**DevOps Scenario**

Please create an infrastructure solution diagram using AWS as the chosen cloud platform. In your diagram, ensure that you include the names of the AWS services you use. You can utilize your preferred tools, such as draw.io or others. Your diagram should encompass the following components:

1. **Backend:**
   * Two API Servers (e.g., API Server, File Service)
   * Implement autoscaling for one of the API servers
   * RDS (Relational Database Service)
   * Redis Server
   * Include components necessary for configuring SSL certificates and domains with the servers.
2. **Frontend:**
   * Implement Amazon CloudFront for the frontend
   * Describe how you plan to restrict access to the frontend.
3. **Continuous Integration/Deployment:**
   * Include an option for CI/CD in the diagram.
4. **Networking:**
   * Illustrate all essential network components.
   * Write terraform code VPC configuration.

Please ensure your diagram is comprehensive and clearly represents the AWS infrastructure solution you intend to design

Instruction

**Candidate Deliverables:**

1. Cost estimation & Work breakdown structure (WBS)
2. Diagram
3. Implementation
4. Readme file for implementation guide.
5. GitHub link for access terraform code.

Design:

Certainly! Let’s break down the components and their connections step by step for your AWS infrastructure solution diagram:

1. **Backend**:
   * **API Servers**:
     + **API Server A** and **API Server B** are part of the backend.
     + They handle incoming requests from clients.
     + **API Server A** is set up with **autoscaling** to dynamically adjust capacity based on demand.
     + Both servers connect to the RDS database and Redis server.
   * **RDS (Relational Database Service)**:
     + The RDS instance stores data for your application.
     + It is connected to both API servers.
   * **Redis Server**:
     + The Redis cluster provides caching and fast data retrieval.
     + Both API servers use Redis for performance optimization.
   * **SSL Certificates and Domains**:
     + The API servers need SSL certificates for secure communication.
     + Use ACM to manage SSL certificates.
     + Associate the certificates with your custom domains.
2. **Frontend**:
   * **Amazon CloudFront**:
     + CloudFront acts as a CDN for the frontend.
     + It distributes content globally, reducing latency.
     + Connect CloudFront to your S3 bucket or API Gateway.
   * **Access Restriction**:
     + Configure CloudFront to restrict access:
       - Create an **Origin Access Identity (OAI)**.
       - Associate the OAI with your S3 bucket or API Gateway.
       - Set up a CloudFront behavior to allow access only from the OAI.
       - Optionally, use AWS WAF for additional security.
3. **Continuous Integration/Deployment (CI/CD)**:
   * **AWS CodePipeline**:
     + Set up a pipeline for automated CI/CD.
     + Connect it to your source code repository (e.g., GitHub).
     + Define stages for building, testing, deploying, and monitoring.
   * **AWS CodeBuild**:
     + CodeBuild builds and packages your application.
     + Specify build commands and environment settings.
   * **AWS Elastic Beanstalk** (optional):
     + Deploy web applications easily using Elastic Beanstalk.
4. **Networking**:
   * **Amazon VPC (Virtual Private Cloud)**:
     + Create a VPC to isolate resources.
     + Define public and private subnets.
   * **Subnet Configuration**:
     + Place API servers, RDS, and Redis in private subnets.
     + Associate the public subnet with CloudFront.
   * **Security Groups and Network ACLs**:
     + Configure security groups for inbound/outbound traffic.
     + Use network ACLs for subnet-level rules.
   * **Route Tables**:
     + Define route tables for public and private subnets.
     + Route public traffic through the Internet Gateway.
     + Route private traffic through NAT Gateways or instances.
5. **Terraform Configuration**:
   * Write Terraform code for VPC, subnets, security groups, etc.
   * Define your infrastructure as code (IaC) using Terraform modules.

Remember to adapt this overview to your specific requirements and create your detailed AWS architecture diagram using your preferred tool! 🏢🌟