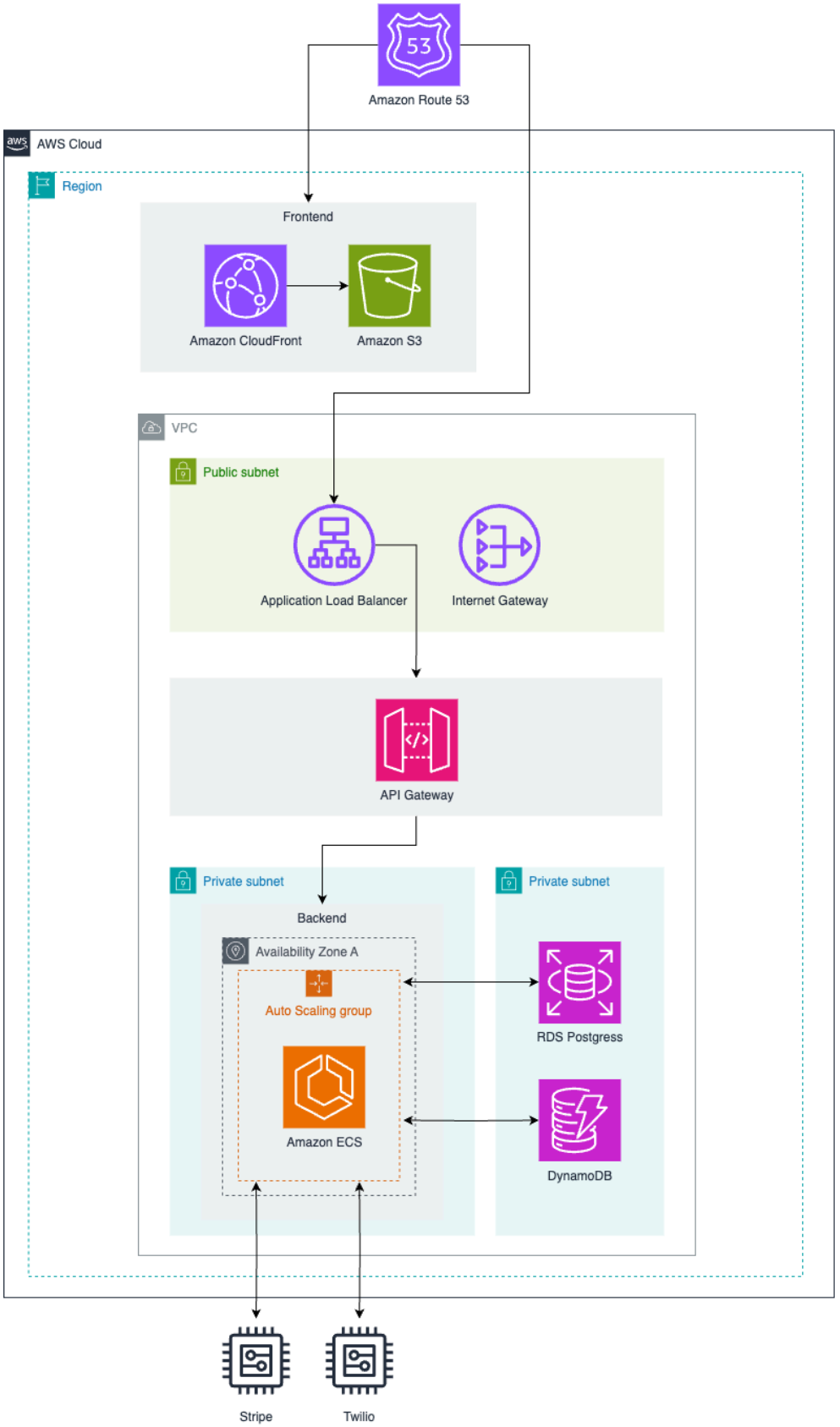
Prueba 1 - Diagrama de Red

Para el primer punto se solicitó diagramar la arquitectura de una aplicación web en AWS. La misma cuenta con frontend en JS y backend conectado a bases de datos relacional y no relacional. Debe soportar cargas variables, alta disponibilidad y consumir dos microservicios externos.

Es importante mencionar que, por razones de simplificar el diagrama, no se dibujaron elementos de alta disponibilidad tales como multiregión y multi-AZ, ni tampoco se tuvieron en cuenta cuestiones inherentes a la optimización de costos en AWS. Se propone el siguiente diagrama de alto nivel:



Si tiene problemas para visualizar el diagrama, puede encontrar el archivo original aquí.

Justificación de la Propuesta

A continuación, se detalla brevemente el porqué de la elección de cada componente de la arquitectura.

Frontend

- Amazon S3: se asume una aplicación en React, compilada, por lo cual el código se almacena en un Bucket S3.
- Amazon CloudFront: se propone utilizar una CDN (red de distribución de contenido) para reducir la latencia y obtener alta disponibilidad en los assets estáticos.

Backend

- Amazon ECS (con Auto Scaling Group): se propone la utilización del servicio de ECS como orquestador de contenedores para implementar, administrar y escalar la aplicación.
- Application Load Balancer (ALB): se sugiere la utilización de un balanceador de carga de aplicación para distribuir la carga de trabajo entre las instancias ECS del backend.
- Amazon RDS: para que la aplicación cuente con una BD relacional, se propone utilizar el servicio RDS con el motor PostgreSQL.
- Amazon DynamoDB: para el manejo de datos no relacionales se optó por el servicio administrado de BD noSQL DynamoDB.
- Private Subnets dentro de una VPC (Virtual Private Cloud): se sugiere utilizar subnets privadas para aislar el tráfico del backend sólo entre los componentes deseados.

Microservicios Externos

- Internet Gateway: se despliega un Internet Gateway en una subred pública para la comunicación entre los microservicios externos y el backend.

Otros

- API Gateway: se sugiere utilizar éste servicio administrado para la gestión y acceso a datos de la aplicación a través de APIs. Al ser un servicio administrado no existen limitaciones en cuanto a escalabilidad.
- Amazon Route 53: se propone utilizar el servicio de DNS de AWS para la gestión de un potencial nombre de dominio y el ruteo de las solicitudes a la infraestructura en AWS.