Step 1: Create an EC2 Instance for Backend & Frontend

1. Go to **AWS Console** → **EC2** → **Launch Instance**.
2. Choose **Amazon Linux 2 AMI**.
3. Instance Type: `t2.micro` (or higher if needed).
4. **VPC & Subnet**:
    a. VPC: `test-vpc`.
    b. Subnet: `test-public-subnet-2a`.
    c. Enable **Auto-assign Public IP**.
5. **Security Group Rules**:
    a. SSH (22) → Your IP.
    b. HTTP (80) & HTTPS (443) → Anywhere.
    c. Custom TCP 3000, 5000 → Anywhere (or restrict).
6. **Attach an Elastic IP** to ensure a static public IP.

Step 2: Create an RDS MySQL Database

1. Go to **AWS Console** → **RDS** → **Create Database**.
2. Choose **MySQL** → Free Tier.
3. DB Name: `your-database`.
4. Username: `your-user`, Password: `your-password`.
5. **VPC & Subnet**:
    a. Select `test-vpc`.
    b. Choose `test-private-subnet-2a` (to keep it private).
6. **Security Group**:
    a. Create a new group allowing MySQL (3306) from the EC2 backend.

Step 3: Store Secrets Securely in AWS Secrets Manager

1. Go to **AWS Console** → **Secrets Manager** → **Store a new secret**.

2. Choose **Other type of secret**.
3. Add key-value pairs (JSON format):

```
{
  "DB_HOST": "your-rds-endpoint",
  "DB_USER": "your-user",
  "DB_PASS": "your-password",
  "DB_NAME": "your-database",
  "JWT_SECRET": "your-secure-secret"
}
```

4. Name the secret: `backend-secrets`.
5. Attach IAM policy to EC2 role:
    a. Go to **IAM** → **Roles** → Select EC2 role.
    b. Attach this policy:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["secretsmanager:GetSecretValue"],
      "Resource": "arn:aws:secretsmanager:your-region:your-
account-id:secret:backend-secrets-*"
    }
  ]
}
```

Step 4: Install Required Software on EC2

SSH into EC2:

```
ssh -i your-key.pem ec2-user@your-ec2-public-ip
```

Install required packages:

```
sudo yum update -y
sudo yum install -y gcc-c++ make
curl -fsSL https://rpm.nodesource.com/setup_18.x | sudo bash -
sudo yum install -y nodejs nginx certbot python3-certbot-nginx
```

Step 5: Deploy Backend (Express.js)

1. **Clone your backend repo**:

   ```
   git clone https://github.com/your-repo/backend.git
   cd backend
   npm install
   ```

2. **Retrieve Secrets from AWS Secrets Manager**:

   ```
   const AWS = require("aws-sdk");
   AWS.config.update({ region: "your-region" });
   const secretsManager = new AWS.SecretsManager();
   async function getSecrets() {
       const secretData = await
   secretsManager.getSecretValue({ SecretId: "backend-
   secrets" }).promise();
       return JSON.parse(secretData.SecretString);
   }
   (async () => {
       const secrets = await getSecrets();
       process.env.DB_HOST = secrets.DB_HOST;
       process.env.DB_USER = secrets.DB_USER;
       process.env.DB_PASS = secrets.DB_PASS;
       process.env.DB_NAME = secrets.DB_NAME;
       process.env.JWT_SECRET = secrets.JWT_SECRET;
       require("./server");
   })();
   ```

3. **Start the backend with PM2**:

```
pm2 start server.js
pm2 save
pm2 startup
```

Step 6: Deploy Frontend (React)

1. **Clone frontend repo**:

```
git clone https://github.com/your-repo/frontend.git
cd frontend
npm install
npm run build
```

2. **Move build files to Nginx root**:

```
sudo mv build /var/www/frontend
```

Step 7: Configure Nginx Reverse Proxy

1. **Edit Nginx config**:

```
sudo nano /etc/nginx/nginx.conf
```

2. **Update with**:

```
server {
    listen 80;
    server_name yourdomain.com www.yourdomain.com;
    location / {
        root /var/www/frontend;
        index index.html;
```

```
            try_files $uri /index.html;
        }
        location /api/ {
            proxy_pass http://localhost:5000/;
            proxy_set_header Host $host;
            proxy_set_header X-Real-IP $remote_addr;
            proxy_set_header X-Forwarded-For
    $proxy_add_x_forwarded_for;
            proxy_set_header X-Forwarded-Proto $scheme;
        }
    }
```

3. **Restart Nginx**:

```
sudo systemctl restart nginx
```

Step 8: Secure with SSL (HTTPS)

1. **Ensure domain points to EC2**.
2. **Run Certbot**:

```
sudo certbot --nginx -d yourdomain.com -d www.yourdomain.com
```

3. **Test SSL**:

```
sudo certbot renew --dry-run
```

Step 9: Secure the Server

1. **Configure Firewall (UFW)**:

```
sudo yum install -y firewalld
sudo systemctl enable firewalld
sudo systemctl start firewalld
sudo firewall-cmd --permanent --add-service=http
```

```
sudo firewall-cmd --permanent --add-service=https
sudo firewall-cmd --permanent --add-service=ssh
sudo firewall-cmd --reload
```

2. **Disable Root SSH Login**:

```
sudo nano /etc/ssh/sshd_config
```

Set:

```
PermitRootLogin no
```

Restart SSH:

```
sudo systemctl restart sshd
```