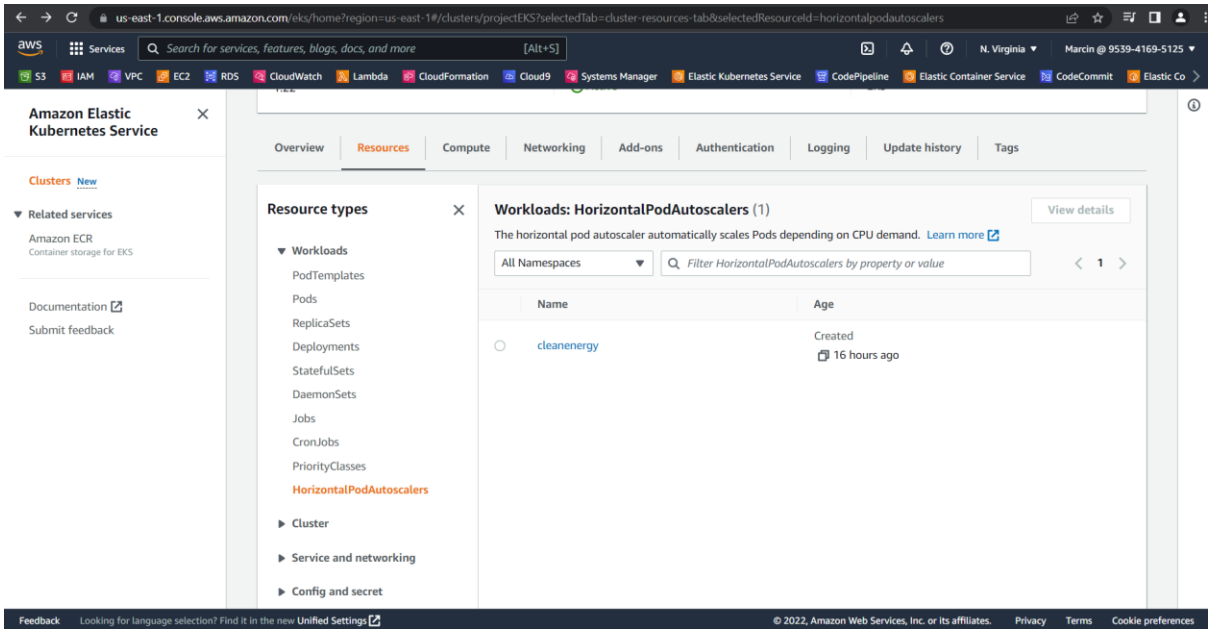
Have 1 pod deployed for metric server that helps manage Horizontal Auto Scaling:



Have 3 deployments running clean energy application, metric-server and dns:



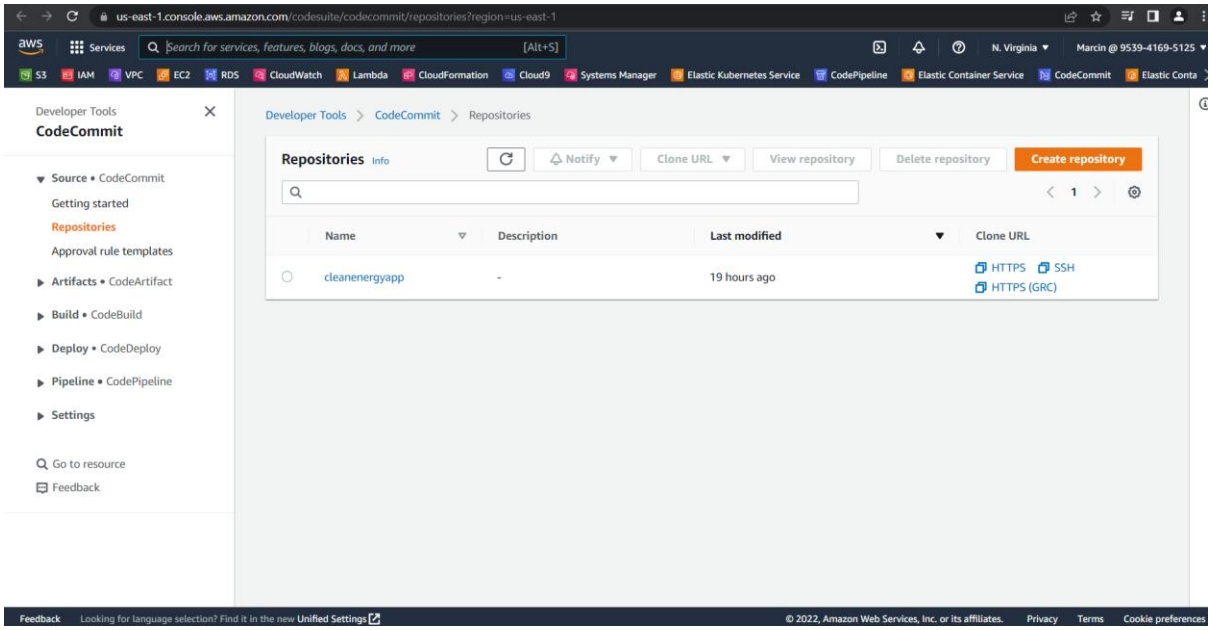
Have 1 HorizontalPodAutoscaler:



- AWS CodeCommit

AWS CodeCommit is managed source control service that hosts private Git repositories. It is secure, highly scalable and is easy for team to securely collaborate on code with contribution encrypted in transit. CodeCommit support standard functionality of Git and you can store anything form code to binaries.

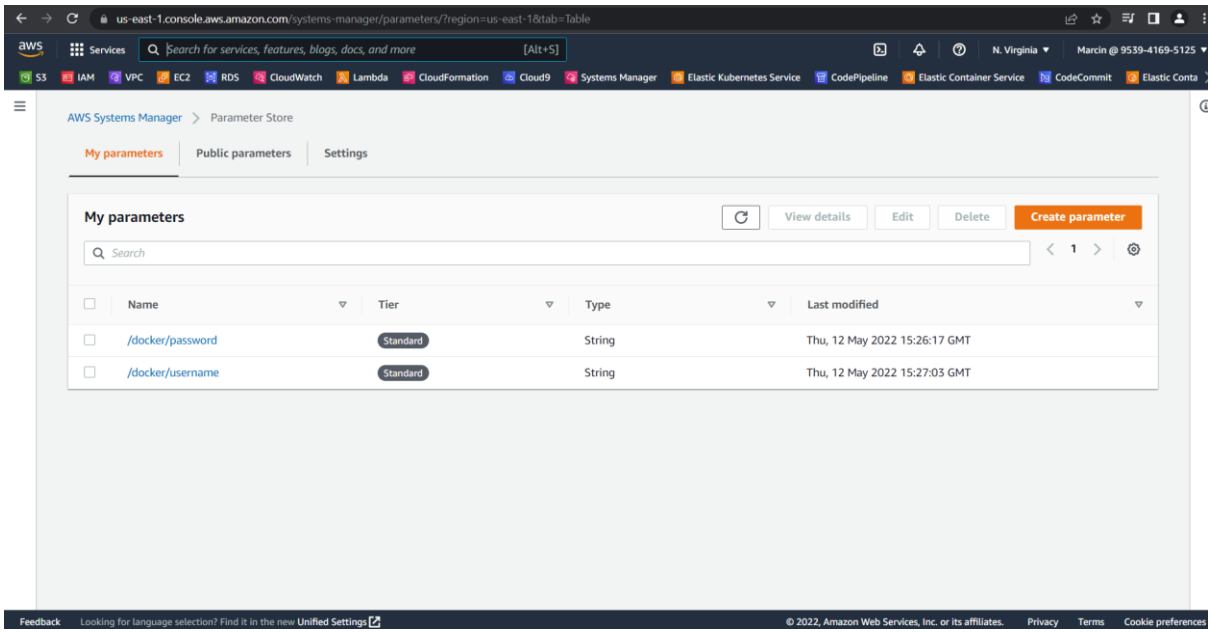
For this project 1 repository was created:



- AWS Parameter Store

AWS Parameter Store is part of the AWS System Manager for holding secrets, providing secure, hierarchical storage. You can store passwords, connection strings and other secrets as parameter values. All this information stored as plain text or encrypted, later can be used in your scripts, commands, documents, and configuration and automation workflows. Parameter store is integrated with other AWS services, and you can reference to Parameter store when needed.

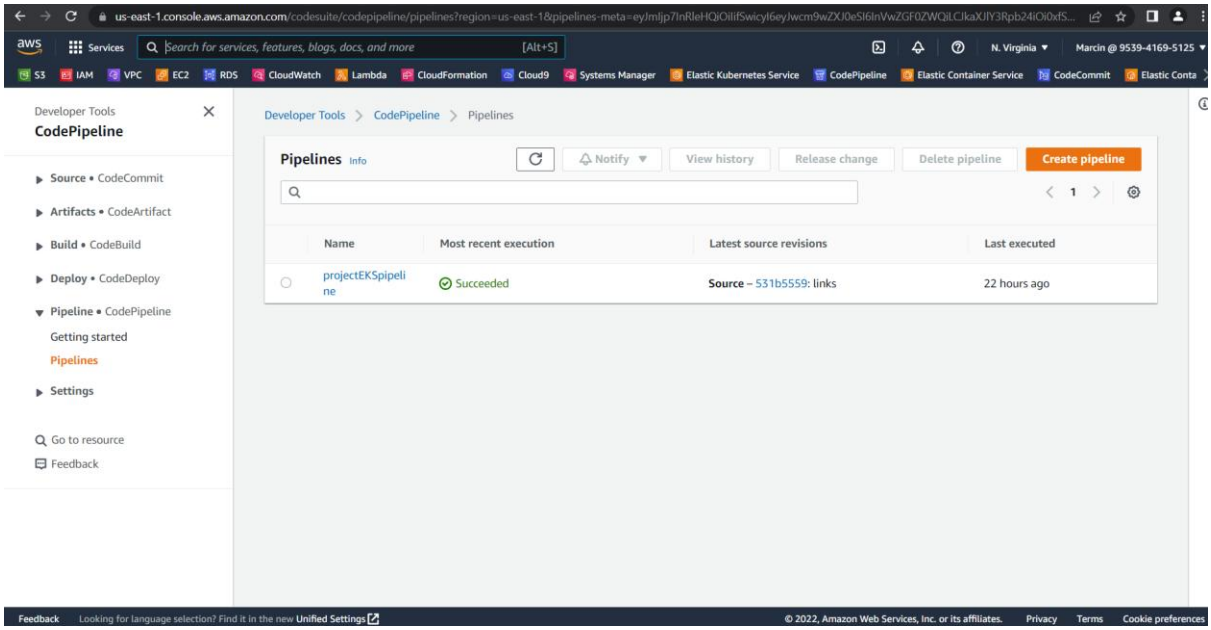
For this project there was 2 values stored in Parameter store:



- AWS CodePipeline

AWS CodePipeline is a fully managed continuous delivery service that helps to automate new applications or infrastructure updates. CodePipeline automate builds, test, and deploy phases once code is change. You can easily integrate third party services like Git, Jenkins, or other plugins. It is enabling features and updates for your application.

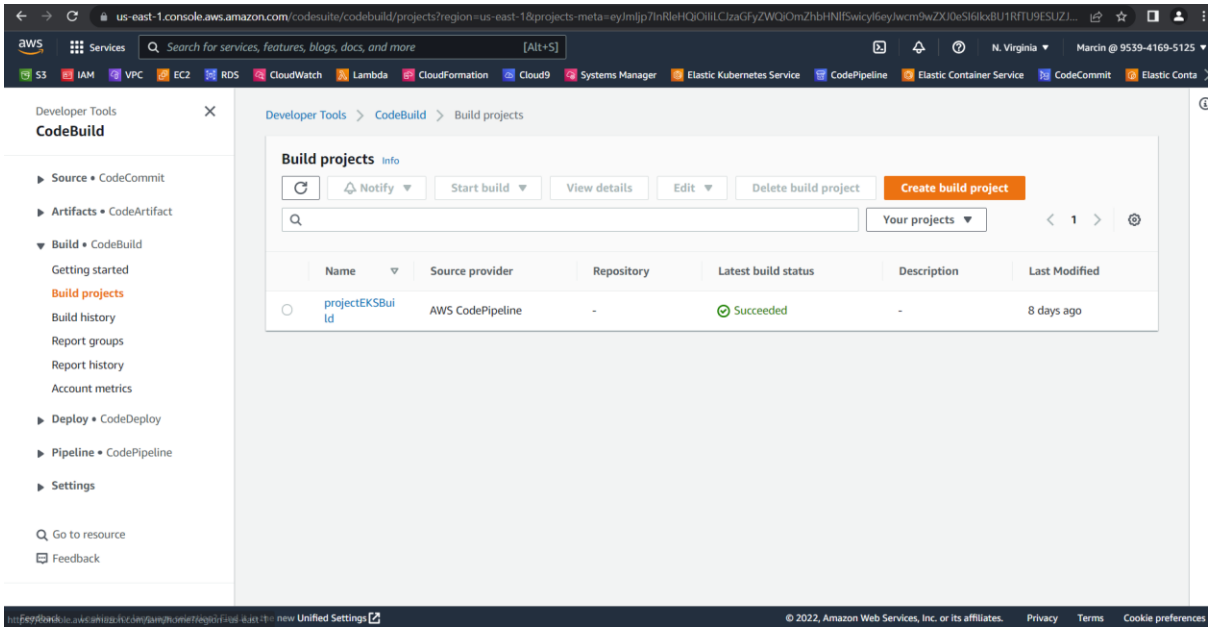
For this project was setup 1 CodePipeline project:



- AWS CodeBuild

AWS CodeBuild is fully managed continuous integration service that allow to compile source code, run tests, and produce software packages that are ready to deploy. CodeBuild automatically patch and build servers, can build docker images and other software builds. It supports few programming languages and frameworks like Java, Bash, Go, Ruby, Python. It can concurrently run multiple builds; you pay for time spent to process tasks.

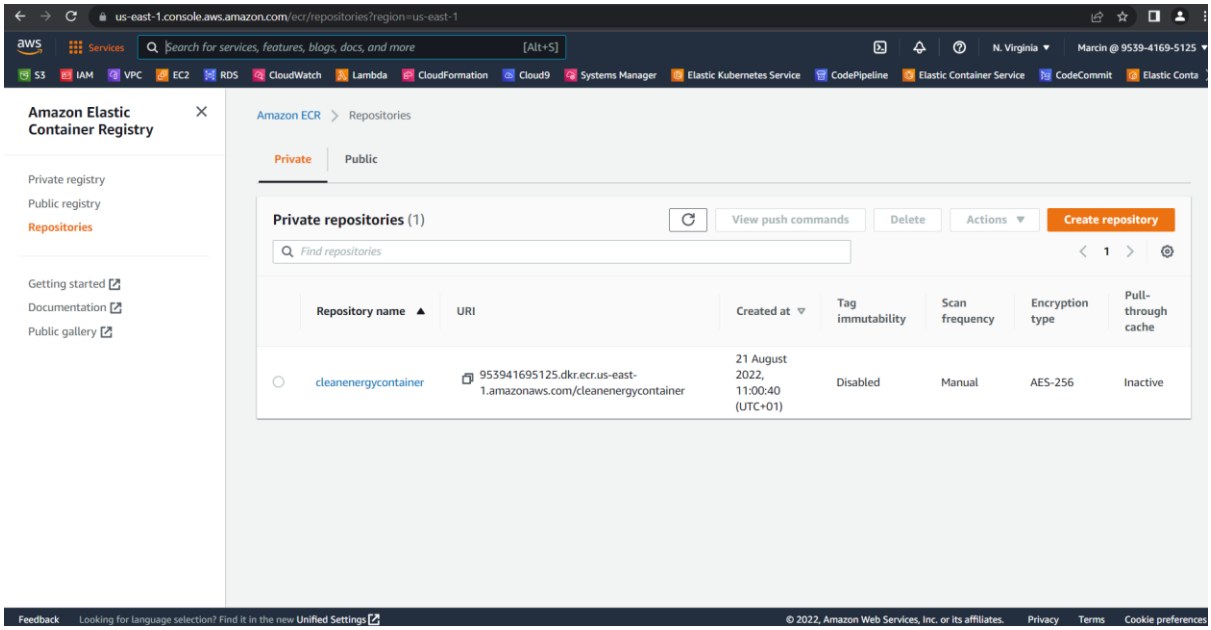
For this project 1 CodeBuild project was setup:



- AWS Elastic Container Repository (ECR)

Amazon Elastic Container Repository is managed container image registry service that is secure, scalable, and reliable. ECR is integrated with AWS Identity Access Management (IAM) for accessing images.

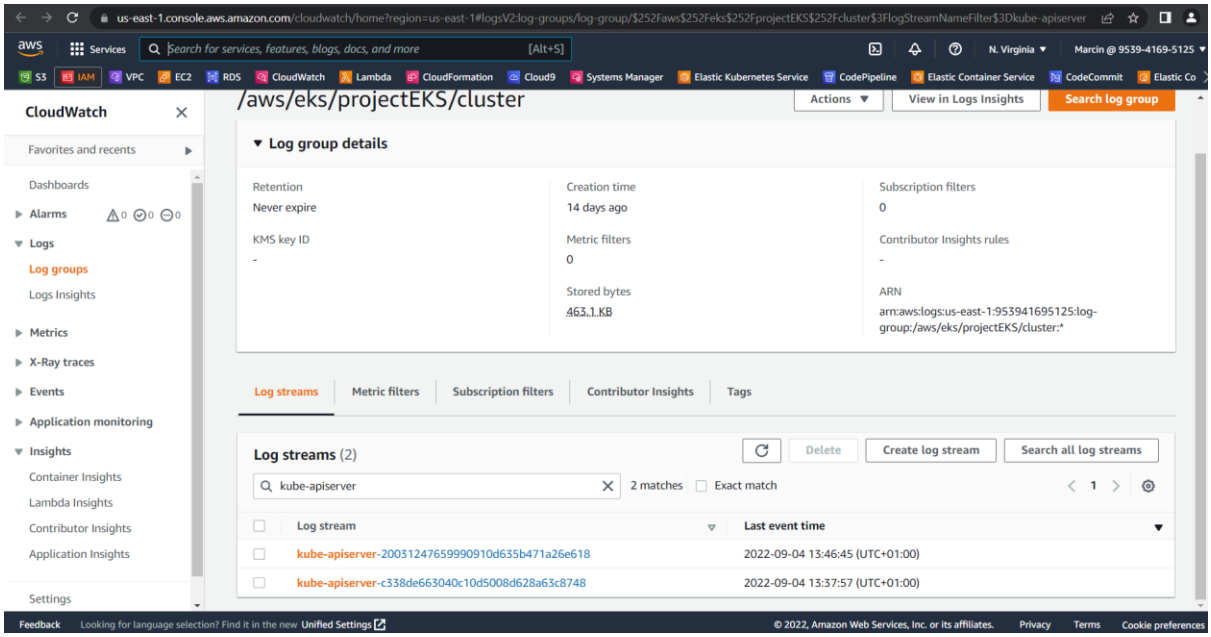
For this project 1 repository was created and 24 hours retention policy was added to reduce cost:



- AWS CloudWatch

AWS CloudWatch is a monitoring and observability service build for DevOps, developers, site reliability engineers and other who use AWS services. CloudWatch collects data in form of logs, events, and metrics to provide complete visibility of AWS resources, applications and services running on AWS. You can use CloudWatch set alarms, visualize metrics and logs, take automated actions, or troubleshoot issues.

For this project, CloudWatch was used for EKS cluster, CodeBuild and for CodePipline to trigger event when code is change on CodeCommit:

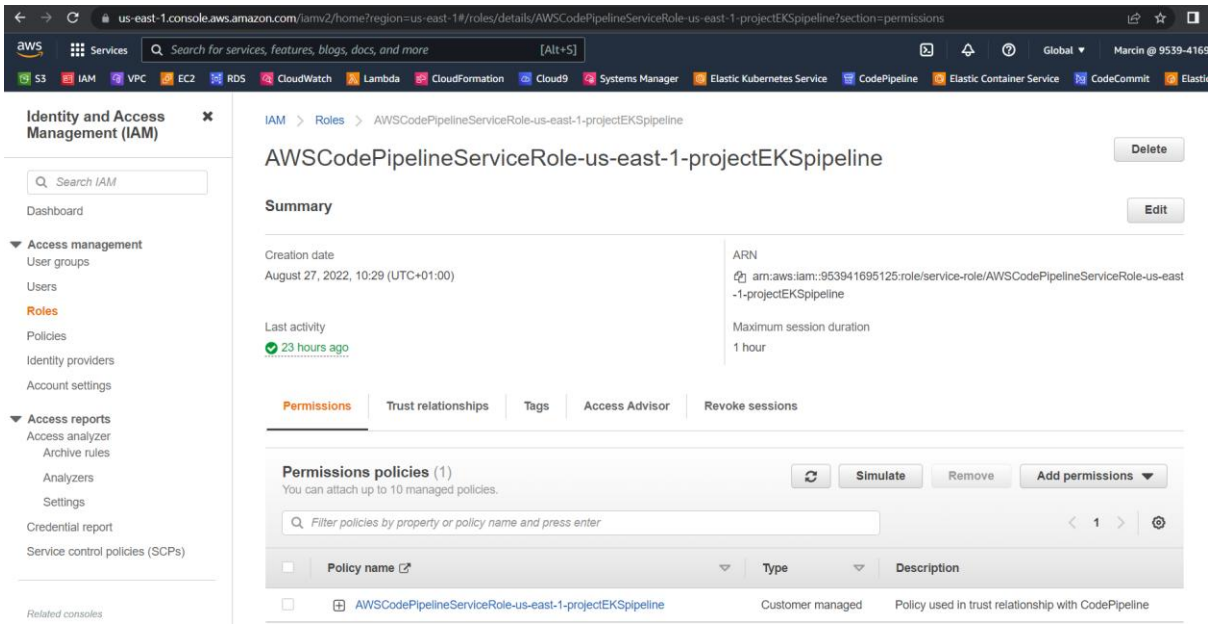


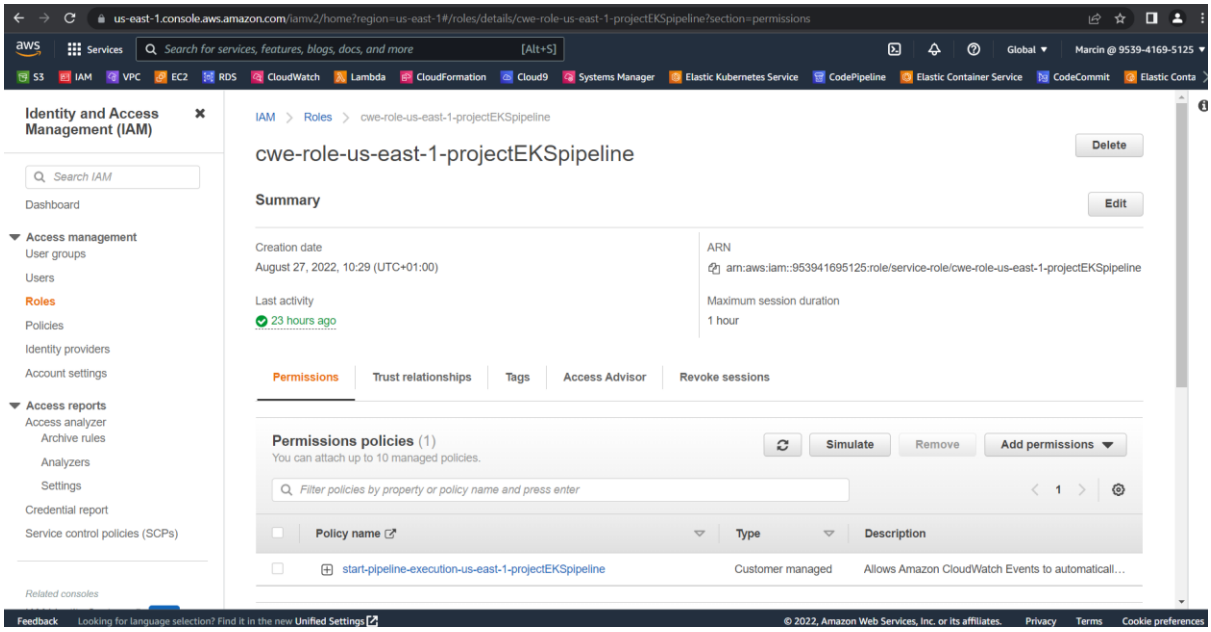
- AWS Identity Access Management (IAM)

With AWS Identity Access Management, you can specify who and what have access to services and AWS resources.

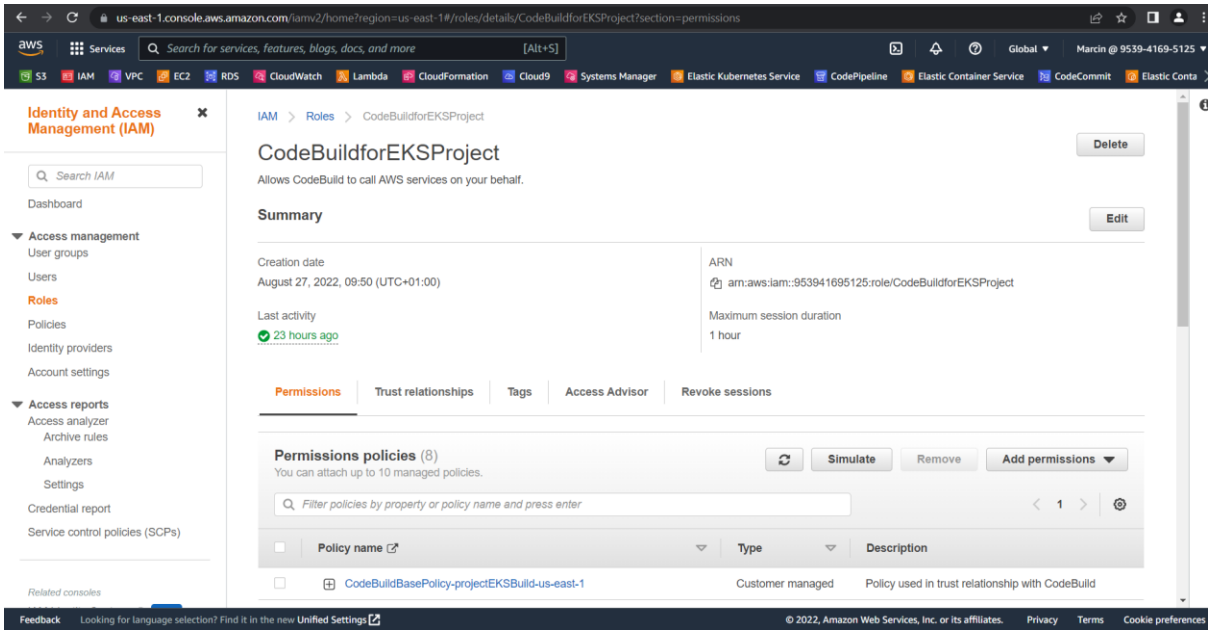
For this project a few IAM roles were created:

CodePipeline

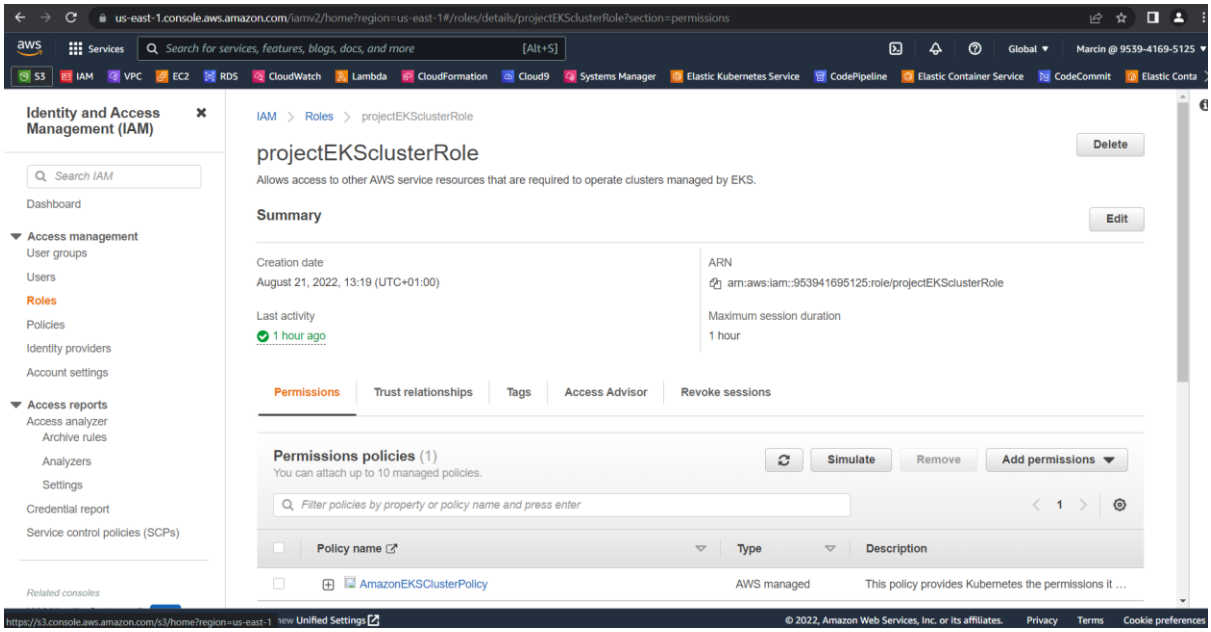




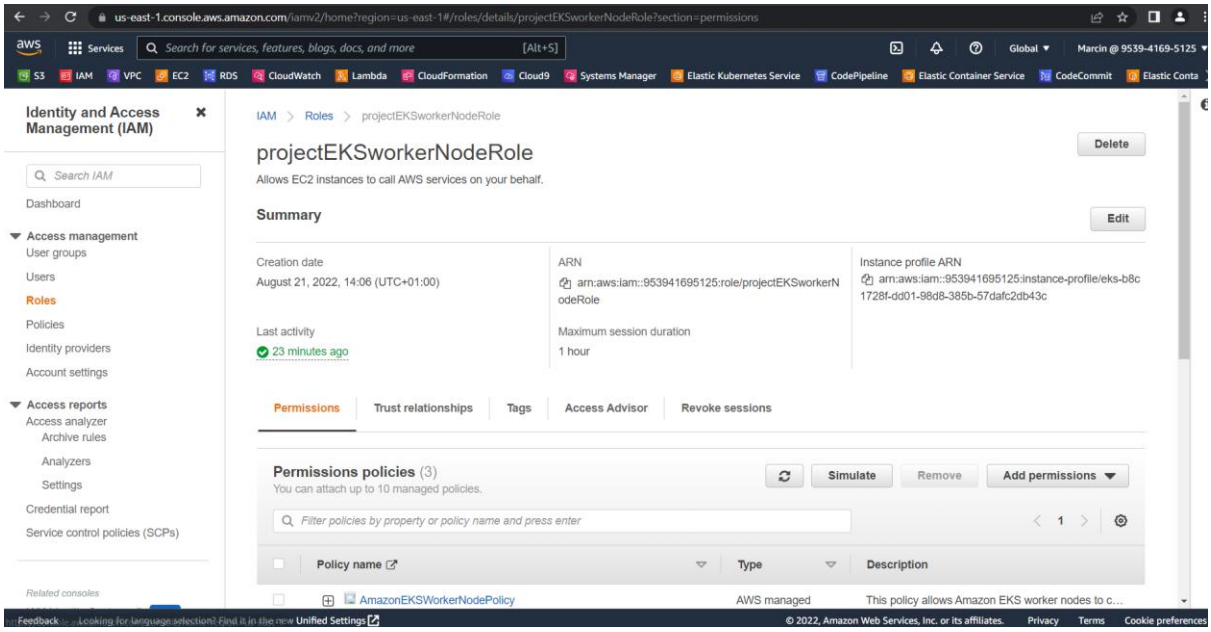
CodeBuild



Elastic Kubernetes Service



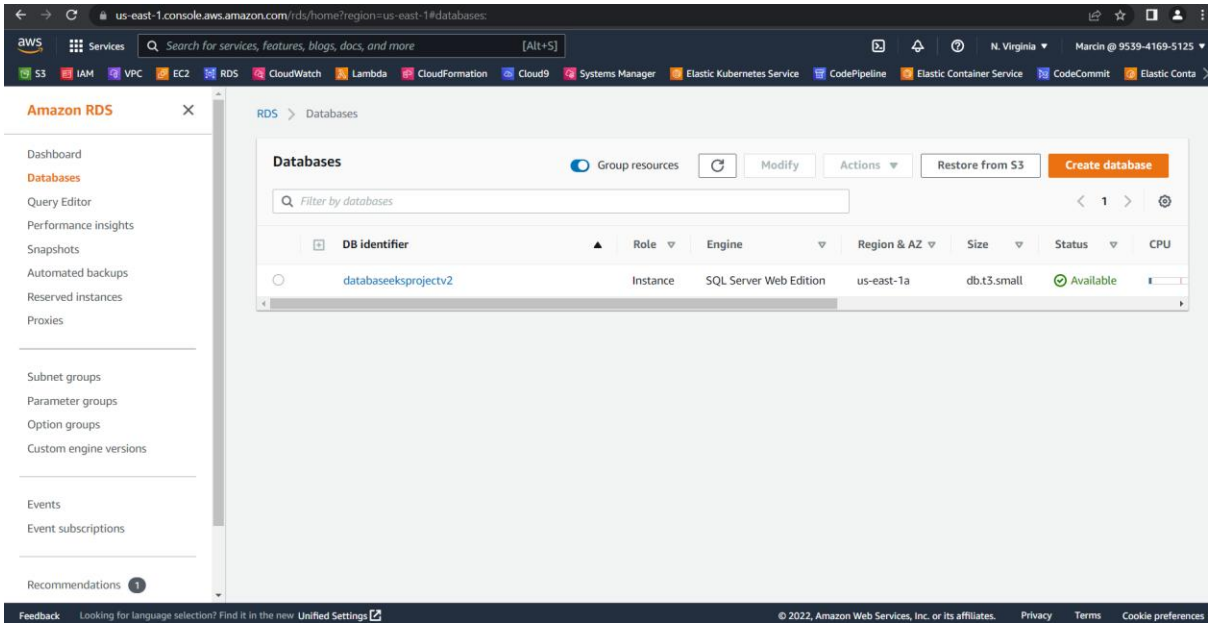
Elastic Kubernetes Service Nodes



- AWS RDS Database

AWS RDS service provides selection of purpose-build databases.

For this project SQL server database was setup:

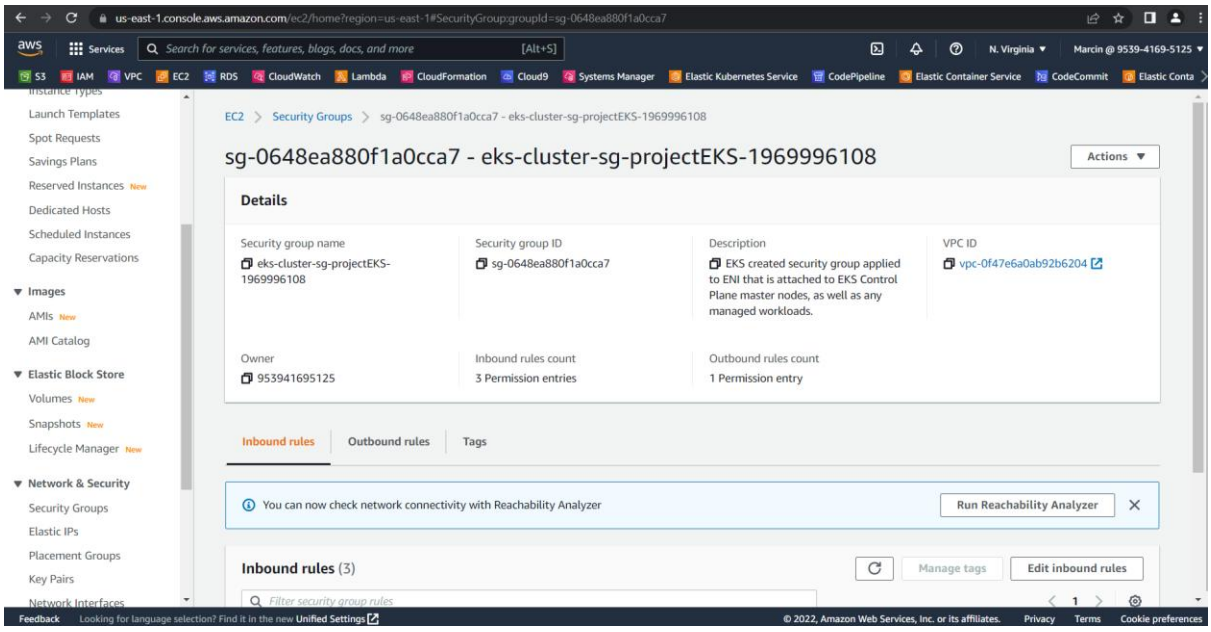


- AWS Security Groups

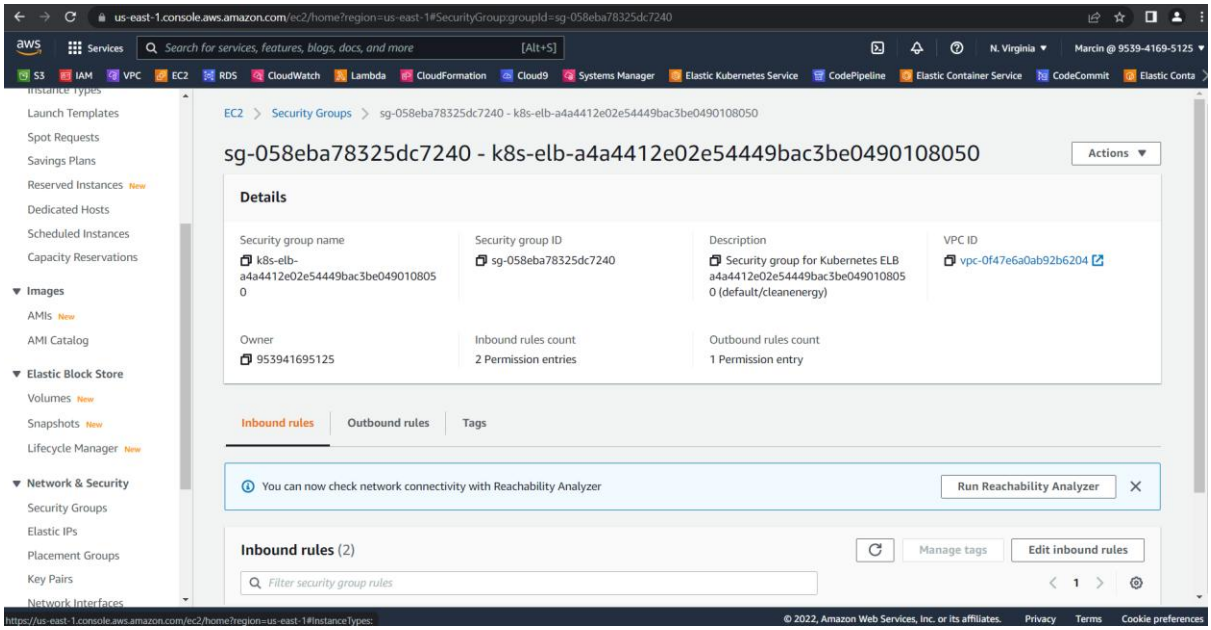
AWS Security Groups control the network traffic that is allowed to reach and leave resources that associates with it. For security reasons EKS nodes can allow inbound traffic just from EKS control plane, load balancer and database security groups. Database allow inbound traffic just from EKS nodes security group.

For this project 3 security groups were created:

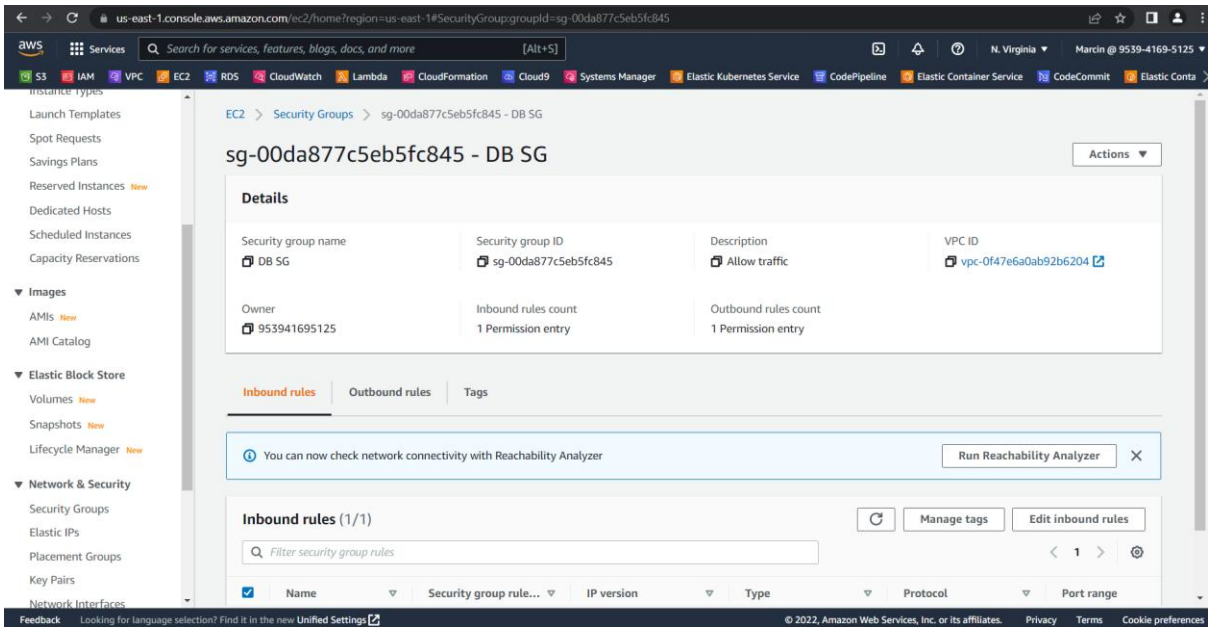
Elastic Kubernetes Service Nodes



Load balancer



RDS SQL Server database

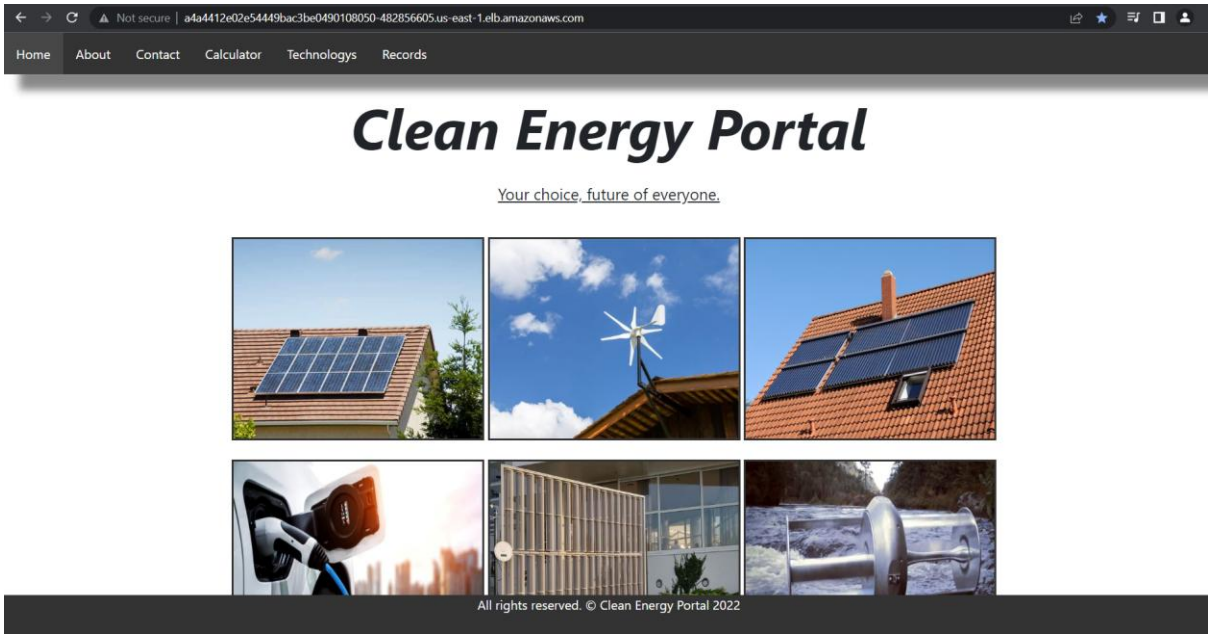


d. Application Design and Specification

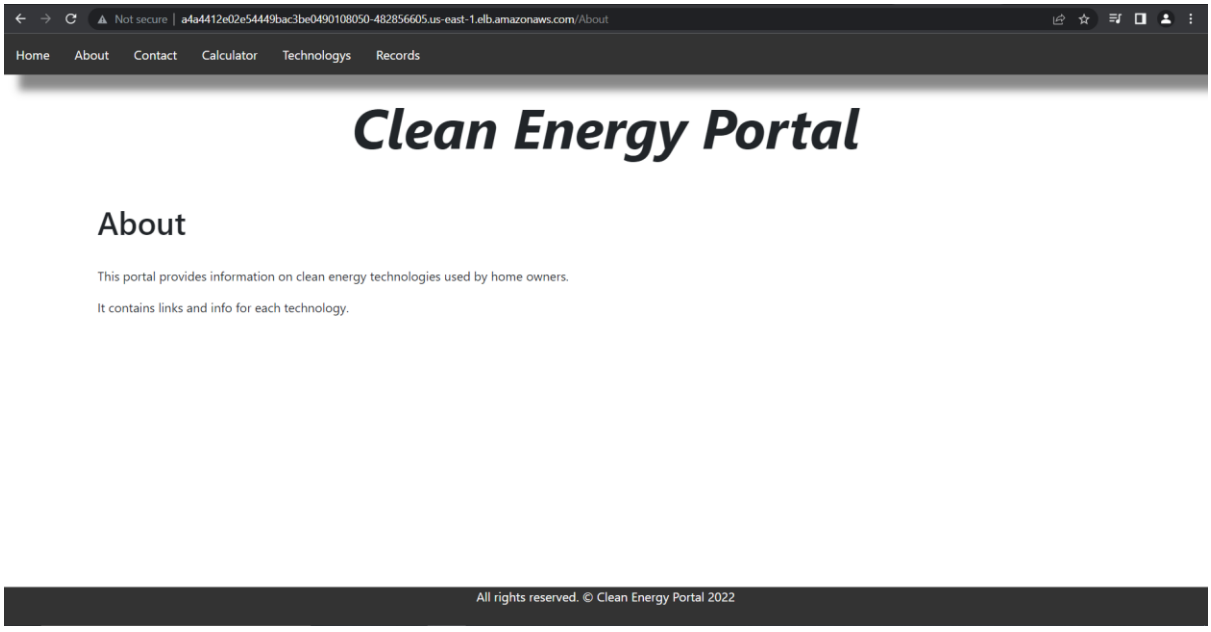
Application is written in C# ASP.NET core on Razor pages with CSS. Application contains 11 pages: Home, About, Contact, Records, Calculator, Solar Panels, Wind Turbines, Water Turbines, Solar Water Panels, Electric Cars, Fence Turbines. All pages have a navigation bar that allow to change pages and Home page has pictures that works as links to other pages. Each picture expands if hovering over it and displaying information about link, additional links on pages highlight.

Application is connected to AWS RDS database and stores information used by Calculator and results of calculation for future use. Records page display stored values in database but just after successful login on this page and is not visible for everyone that visit the site. Site is cross-web browser is working fine on Chrome, Internet Explorer, Firefox, Opera, Edge and probably others. It also works on any mobile devices.

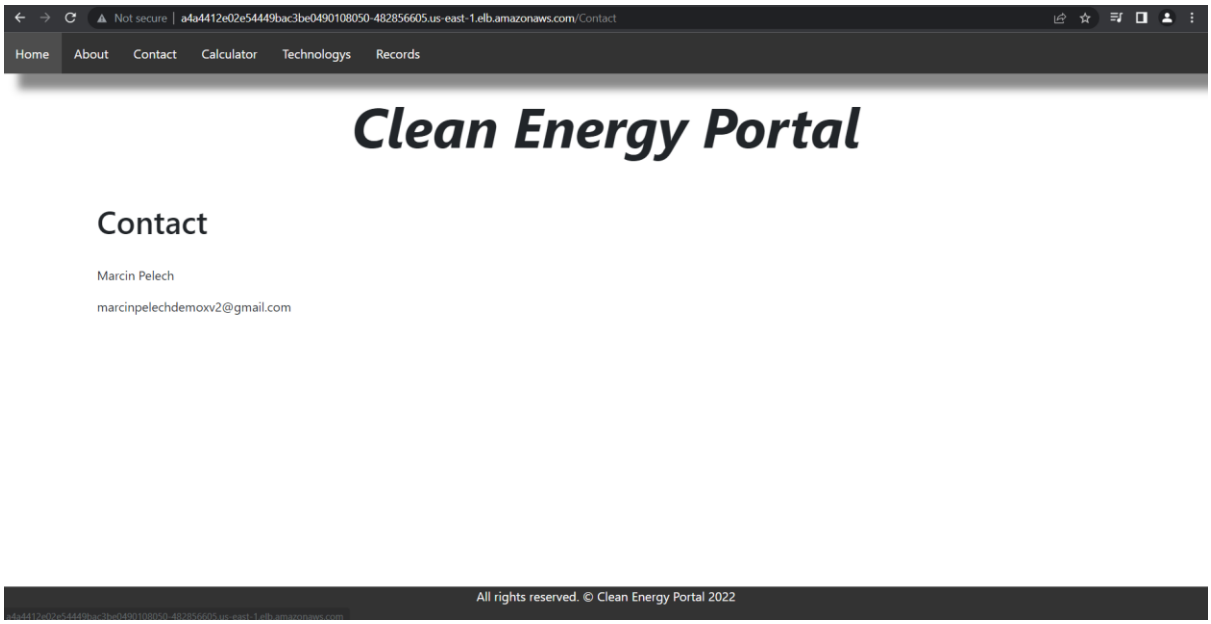
- Home page



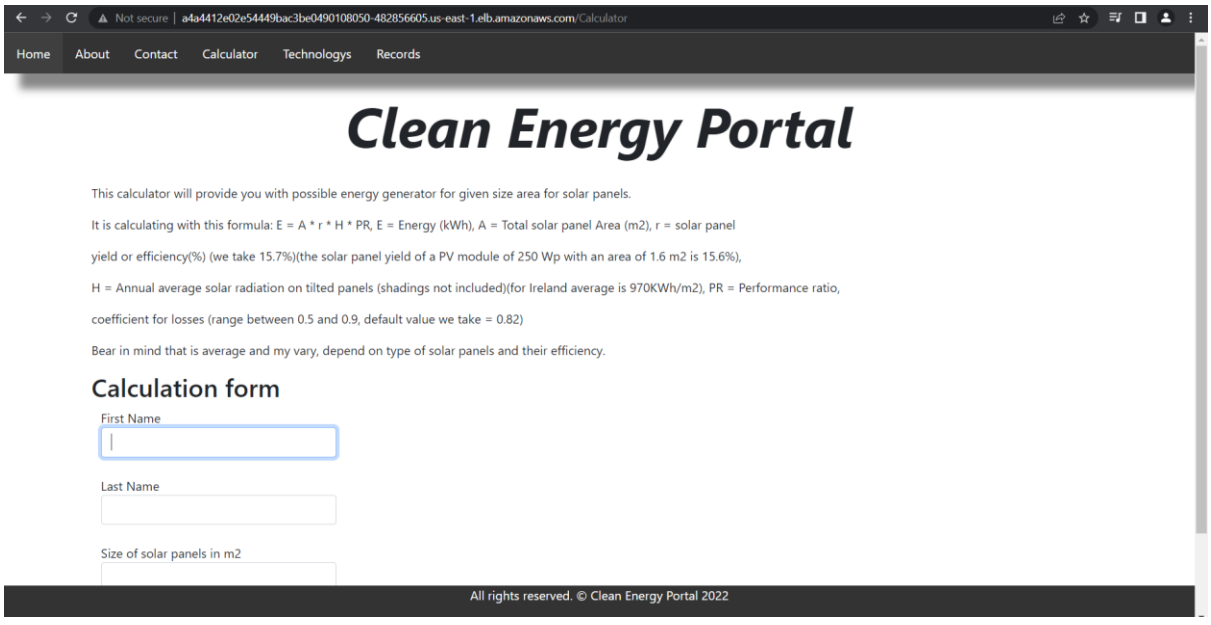
- About page



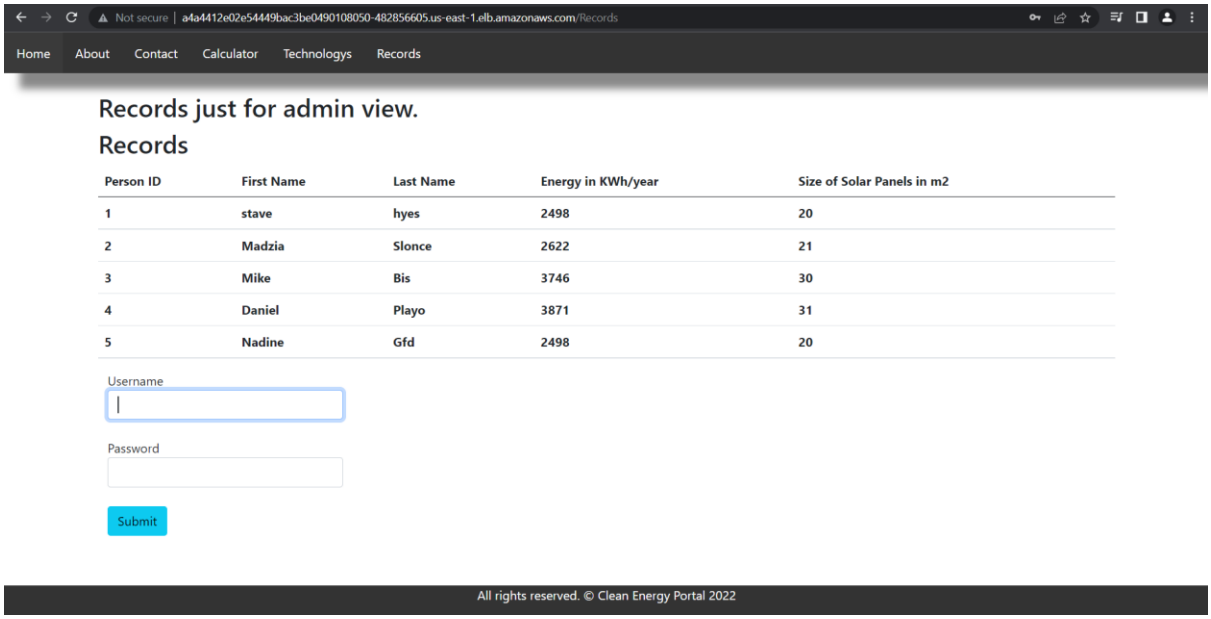
- Contact page

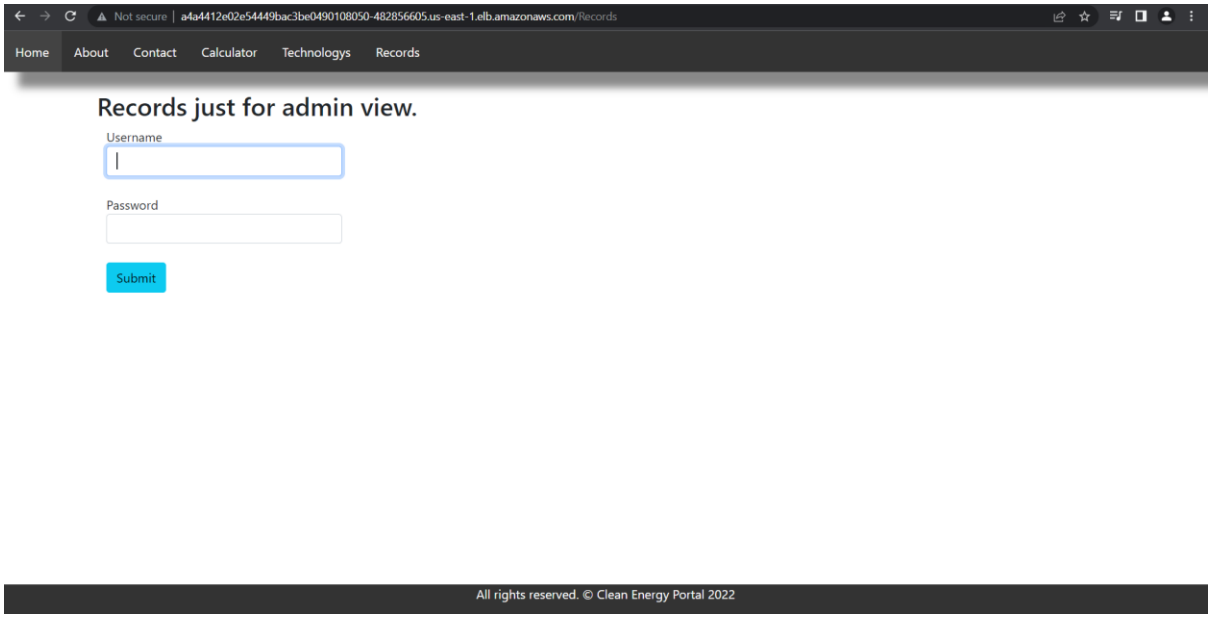


- Calculator page

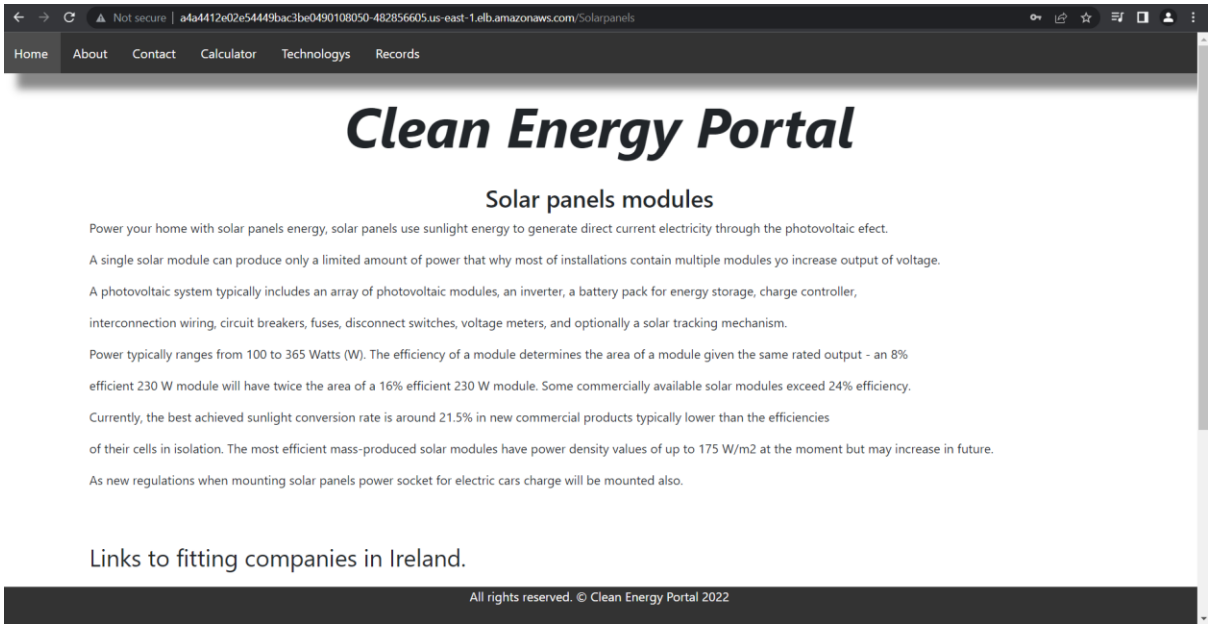


- Records

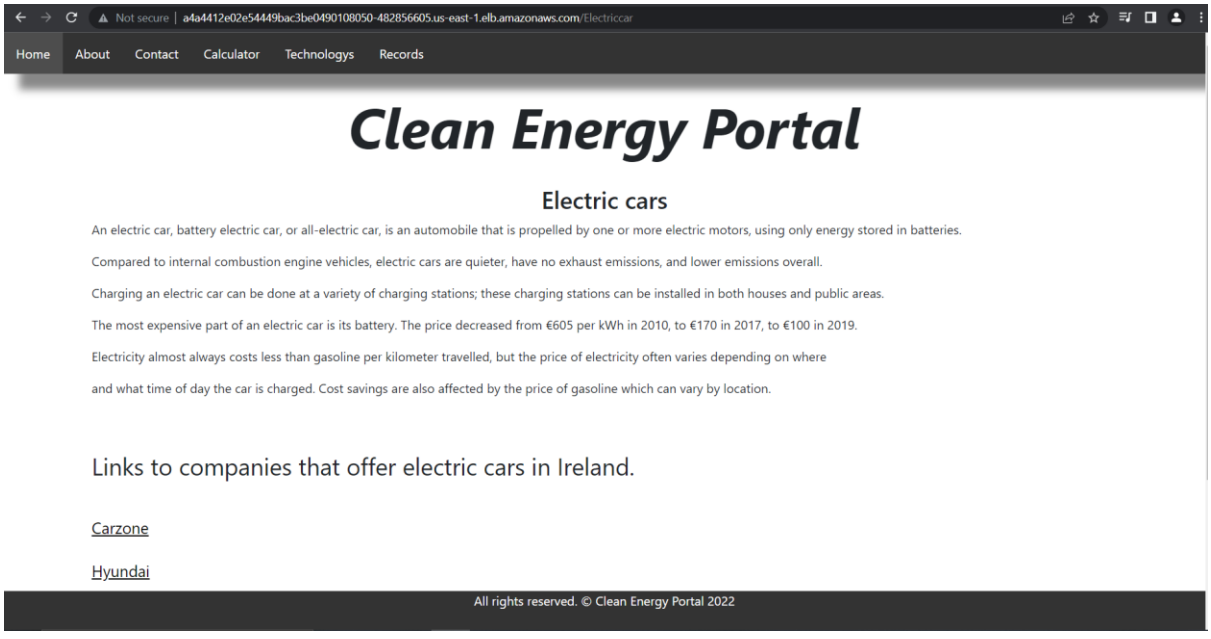




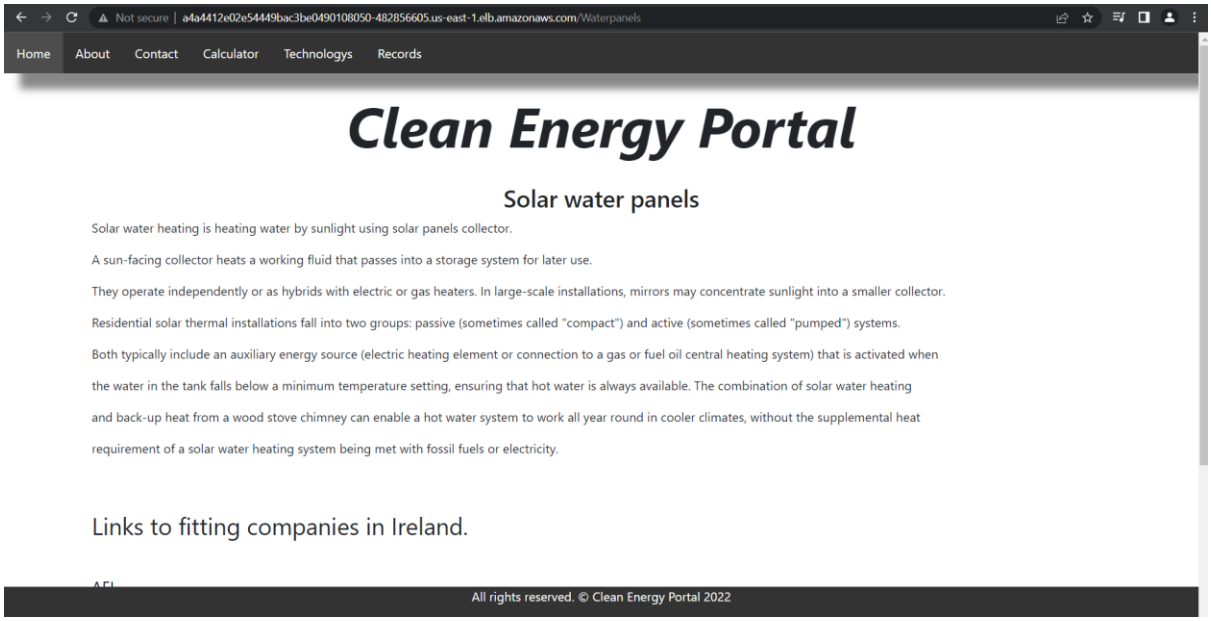
- Solar Panels



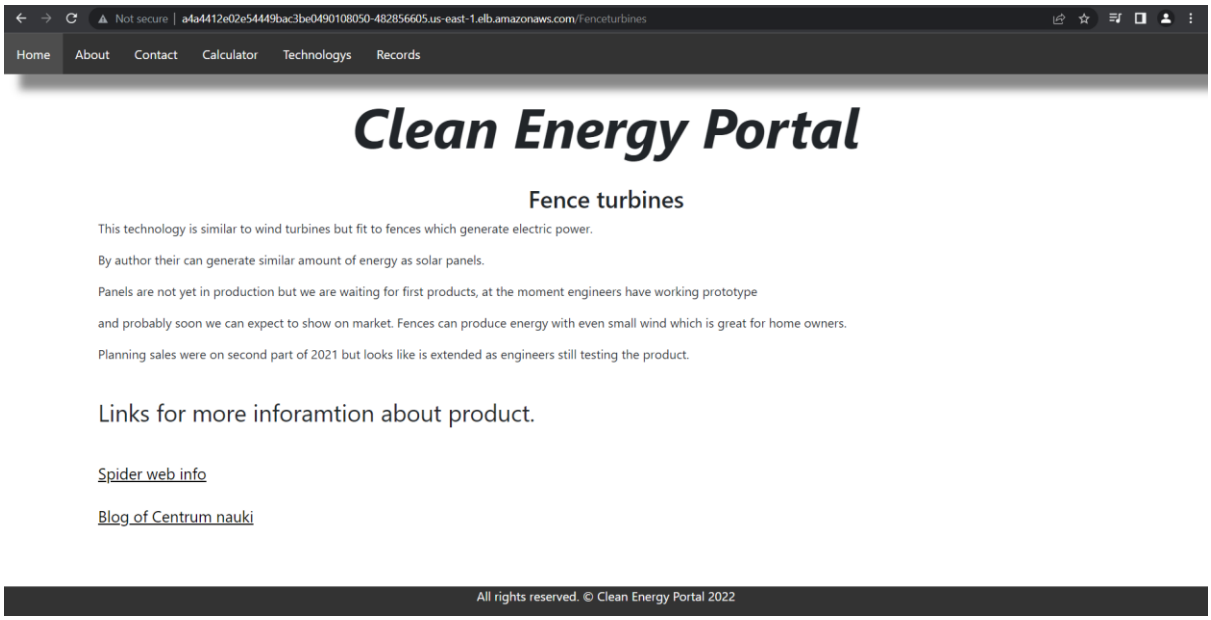
- Electric Cars



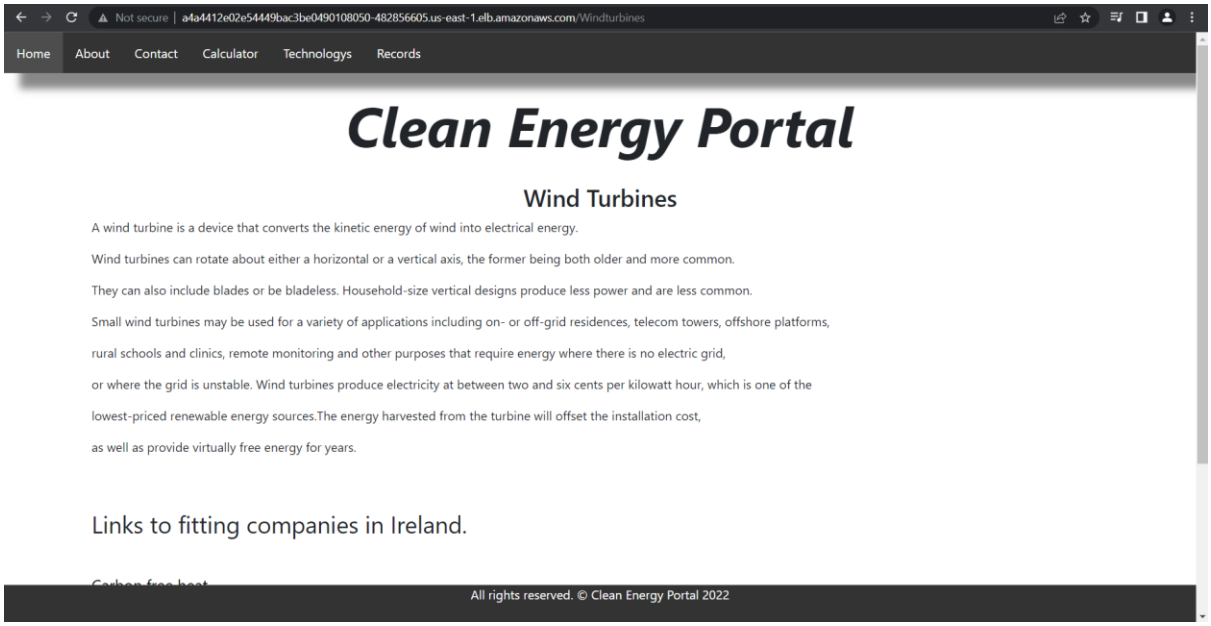
- Solar Water Panels



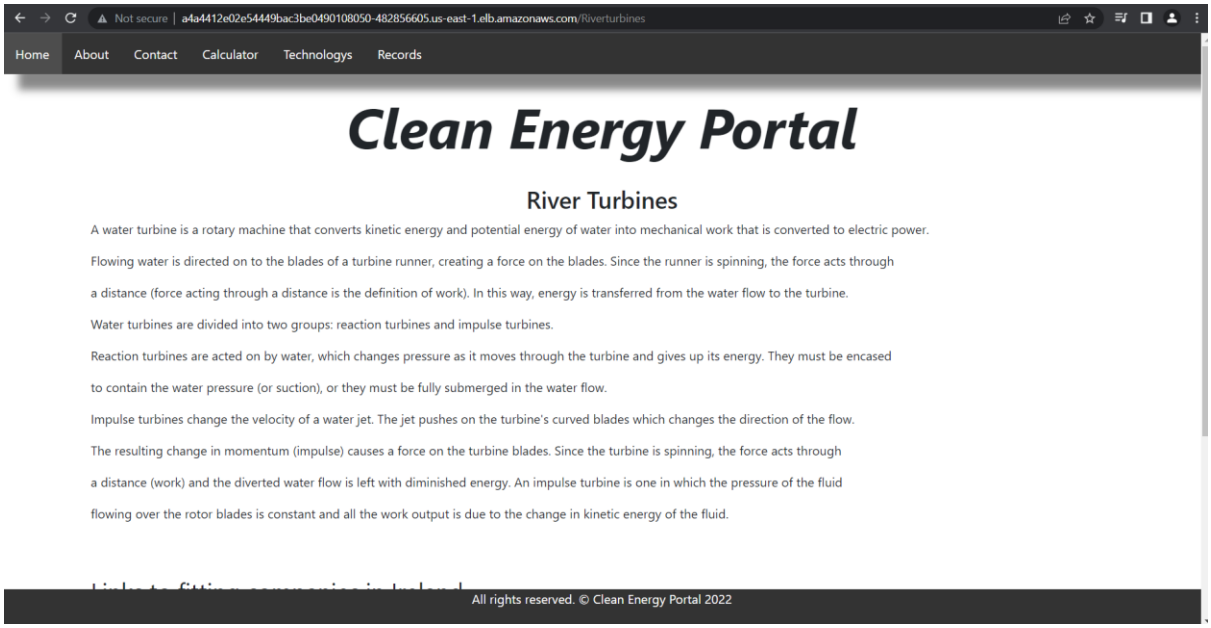
- Fence turbines



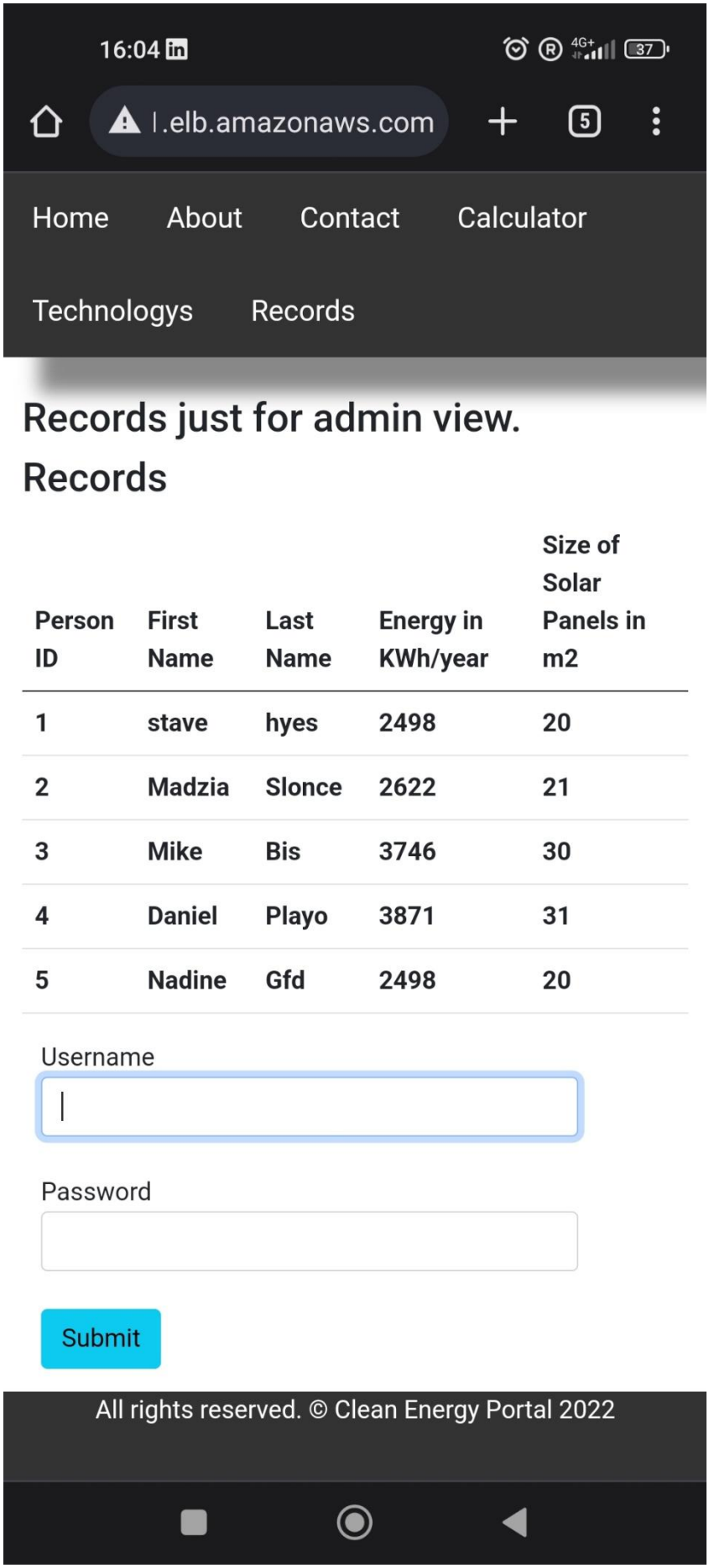
- Wind Turbines

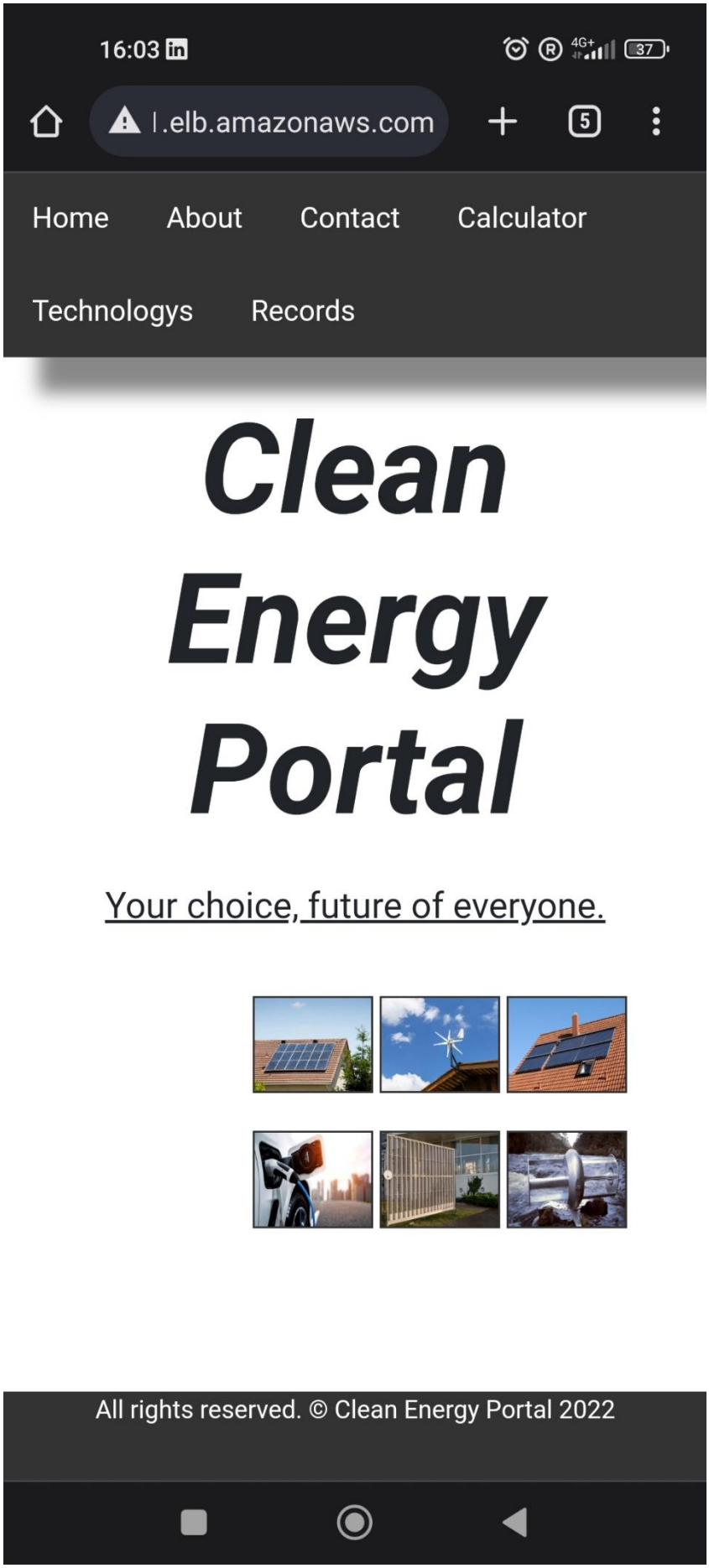


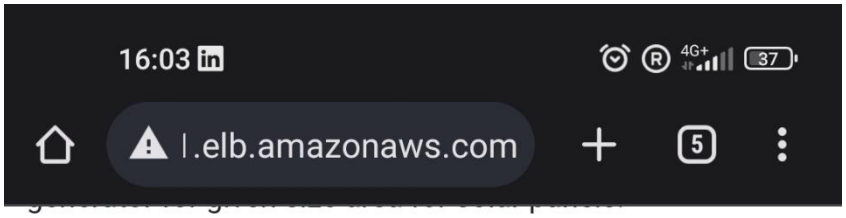
- River Turbines



- Mobile page's view


16:04
l.elb.amazonaws.com
Home
About
Contact
Calculator
Technologys
Records
Records just for admin view.
Records
Person ID
First Name
Last Name
Energy in KWh/year
Size of Solar Panels in m2
1 stave hyes 2498 20
2 Madzia Slonce 2622 21
3 Mike Bis 3746 30
4 Daniel Playo 3871 31
5 Nadine Gfd 2498 20
Username
Password
Submit
All rights reserved. © Clean Energy Portal 2022


16:03
l.elb.amazonaws.com
5
Home
About
Contact
Calculator
Technologys
Records
Clean Energy Portal
Your choice, future of everyone.
All rights reserved. © Clean Energy Portal 2022



It is calculating with this formula: E = A * r * H * PR, E = Energy (kWh), A = Total solar panel Area (m2), r = solar panel

yield or efficiency(%) (we take 15.7%)(the solar panel yield of a PV module of 250 Wp with an area of 1.6 m2 is 15.6%),

H = Annual average solar radiation on tilted panels (shadings not included)(for Ireland average is 970KWh/m2), PR = Performance ratio,

coefficient for losses (range between 0.5 and 0.9, default value we take = 0.82)

Bear in mind that is average and my vary, depend on type of solar panels and their efficiency.

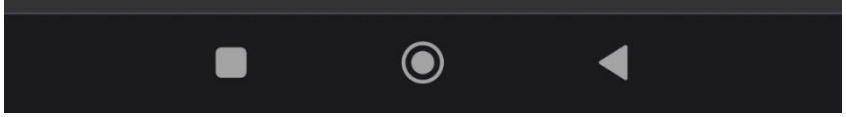
Calculation form

First Name

Last Name

Size of solar panels in m2

Submit



- _Layout.cshtml file to show CSS implementation for the website

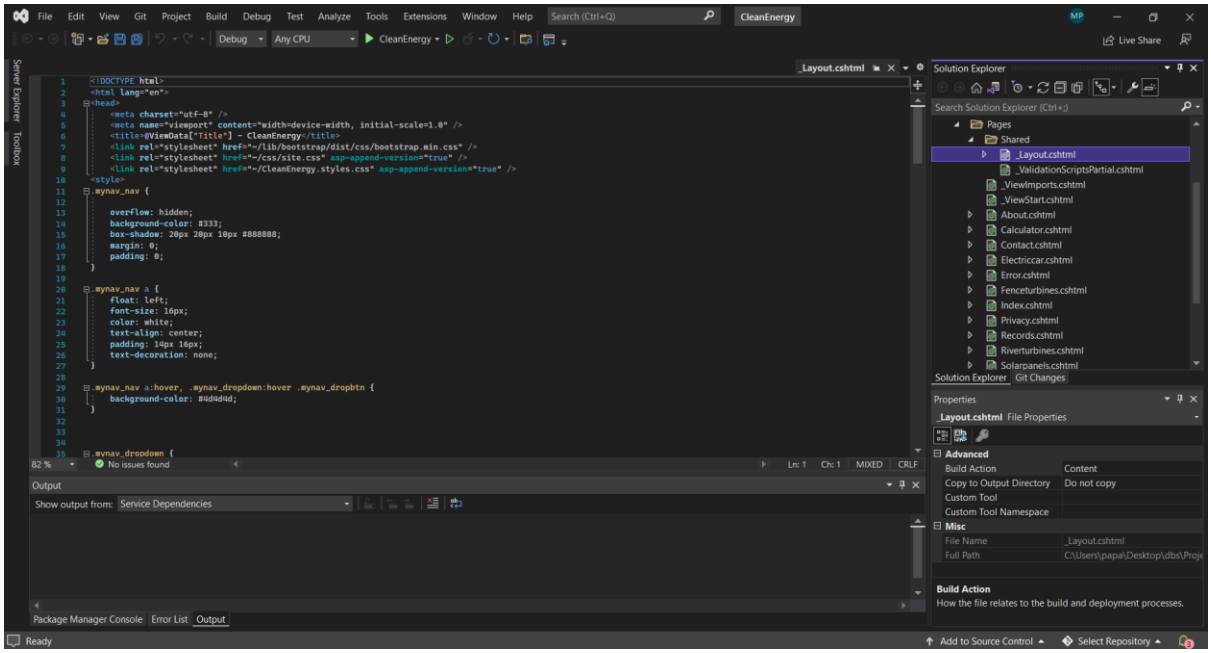
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>@ViewData["Title"] - CleanEnergy</title>
    <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
    <link rel="stylesheet" href="~/css/site.css" asp-append-version="true" />
    <link rel="stylesheet" href="~/CleanEnergy.styles.css" asp-append-
version="true" />
<style>
.mynav_nav {

    overflow: hidden;
    background-color: #333;
    box-shadow: 20px 20px 10px #888888;
    margin: 0;
    padding: 0;
}

.mynav_nav a {
    float: left;
    font-size: 16px;
    color: white;
    text-align: center;
    padding: 14px 16px;
    text-decoration: none;
}

.mynav_nav a:hover, .mynav_dropdown:hover .mynav_dropbtn {
    background-color: #4d4d4d;
}



.mynav_dropdown {
    float: left;
    overflow: hidden;
}

.mynav_dropdown .mynav_dropbtn {

    font-size: 16px;
    border: none;
    outline: none;
    color: white;
    padding: 14px 16px;
    background-color: inherit;
    font-family: inherit;
    margin: 0;
}

.mynav_dropdown-content {
    display: none;
    position: absolute;
    background-color: #f9f9f9;
    min-width: 160px;
```

```
    box-shadow: 0px 8px 16px 0px rgba(0,0,0,0.2);
    z-index: 1;
}

.mynav_dropdown-content a {
    float: none;
    color: black;
    padding: 12px 16px;
    text-decoration: none;
    display: block;
    text-align: left;
}

.mynav_dropdown-content a:hover {
    background-color: #ddd;
}

.mynav_dropdown:hover .mynav_dropdown-content {
    display: block;
}
.my_footer {
       list-style-type: none;
       background-color: #333;
       overflow: hidden;
       position: fixed;
       bottom: 0;
       color: white;
       text-align: center;
       height: 50px;
       width: 100%;

}
#pictures {
    display: block;
       width: 90%;
       height: 100%;
       object-fit: contain;
    margin-left: auto;
    margin-right: auto;
    padding-left: 110px;

}

#images {
    width: 30%;

    border: solid #333;

}

#images:hover {
    transform: scale(1.15);
}


h3 {
    text-align: center;
    font-size: 30px;
}

#links:hover {
    background-color: #33A8FF;
}
```

```
</style>

</head>
<body>
    <header>

        <div class="mynav_nav">

           <a class="nav-link text-white" asp-area="" asp-page="/Index">Home</a>
           <a class="nav-link text-white" asp-area="" asp-page="/About">About</a>
                <a class="nav-link text-white" asp-area="" asp-
page="/Contact">Contact</a>
           <a class="nav-link text-white" asp-area="" asp-
page="/Calculator">Calculator</a>
                <div class="mynav_dropdown">

                <button class="mynav_dropbtn">Technologys</button>
                <div class="mynav_dropdown-content">
                    <a class="nav-link text-dark" asp-area="" asp-
page="/Solarpanels">Solar Panels</a>
                    <a class="nav-link text-dark" asp-area="" asp-
page="/Windturbines">Wind Turbines</a>
                    <a class="nav-link text-dark" asp-area="" asp-
page="/Waterpanels">Water Panels</a>
                    <a class="nav-link text-dark" asp-area="" asp-
page="/Electriccar">Electric Cars</a>
                    <a class="nav-link text-dark" asp-area="" asp-
page="/Fenceturbines">Fence Turbines</a>
                    <a class="nav-link text-dark" asp-area="" asp-
page="/Riverturbines">River Turbines</a>
                </div>
            </div>
           <a class="nav-link text-white" asp-area="" asp-
page="/Records">Records</a>


             </div>
    </header>
    <div class="container">
        <main role="main" class="pb-3">
            @RenderBody()
        </main>
    </div>

    <footer>

                    <div class="my_footer">
                          <p>All rights reserved. &copy; Clean Energy Portal
2022</p>
                    </div>

      </footer>

    <script src="~/lib/jquery/dist/jquery.min.js"></script>
    <script src="~/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
    <script src="~/js/site.js" asp-append-version="true"></script>

    @await RenderSectionAsync("Scripts", required: false)
</body>
</html>
```

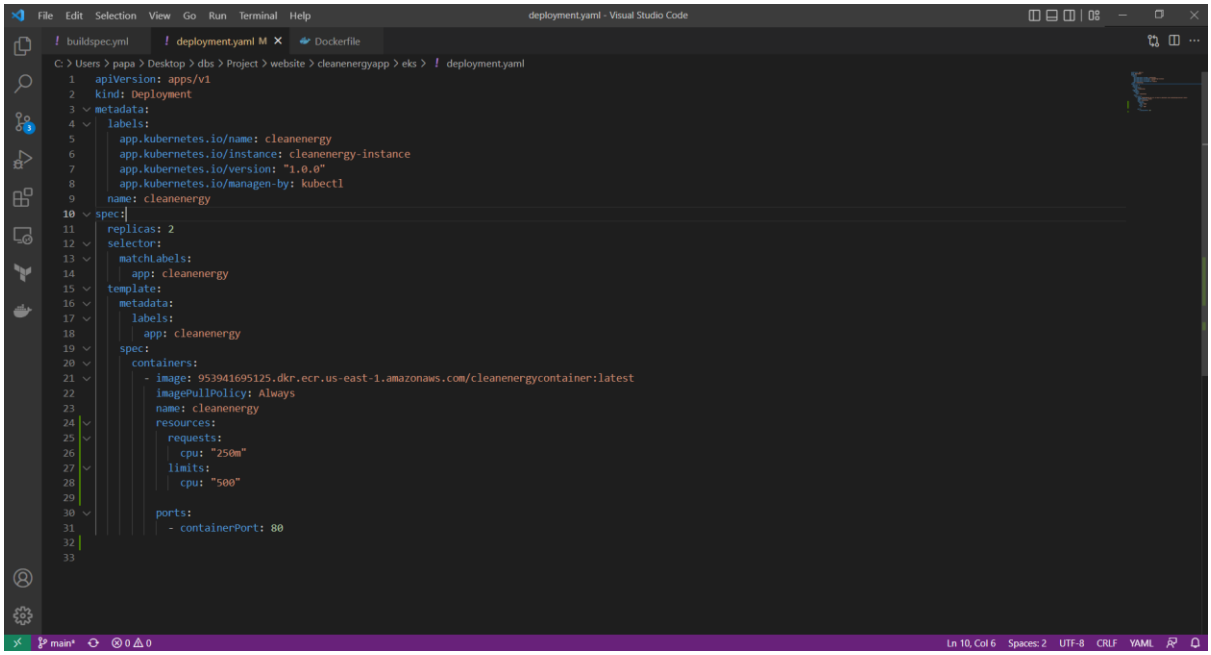
6. Application Development environment

For developing frontend of this application, Visual Studio 2022 was used and for backend SQL express with Microsoft SQL Server Management studio 18. To store code AWS CodeCommit was used. Visual Studio is widely used IDE for developing console and graphical interface applications, Windows Forms applications, websites, ASP.NET applications.



7. Infrastructure Development Environment

For this project Visual Studio Code was used, it is widely used code editor with support for development operations like debugging, task running, and version control. Consist of many various plugins and support many different programming languages.



8. Database Design

For this project AWS RDS SQL Server Database was used, it is relational database which contains highly structured tables, where each row reflects a data entity, and every column defines specific information field. Relational databases are build using the structured query language SQL to create, store, update or retrieve data.

Database records and table Person was created for propose of this project:

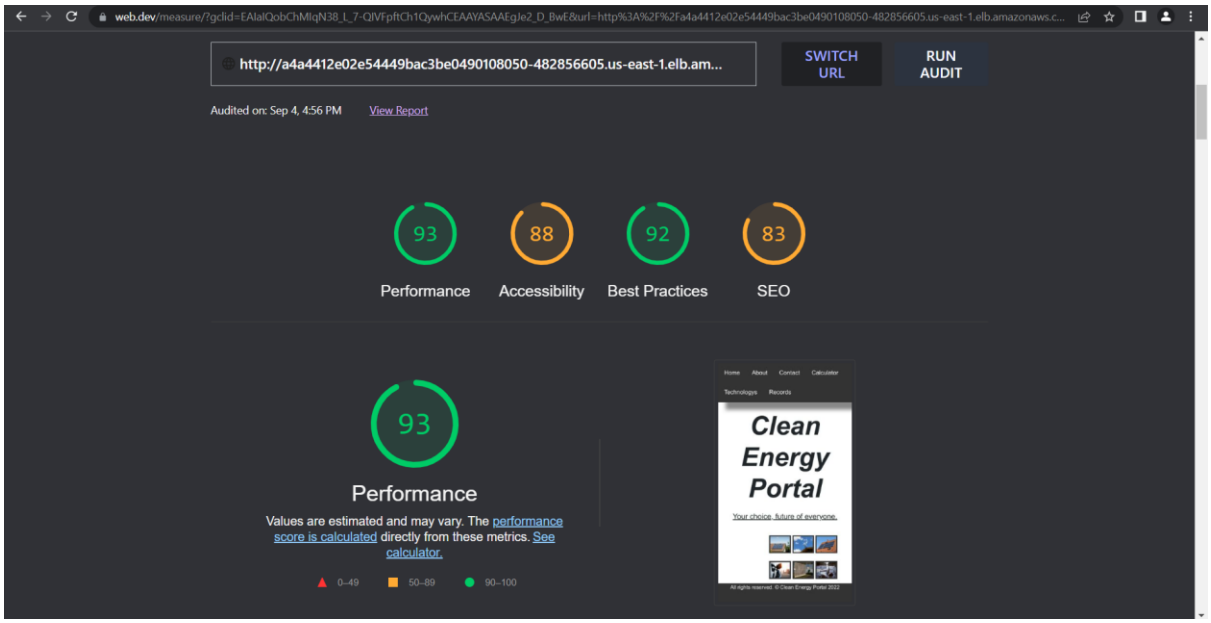
Column Name	Data Type	Allow Nulls
PersonID	Int	No
FirstName	Varchar(250)	Yes
LastName	Varchar(250)	Yes
Energy	numeric(30, 0)	Yes
Size	numeric(30, 0)	yes

9. Application Testing

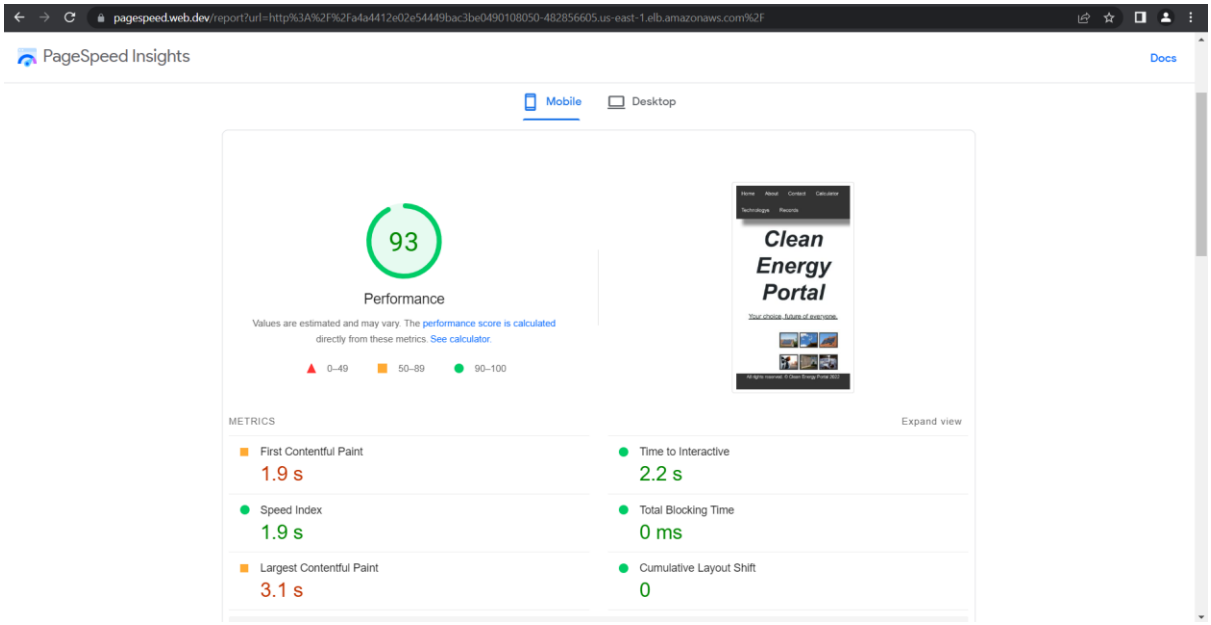
Purpose of testing is to give us confidence that such application is working correctly when interacting with users, has good performance, and all input forms work as intended.

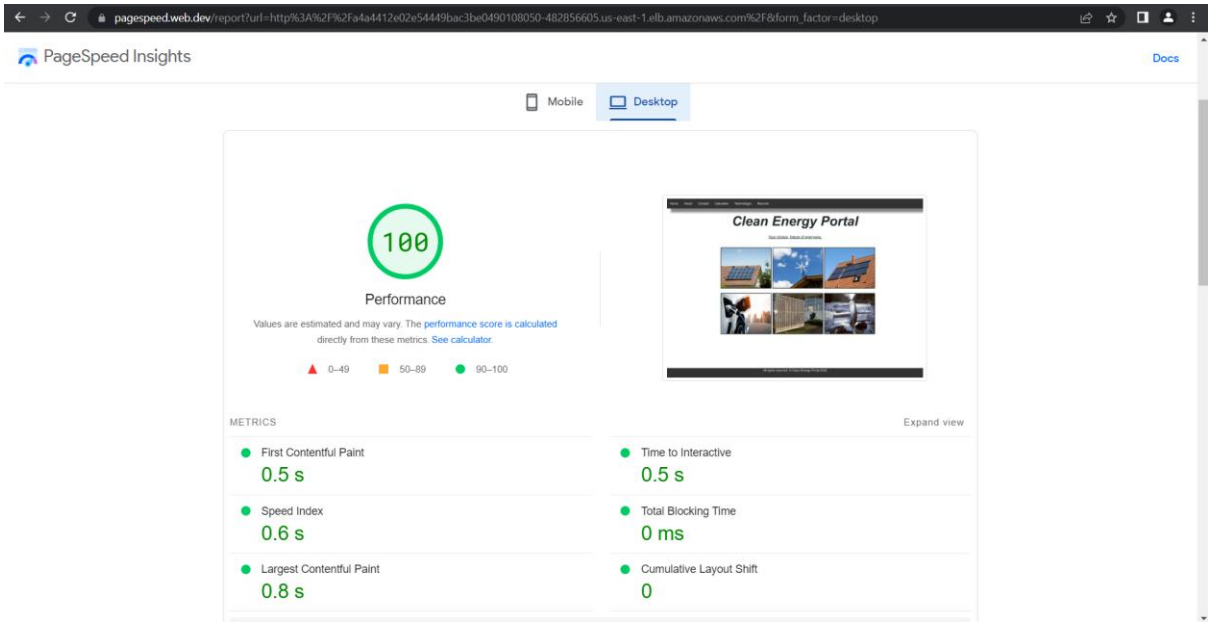
a. Performance testing

Testing with Lighthouse from Google, it is open-source tool for running technical website audits. This tool was developed by Google, and it analyses: Performance, Progressive Web App, Accessibility, Best Practices and SEO (search website optimization).



Testing with Page Speed Insight, this is an online synthetic benchmark tool which helps in identifying performance best practices.





b. Manual testing

In manual testing each link on website was tested and is working as supposed to:

Website link	Link	Working (yes or no)
Navigation bar home	asp-page="/Index"	yes
Navigation bar About	asp-page="/About"	Yes
Navigation bar Contact	asp-page="/Contact"	Yes
Navigation bar Records	asp-page="/Records"	Yes
Navigation bar Calculator	asp-page="/Calculator"	Yes
Navigation bar Solar Panels	asp-page=”/Solarpanels”	Yes
Navigation bar Solar Water Panels	asp-page="/Waterpanels"	Yes
Navigation bar Water Turbines	asp-page="/Waterpanels"	Yes
Navigation bar Wind Turbines	asp-page="/Windturbines"	Yes

Navigation bar Electric Car	asp-page="/Electriccar"	Yes
Navigation bar Fence Turbines	asp-page="/Fenceturbines"	Yes
Navigation bar River Turbines	asp-page="/Riverturbines"	Yes
Picture Solar Panels on home site	asp-page=”/Solarpanels”	Yes
Picture Electric Cars on home page	asp-page="/Electriccar"	Yes
Picture Water Turbines on home page	asp-page="/Riverturbines"	Yes
Picture Wind Turbines on home page	asp-page="/Windturbines"	Yes
Picture Fence Turbines on home page	asp-page="/Fenceturbines"	Yes
Picture Solar Water Panels on home page	asp-page="/Waterpanels"	Yes
Spider web info	https://spidersweb.pl/2022/08/plot-ktory-wytwarza-prad.html	Yes
Blog of Centrum nauki	https://centrumnauki.eu/prad-z-plotu-chodnika-i-okiennej-szyby/	Yes
BCD Energy	http://www.bcdenergy.ie/renewables-hydro-power.php	Yes
Suneco	https://www.micro-hydro-power.com/	Yes
Eco Evolution	http://www.ecoevolution.ie/small-scale-hydro.html	Yes
Caldosolar	https://caldorsolar.ie/	Yes
Wizer Energy	https://wizerenergy.ie/	Yes
AEI	https://www.aei.ie/	Yes
Solarstream	https://www.solarstream.ie/	Yes
AEI	https://www.aei.ie/	Yes
Pure Energy Technology	http://www.pet.ie/solar-heating.html	Yes
Glenn Dimplex Ireland	https://www.glendimplexireland.com/brands/dimplex/domestic-heating-systems/solar-thermal-hot-water-systems	Yes

Carbon free heat	https://carbonfreeheat.ie/wind-turbines-for-sale-ireland	Yes
Wind and Sun	https://www.windandsun.ie/product-category/wind-turbines/	Yes
Sun stream energy	https://sunstreamenergy.ie/utility/wind/	Yes
Sustainable Energy Authority Ireland	https://www.seai.ie/grants/home-energy-grants/	Yes

Testing Calculator form on Calculator page:

Test	Result	Pass
When no entries provided and click submit	First filed information that filed can't be empty	Yes
When no entries in last 2 fields	Second Filed information that field can't be empty	Yes
When no entries in last filed	Last filed information that can't be empty	Yes
When in last filed we try insert not number	Unable to place number	Yes
When in last field we try insert fractions	Information on filed "Please provide valid value. The two nearest numbers are x and x"	Yes
When all fields are correct entries	Information on web "Expected energy generation: x KWh per year."	Yes

Testing form on Records page:

Test	Result	Pass
Run page	Is not displaying records	Yes
Entering incorrect username or password	It is displaying on page "Wrong username or password"	Yes
Entering correct username and password	Displaying records from database on page	Yes

10. Infrastructure testing

Testing if it is possible to access database from internet by using Microsoft SQL Server Management Studio 18:

Error: "Cannot connect to databaseeksprojectv2.c0l2bn8ji4on.us-east-1.rds.amazonaws.com.

Additional information:

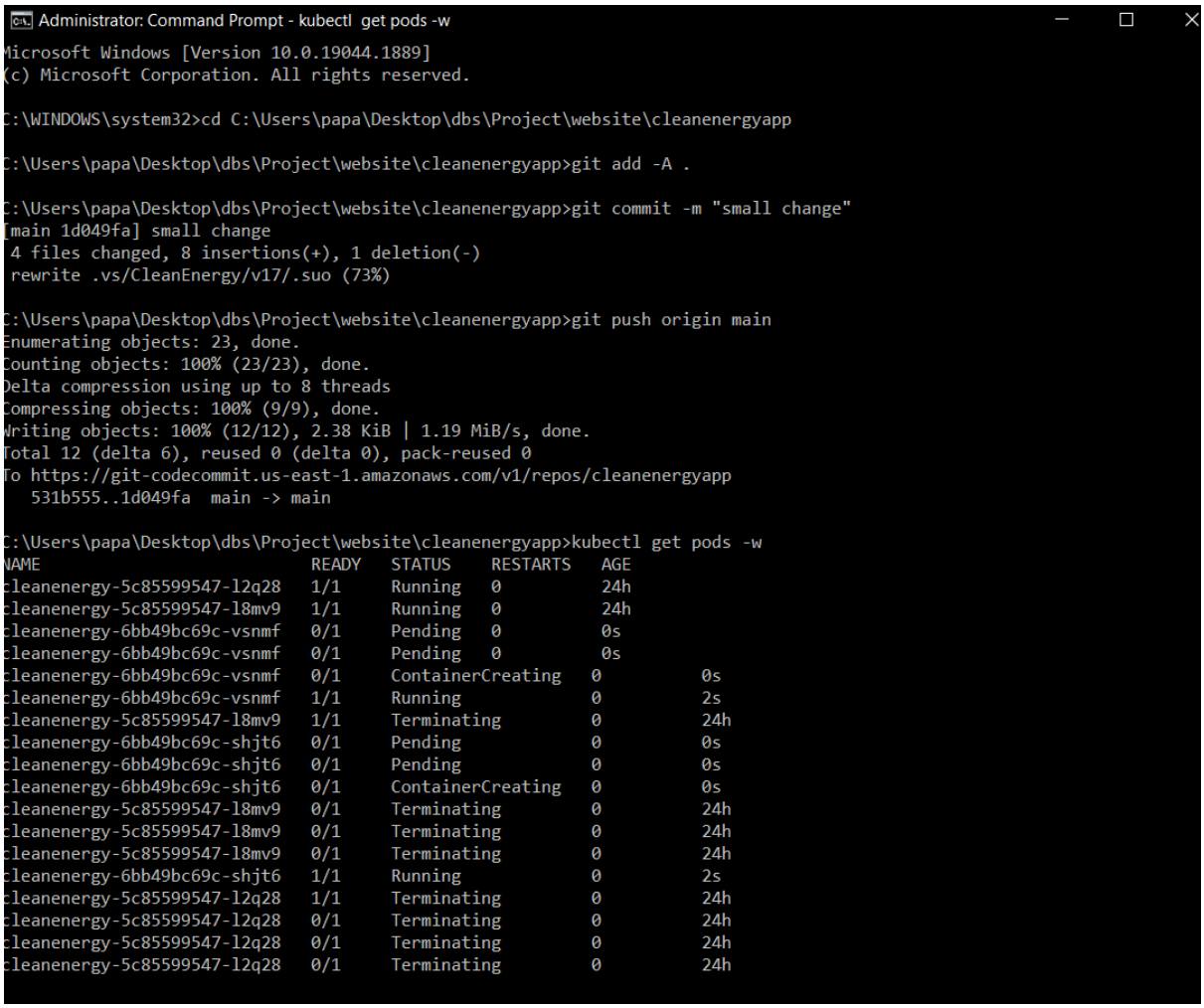
A network-related or instance-specific error occurred while establishing a connection to SQL Server. The Server was not found or was not accessible. Verify that the instance name is correct, and that SQL server is configured to allow remote connections. (Provider: Named Pipes Provider, error: 40 – Could not open a connection to SQL Server) (Microsoft SQL Server, Error: 53)

Network path was not found”

Testing if website is accessible: yes

Testing Continuous Integration and Continuous Deployment to see if is working as supposed to:

Doing small change to code and uploading to AWS CodeCommit and is working fine.



Testing if one of nodes fail another one will be deployed automatically, terminating one node and see if new is deployed to the EKS cluster, yes is working as intended, new node is deployed after 1-2 minutes.

11. Conclusions

Building system that is Highly Available, Secure, Redundant and Scalable, which consists automated procedures for continuous integration and continuous deployment was a big, exciting challenge. It was a fascinated adventure to bringing many different tools and building blocks to see like all these pieces work together as one system. During this project I have built first webapp in pure HTML, CSS and JavaScript realizing later that if application will be connected to database, every user can see code as JavaScript run on web browser, not on server and then I decided re-write application in ASP.NET core. Implementing Kubernetes orchestrator tool developed by Google (in AWS known as EKS) wasn't easy as it was new to me. Getting all features ready like Horizontal Pod Autoscaler which require additional pods logging to work, give me little bit of troubles and few moments of troubleshooting, finally finding that metrics-server need to be installed on Kubernetes cluster for autoscaling to work. It is nice to see how new technologies can reduce the time needed to deploy new updates or changes in your applications, by using containerization, continuous integration and continuous deployment time is reduced from days to few minutes. Developers can make changes in code many times a day and tested in minutes, if something is wrong with application previous working docker image can be deployed in minutes.

12. References / Bibliography

Industrial IT and Automation, 2021: https://www.youtube.com/watch?v=iKmUMgZj-cE

BoostMyTool, 2022: https://www.youtube.com/watch?v=YUPg41kG_kw

W3schools, 2022: https://www.w3schools.com/asp/default.asp

Sandip Das, 2019: https://www.youtube.com/watch?v=LlisKI-gN5w

Sandip Das, 2021: https://www.youtube.com/watch?v=nEK7e0QUVio

AWS, 2022: https://docs.aws.amazon.com/eks/latest/userguide/getting-started.html

AWS, 2022: https://docs.aws.amazon.com/eks/latest/userguide/add-user-role.html

AWS, 2022: https://docs.aws.amazon.com/eks/latest/userguide/metrics-server.html

Saur Energy International, 2022: https://www.saurenergy.com/solar-energy-blog/here-is-how-you-can-calculate-the-annual-solar-energy-output-of-a-photovoltaic-system

AWS, 2022: https://docs.aws.amazon.com/whitepapers/latest/overview-deployment-options/amazon-elastic-kubernetes-service.html

AWS, 2022: https://docs.aws.amazon.com/eks/latest/userguide/metrics-server.html

AWS, 2022: https://aws.amazon.com/codecommit/

AWS, 2022: https://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-parameter-store.html

AWS, 2022: https://aws.amazon.com/codepipeline/

AWS, 2022: https://aws.amazon.com/codebuild/

AWS, 2022: https://aws.amazon.com/ecr/

AWS, 2022: https://aws.amazon.com/ecr/

AWS, 2022: https://aws.amazon.com/cloudwatch/

AWS, 2022: https://aws.amazon.com/iam/

Solargis, 2022: https://solargis.com/maps-and-gis-data/download/ireland