

AWS PROJECT

📌 Project Documentation: Full-Stack Project Deployment on AWS

✳️ Project Description

This project focuses on deploying a **Full-Stack Project** using **AWS cloud services** following industry-level deployment standards. The application codebase is already developed — our primary objective is to **configure, integrate, and deploy** the full architecture on AWS.

🛠️ Tech Stack Summary

✓ Frontend

- **Next.js** – React framework for production-ready UI
- **Tailwind CSS** – utility-first styling
- **Material UI Data Grid** – advanced tabular data management
- **Redux Toolkit + RTK Query** – state and API data handling

✓ Backend

- **Node.js + Express.js** – REST API services
- **PostgreSQL** – relational database
- **Prisma ORM** – database mapping and migrations
- **PgAdmin** – database GUI for monitoring

🎯 Project Goal

To deploy a **secure, scalable, and maintainable** web application on AWS, using serverless principles and managed authentication. The deployment process includes detailed, step-by-step guidance with screenshots to avoid misconfigurations.

💡 Setup & Installation:

Follow these steps before deploying to AWS:

✓ 1 Clone & Initialize the Repository(<https://github.com/ITS-NAYAN/project-management.git>)

Download .zip — If you received a ZIP folder

1. Create a folder named **project-management**
2. Extract ZIP contents into the folder
3. Open the folder in **VS Code**

Then initialize Git:

```
git init
```

```
git add .
```

```
git commit -m "Initial commit"
```

Now, create a new GitHub repository:

➡ <https://github.com/new>

After creating the repo, connect and push code:

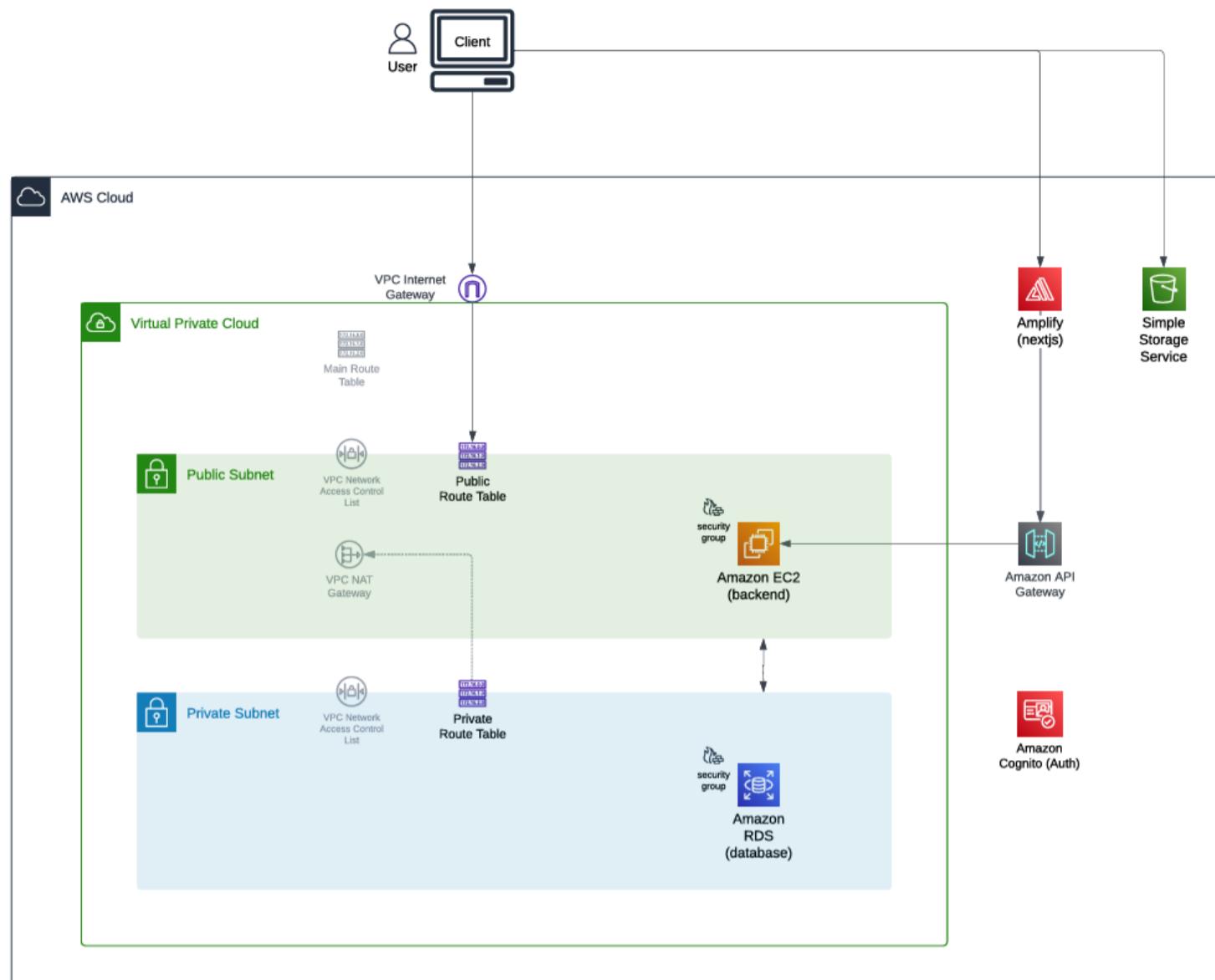
```
git remote add origin https://github.com/your-username/project-management.git
```

```
git branch -M main
```

```
git push -u origin main
```

✓ **Repository successfully stored on GitHub**

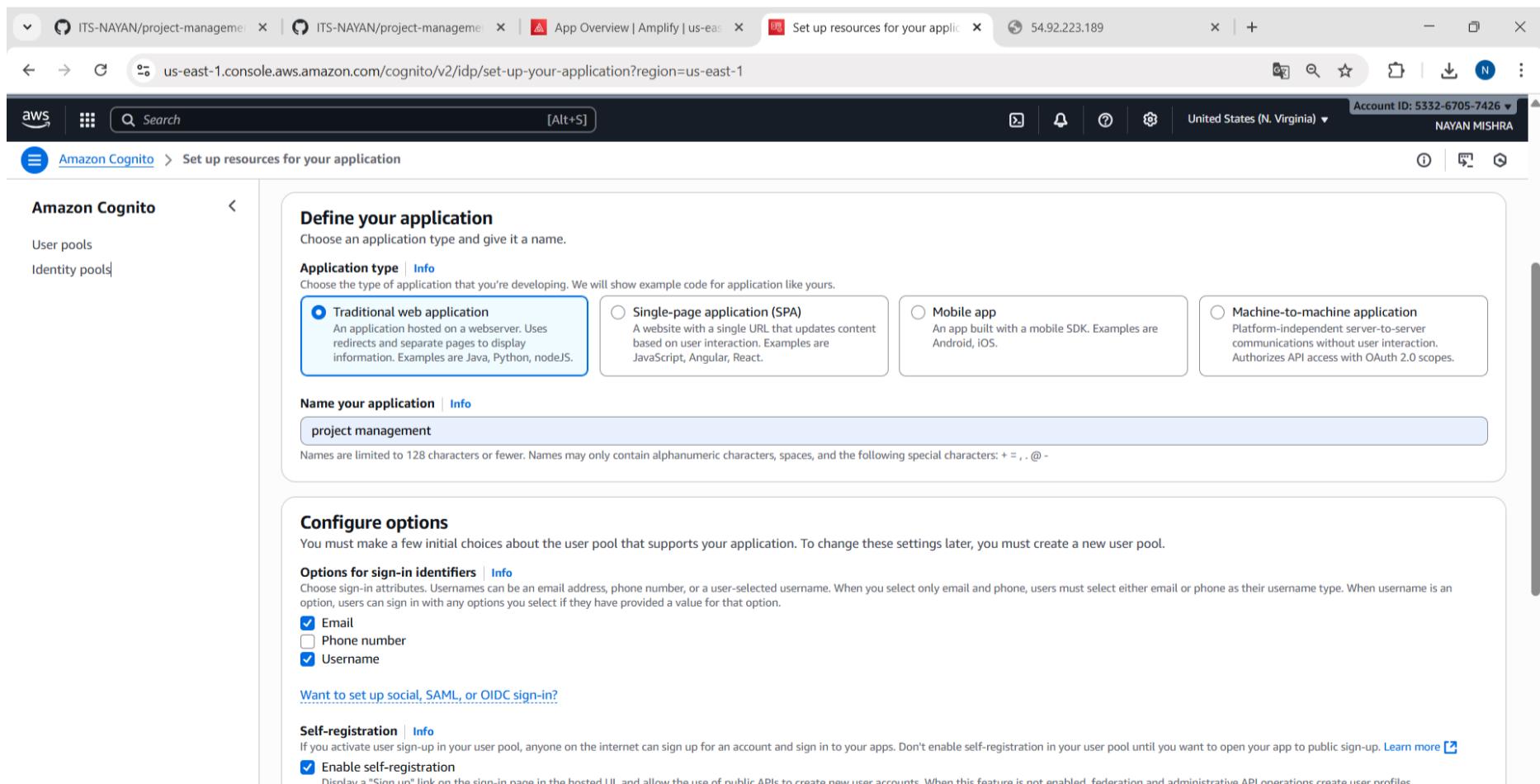
AWS DEPLOYMENT ARCHITECTURE



Step 2 — Configure AWS Cognito Authentication

Create User Pool

Use AWS Console → Cognito → Create User Pool



The screenshot shows the 'Set up resources for your application' page in the AWS Cognito console. The left sidebar shows 'Amazon Cognito' with 'User pools' selected. The main area starts with a 'Define your application' section where the 'Application type' is set to 'Traditional web application'. Below this, a 'Name your application' input field contains 'project management'. The next section is 'Configure options', which includes settings for sign-in identifiers ('Email', 'Phone number', 'Username') and self-registration ('Enable self-registration'). The URL in the browser is us-east-1.console.aws.amazon.com/cognito/v2/idp/set-up-your-application?region=us-east-1.

The screenshot shows the 'Set up resources for your application' page in the AWS Cognito console. Under 'Self-registration', the 'Enable self-registration' checkbox is checked. In the 'Required attributes for sign-up' section, 'email' is selected. Under 'Add a return URL - optional', 'https://' is entered. At the bottom right are 'Cancel' and 'Create user directory' buttons.

✓ Note down the User Pool ID

Example: us-east-1_2Kw2sXuWQ

✓ Create User Pool Client (via CloudShell)

Run the following command in AWS CloudShell:

The screenshot shows the AWS CloudShell interface. A terminal window is open with the command `aws cognito-idp create-user-pool-client` entered. The command includes parameters: `--user-pool-id us-east-1_2Kw2sXuWQ`, `--client-name amplify-client-public`, `--no-generate-secret`, `--explicit-auth-flows ALLOW_USER_SRP_AUTH ALLOW_REFRESH_TOKEN_AUTH ALLOW_ADMIN_USER_PASSWORD_AUTH`, `--query 'UserPoolClient.ClientId'`, and `--output text`. The output of the command is visible at the bottom of the terminal window.

```

~ $ aws configure
AWS Access Key ID [*****E7YA]:
AWS Secret Access Key [*****Xnkb]:
Default region name [eu-east-1]:
Default output format [None]:
~ $ aws cognito-idp create-user-pool-client \
>   --user-pool-id us-east-1_2Kw2sXuWQ \
>   --client-name amplify-client-public \
>   --no-generate-secret \
>   --explicit-auth-flows ALLOW_USER_SRP_AUTH ALLOW_REFRESH_TOKEN_AUTH ALLOW_ADMIN_USER_PASSWORD_AUTH \
>   --query 'UserPoolClient.ClientId' \
>   --output text
7qq0951gdal3ojla0csv110sm7
~ $ 

```

```

aws cognito-idp create-user-pool-client \
--user-pool-id us-east-1_2Kw2sXuWQ \
--client-name amplify-client-public \
--no-generate-secret \
--explicit-auth-flows ALLOW_USER_SRP_AUTH ALLOW_REFRESH_TOKEN_AUTH ALLOW_ADMIN_USER_PASSWORD_AUTH \
--query 'UserPoolClient.ClientId' \
--output text

```

Example output:

7qq0951gdal3ojla0csv110sm7

📌 Important IDs to Store

ID Type	Value
Cognito User Pool ID	us-east-1_2Kw2sXuWQ
Cognito Client ID	7qq0951gdal3ojla0csv110sm7
💡 Keep these safe — required for authentication integration.	

✓ Update Environment Variables in Frontend

```
1 NEXT_PUBLIC_API_BASE_URL=http://localhost:8000
2 NEXT_PUBLIC_COGNITO_USER_POOL_ID="us-east-1_2Kw2sXuWQ"
3 NEXT_PUBLIC_COGNITO_USER_POOL_CLIENT_ID="7qq0951gdal3ojla0csv110sm7"
4

PS C:\Users\nayan\Desktop\project-management> git add .
warning: in the working copy of 'client/.env.local', LF will be replaced by CRLF the next time git touches it
PS C:\Users\nayan\Desktop\project-management> git commit -m "Updated cognito user pool id and client"
[main edc063d] Updated cognito user pool id and client
 1 file changed, 2 insertions(+), 2 deletions(-)
PS C:\Users\nayan\Desktop\project-management> git push
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 428 bytes | 428.00 KiB/s, done.
Total 4 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To https://github.com/ITS-NAYAN/project-management.git
 0443f15..edc063d main -> main
PS C:\Users\nayan\Desktop\project-management>
```

Go to:

📁 client/src/ → Create / Edit → .env.local

Add:

NEXT_PUBLIC_COGNITO_USER_POOL_ID=us-east-1_2Kw2sXuWQ

NEXT_PUBLIC_COGNITO_CLIENT_ID=7qq0951gdal3ojla0csv110sm7

✓ Also push changes in git using commands shown in screenshot. This connects your frontend to AWS Cognito authentication

PHASE 1 — CORE NETWORKING INFRASTRUCTURE

Components: VPC | Subnets | Internet Gateway | Route Tables

Objective:

To build a secure and isolated network environment in AWS where:

- Public-facing resources get controlled internet access
- Private resources remain fully protected and isolated
- High availability and scalability are supported

1 VPC Creation — PM-VPC

Setting Value

Name PM-VPC

CIDR Block 10.0.0.0/16

Purpose & Benefits:

- Creates a **logically isolated** network from the AWS public cloud
- /16 CIDR provides **65,536 private IPs**, allowing future scaling
- Forms the **foundation** for all networking in this project

2 Subnet Design & Configuration

To properly segment resources and secure sensitive workloads:

Subnet Name	CIDR	Type	Availability Zone	Purpose
PM-public-subnet-1	10.0.0.0/24	Public	us-east-2a	EC2 / API resources requiring internet access
PM-private-subnet-1	10.0.1.0/24	Private	us-east-2a	Backend + Database (sensitive components)
PM-private-subnet-2	10.0.2.0/24	Private	us-east-2b	High availability + RDS Multi-AZ support

Why two private subnets?

- Ensures **fault tolerance**
- Required for **RDS Multi-AZ** deployments
- Supports future auto-scaling in multiple zones

3 Internet Gateway — PM-Internet-Gateway

Element Description

Internet Gateway Created and attached to **PM-VPC**

Purpose:

- Provides **internet connectivity** for resources in public subnets
- Enables inbound/outbound web traffic routing

 *Private subnets remain isolated — no IGW route*

Route Tables — Public & Private Segregation

Route Table Name	Associated Subnets	Internet Route	Purpose
PM-public-route-table-1	PM-public-subnet-1	0.0.0.0/0 → Internet Gateway	Allows public resources to access the internet
PM-private-route-table-1	PM-private-subnet-1	 No internet route	Full isolation for sensitive resources
PM-private-route-table-2	PM-private-subnet-2	 No internet route	Supports HA database deployment

Routing Benefits:

- Public subnets handle internet-facing workloads securely
- Private subnets restrict exposure and reduce attack surface

Phase 2 — Backend Deployment

Components: EC2 | Security Groups | Node.js Backend

Objective: Deploy the backend application on an EC2 instance and connect it securely with a PostgreSQL database hosted on AWS.

1 EC2 Instance Deployment — pm-ec2-backend

Configuration Details

Instance Type t2.micro (Free Tier Eligible)

AMI Amazon Linux 2

Network Placement PM-VPC → PM-public-subnet-1

Public IP Enabled (for internet access)

Key Pair Created for secured SSH login

Security Group — pm-ec2-sg

Port Protocol Purpose

22 SSH Remote admin access

80 HTTP Public access to backend APIs

443 HTTPS Secure traffic support

 Only required and safe communication ports are allowed → **enhanced security**

Software & Backend Setup (SSH Deployment Steps)

Once the EC2 instance is live:

 Connect using SSH and configure: And run all commands in **aws-ec2-instructions.md**

 THIS IS FROM EC2 INSTRUCTION.MD Create a pm2 ecosystem configuration file (inside server directory):

```
[root@ip-10-0-0-176 server]# 
[root@ip-10-0-0-176 server]# ls
aws-ec2-instructions.md  dist  ecosystem.config.js  node_modules  package-lock.json  package.json  prisma  src  tsconfig.json
[root@ip-10-0-0-176 server]# nano ecosystem.config.js
[root@ip-10-0-0-176 server]# |
```



The screenshot shows a terminal window titled "root@ip-10-0-0-176:~/project". The command "nano ecosystem.config.js" is being run to edit the file. The content of the file is displayed in the terminal:

```
GNU nano 8.3
module.exports = {
  apps: [
    {
      name: "project-management",
      script: "npm",
      args: "run dev",
      env: {
        NODE_ENV: "development",
      },
    },
  ],
};
```

RDS DATABASE SETUP — PM-RDS

Objective: Deploy a secured, private PostgreSQL database accessible **only** by the backend EC2 instance.

Database Configuration

Setting	Value
Engine	PostgreSQL
DB Identifier	pm-rds
Instance Class	Free-Tier Eligible
Public Access	 Disabled
Storage Autoscaling	 Disabled
Backups	Disabled (cost optimization)
Encryption	Disabled (free-tier compliance)

Master Username & Password

→ Stored securely and **not exposed publicly**

Network Placement

Component	Configuration
VPC	PM-VPC
Subnet Group	Private — includes PM-private-subnet-1 & PM-private-subnet-2
High Availability	 Multi-AZ enabled via subnet group placement

Benefits:

- Private subnets → **No internet exposure**
- High availability → **AZ failure resilience**
- Backend-only access → **Zero unwanted traffic**

Security Group — PM-RDS-SG(**SCREENSHOT AT END**)

Rule Type	Source	Port	Purpose
Inbound	pm-ec2-sg	5432	Allow database access only from backend
Outbound	pm-ec2-sg	allowed	Ensures return traffic to backend

 Strong security posture:

Database invisible to the outside world and other AWS services

Screenshot of the AWS RDS 'Create database' wizard showing the 'Engine options' step.

Choose a database creation method

- Standard create
You set all of the configuration options, including ones for availability, security, backups, and maintenance.
- Easy create
Use recommended best-practice configurations. Some configuration options can be changed after the database is created.

Engine options

Engine type: [Info](#)

<input type="radio"/> Aurora (MySQL Compatible) 	<input type="radio"/> Aurora (PostgreSQL Compatible) 	<input type="radio"/> MySQL
<input checked="" type="radio"/> PostgreSQL 	<input type="radio"/> MariaDB 	<input type="radio"/> Oracle
<input type="radio"/> Microsoft SQL Server 	<input type="radio"/> IBM Db2 	

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Screenshot of the AWS RDS 'Create database' wizard showing the 'Engine options' step, with 'IBM Db2' selected.

Engine version [Info](#)
View the engine versions that support the following database features.

Hide filters

Show only versions that support the Multi-AZ DB cluster [Info](#)
Create a A Multi-AZ DB cluster with one primary DB instance and two readable standby DB instances. Multi-AZ DB clusters provide up to 2x faster transaction commit latency and automatic failover in typically under 35 seconds.

Engine version
PostgreSQL 17.4-R2

Enable RDS Extended Support [Info](#)
Amazon RDS Extended Support is a paid offering [\[?\]](#). By selecting this option, you consent to being charged for this offering if you are running your database major version past the RDS end of standard support date for that version. Check the end of standard support date for your major version in the [RDS for PostgreSQL documentation](#) [\[?\]](#).

Templates
Choose a sample template to meet your use case.

Production
Use defaults for high availability and fast, consistent performance.

Dev/Test
This instance is intended for development use outside of a production environment.

Free tier
Use RDS Free Tier to develop new applications, test existing applications, or gain hands-on experience with Amazon RDS. [Info](#)

Availability and durability
[Deployment options](#) [Info](#)

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

project-management1/service... | ITS-NAYAN/project-manag... | VPC | us-east-1 | Connect to instance | EC2 | Instances | EC2 | us-east-1 | Create database | Aurora | +

us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#launch-dbinstance:

Aurora and RDS > Databases > Create database

Availability and durability

Deployment options [Info](#)

Choose the deployment option that provides the availability and durability needed for your use case. AWS is committed to a certain level of uptime depending on the deployment option you choose. Learn more in the [Amazon RDS service level agreement \(SLA\)](#).

- Multi-AZ DB cluster deployment (3 instances)

Creates a primary DB instance with two readable standbys in separate Availability Zones. This setup provides:

 - 99.95% uptime
 - Redundancy across Availability Zones
 - Increased read capacity
 - Reduced write latency
- Multi-AZ DB instance deployment (2 instances)

Creates a primary DB instance with a non-readable standby instance in a separate Availability Zone. This setup provides:

 - 99.95% uptime
 - Redundancy across Availability Zones
- Single-AZ DB instance deployment (1 instance)

Creates a single DB instance without standby instances. This setup provides:

 - 99.5% uptime
 - No data redundancy

project-management1/service... | ITS-NAYAN/project-manag... | VPC | us-east-1 | Connect to instance | EC2 | Instances | EC2 | us-east-1 | Create database | Aurora | +

us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#launch-dbinstance:

Aurora and RDS > Databases > Create database

Settings

DB instance identifier [Info](#)

Type a name for your DB instance. The name must be unique across all DB instances owned by your AWS account in the current AWS Region.

pm-rds

The DB instance identifier is case-insensitive, but is stored as all lowercase (as in "mydbinstance"). Constraints: 1 to 63 alphanumeric characters or hyphens. First character must be a letter. Can't contain two consecutive hyphens. Can't end with a hyphen.

Credentials Settings

Master username [Info](#)

Type a login ID for the master user of your DB instance.

postgres

1 to 16 alphanumeric characters. The first character must be a letter.

Credentials management

You can use AWS Secrets Manager or manage your master user credentials.

Managed in AWS Secrets Manager - *most secure*
RDS generates a password for you and manages it throughout its lifecycle using AWS Secrets Manager.

Self managed
Create your own password or have RDS create a password that you manage.

Auto generate password
Amazon RDS can generate a password for you, or you can specify your own password.

Master password [Info](#)

.....

Password strength Very weak

Minimum constraints: At least 8 printable ASCII characters. Can't contain any of the following symbols: / \ ^ @

Confirm master password [Info](#)

.....

project-management1/service... | ITS-NAYAN/project-manag... | VPC | us-east-1 | Connect to instance | EC2 | Instances | EC2 | us-east-1 | Create database | Aurora | ...

us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#launch-dbinstance:

Aurora and RDS > Databases > Create database

Instance configuration

The DB instance configuration options below are limited to those supported by the engine that you selected above.

DB instance class [Info](#)

Hide filters

Include previous generation classes

Standard classes (includes m classes)

Memory optimized classes (includes r and x classes)

Burstable classes (includes t classes)

db.t3.micro

2 vCPUs 1 GiB RAM EBS Bandwidth: Up to 2,085 Mbps Network: Up to 5 Gbps

Storage

Storage type [Info](#)

Provisioned IOPS SSD (io2) storage volumes are now available.

General Purpose SSD (gp2)
Baseline performance determined by volume size

Allocated storage [Info](#)

20 GiB

Allocated storage value must be 20 GiB to 6,144 GiB

Additional storage configuration

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

project-management1/service... | ITS-NAYAN/project-manag... | VPC | us-east-1 | Connect to instance | EC2 | Instances | EC2 | us-east-1 | Create database | Aurora | ...

us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#launch-dbinstance:

Aurora and RDS > Databases > Create database

Connectivity [Info](#)

Compute resource

Choose whether to set up a connection to a compute resource for this database. Setting up a connection will automatically change connectivity settings so that the compute resource can connect to this database.

Don't connect to an EC2 compute resource
Don't set up a connection to a compute resource for this database. You can manually set up a connection to a compute resource later.

Connect to an EC2 compute resource
Set up a connection to an EC2 compute resource for this database.

Virtual private cloud (VPC) [Info](#)

Choose the VPC. The VPC defines the virtual networking environment for this DB instance.

PM-VPC (vpc-0225ca86f69c0e54b)
3 Subnets, 2 Availability Zones

Only VPCs with a corresponding DB subnet group are listed.

After a database is created, you can't change its VPC.

DB subnet group [Info](#)

Choose the DB subnet group. The DB subnet group defines which subnets and IP ranges the DB instance can use in the VPC that you selected.

Create new DB Subnet Group

Public access [Info](#)

Yes
RDS assigns a public IP address to the database. Amazon EC2 instances and other resources outside of the VPC can connect to your database. Resources inside the VPC can also connect to the database. Choose one or more VPC security groups that specify which resources can connect to the database.

No
RDS doesn't assign a public IP address to the database. Only Amazon EC2 instances and other resources inside the VPC can connect to your database. Choose one or more VPC security groups that specify which resources can connect to the database.

VPC security group (firewall) [Info](#)

Choose one or more VPC security groups to allow access to your database. Make sure that the security group rules allow the appropriate incoming traffic.

Choose existing
Choose existing VPC security groups

Create new
Create new VPC security group

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

project-management1/service... | ITS-NAYAN/project-manag... | VPC | us-east-1 | Connect to instance | EC2 | Instances | EC2 | us-east-1 | Create database | Aurora | ...

us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#launch-dbinstance:

Aurora and RDS > Databases > Create database

VPC security group (firewall) [Info](#)
Choose one or more VPC security groups to allow access to your database. Make sure that the security group rules allow the appropriate incoming traffic.

Choose existing
Choose existing VPC security groups

Create new
Create new VPC security group

New VPC security group name
MY-RDS-SG

Availability Zone [Info](#)
us-east-1

RDS Proxy [Info](#)
RDS Proxy is a fully managed, highly available database proxy that improves application scalability, resiliency, and security.

Create an RDS Proxy [Info](#)
RDS automatically creates an IAM role and a Secrets Manager secret for the proxy. RDS Proxy has additional costs. For more information, see [Amazon RDS Proxy pricing](#).

Certificate authority - optional [Info](#)
Using a server certificate provides an extra layer of security by validating that the connection is being made to an Amazon database. It does so by checking the server certificate that is automatically installed on all databases that you provision.

rds-ca-rsa2048-g1 (default)
Expiry: May 26, 2061

If you don't select a certificate authority, RDS chooses one for you.

▶ Additional configuration

Tags - optional

A tag consists of a case-sensitive key-value pair.
No tags associated with the resource.

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

project-management1/service... | ITS-NAYAN/project-manag... | VPC | us-east-1 | Connect to instance | EC2 | Instances | EC2 | us-east-1 | Create database | Aurora | ...

us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#launch-dbinstance:

Aurora and RDS > Databases > Create database

Tags - optional

A tag consists of a case-sensitive key-value pair.
No tags associated with the resource.

Add new tag
You can add up to 50 more tags.

Database authentication

Database authentication options [Info](#)

Password authentication
Authenticates using database passwords.

Password and IAM database authentication
Authenticates using the database password and user credentials through AWS IAM users and roles.

Password and Kerberos authentication
Choose a directory in which you want to allow authorized users to authenticate with this DB instance using Kerberos Authentication.

Monitoring [Info](#)
Choose monitoring tools for this database. Database Insights provides a combined view of Performance Insights and Enhanced Monitoring for your fleet of databases. [Database Insights](#) pricing is separate from RDS monthly estimates. See [Amazon CloudWatch pricing](#).

Database Insights - Advanced

- Retains 15 months of performance history
- Fleet-level monitoring
- Integration with CloudWatch Application Signals

Database Insights - Standard

- Retains 7 days of performance history, with the option to pay for the retention of up to 24 months of performance history

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

project-management1/service... | ITS-NAYAN/project-manag... | VPC | us-east-1 | Connect to instance | EC2 | Instances | EC2 | us-east-1 | Create database | Aurora | +

us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#launch-dbinstance:

Aurora and RDS > Databases > Create database

Performance Insights

Enable Performance Insights
With Performance Insights dashboard, you can visualize the database load on your Amazon RDS DB instance load and filter the load by waits, SQL statements, hosts, or users.

Additional monitoring settings

Enhanced Monitoring, CloudWatch Logs and DevOps Guru

Enhanced Monitoring

Enable Enhanced monitoring
Enabling Enhanced Monitoring metrics are useful when you want to see how different processes or threads use the CPU.

Log exports

Select the log types to publish to Amazon CloudWatch Logs

iam-db-auth-error log
 PostgreSQL log
 Upgrade log

IAM role

The following service-linked role is used for publishing logs to CloudWatch Logs.

RDS service-linked role

Devops Guru

Turn on DevOps Guru [Info](#)
DevOps Guru for RDS automatically detects performance anomalies for DB instances and provides recommendations.

Additional configuration

Database options, encryption turned off, backup turned off, backtrack turned off, maintenance, CloudWatch Logs, delete protection turned off.

Database options

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

project-management1/service... | ITS-NAYAN/project-manag... | VPC | us-east-1 | Connect to instance | EC2 | Instances | EC2 | us-east-1 | Create database | Aurora | +

us-east-1.console.aws.amazon.com/rds/home?region=us-east-1#launch-dbinstance:

Aurora and RDS > Databases > Create database

Additional configuration

Database options, encryption turned off, backup turned off, backtrack turned off, maintenance, CloudWatch Logs, delete protection turned off.

Database options

Initial database name [Info](#)

If you do not specify a database name, Amazon RDS does not create a database.

DB parameter group [Info](#)

default.postgres17

Option group [Info](#)

default:postgres-17

Backup

Enable automated backup
Creates a point-in-time snapshot of your database

Enable encryption
Choose to encrypt the given instance. Master key IDs and aliases appear in the list after they have been created using the AWS Key Management Service console. [Info](#)

Maintenance

Auto minor version upgrade [Info](#)

Enable auto minor version upgrade
Enabling auto minor version upgrade will automatically upgrade your database minor version. For limitations and more details, see Automatically upgrading the minor engine version documentation [\[?\]](#).

Maintenance window [Info](#)

Select the period you want pending modifications or maintenance applied to the database by Amazon RDS.

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

The screenshot shows the 'Create database' wizard in the AWS RDS console. Under 'Auto minor version upgrade', 'Enable auto minor version upgrade' is checked. In the 'Maintenance window' section, 'No preference' is selected. Under 'Estimated monthly costs', it states: 'The Amazon RDS Free Tier is available to you for 12 months. Each calendar month, the free tier will allow you to use the Amazon RDS resources listed below for free:' followed by a bulleted list: '750 hrs of Amazon RDS in a Single-AZ db.t2.micro, db.t3.micro or db.t4g.micro Instance.', '20 GB of General Purpose Storage (SSD).', and '20 GB for automated backup storage and any user-initiated DB Snapshots.' A note at the bottom says: 'Learn more about AWS Free Tier.'

🔧 Backend Integration (EC2)

After RDS creation:

✓ 1 Update .env configuration on EC2

Added RDS connection parameters including: you add your own

- RDS Endpoint: pm-rds.cbk00q80wnp4.us-east-1.rds.amazonaws.com
- Database Name
- Master Username
- Master Password: 987654321

```
root@ip-10-0-0-176:~/project ~ + ~
[root@ip-10-0-0-176 server]# nano .env
```

```
root@ip-10-0-0-176:~/project ~ + ~ .env Modified
GNU nano 8.3
PORT=80
DATABASE_URL="postgresql://postgres:987654321@pm-rds.cbk00q80wnp4.us-east-1.rds.amazonaws.com:5432/projectmanagement?schema=public"
```

Paste this URL according to configuration: `DATABASE_URL="postgresql://postgres:987654321@pm-rds.cbk00q80wnp4.us-east-1.rds.amazonaws.com:5432/projectmanagement?schema=public"`

project-management1/sen | ITS-NAYAN/project-manag | VPC | us-east-1 | Connect to instance | EC2 | Instances | EC2 | us-east-1 | ModifyOutboundSecurityG | + | - | X

us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#ModifyOutboundSecurityGroupRules:securityGroupId=sg-01945d3629624caa4

aws Search [Alt+S] Account ID: 5332-6705-7426 United States (N. Virginia) NAYAN MISHRA

EC2 > Security Groups > sg-01945d3629624caa4 - PM-EC2-SG > Edit outbound rules

Edit outbound rules Info

Outbound rules control the outgoing traffic that's allowed to leave the instance.

Outbound rules Info

Security group rule ID	Type	Protocol	Port range	Destination	Description - optional
sgr-00fedf3375c4b009d	All traffic	All	All	Custom 0.0.0.0/0	
-	PostgreSQL	TCP	5432	Custom sg-0e327b23154fb980f	

Add rule

⚠ Rules with destination of 0.0.0.0/0 or ::/0 allow your instances to send traffic to any IPv4 or IPv6 address. We recommend setting security group rules to be more restrictive and to only allow traffic to specific known IP addresses. X

Cancel Preview changes Save rules

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

project-management1/sen | ITS-NAYAN/project-manag | VPC | us-east-1 | Connect to instance | EC2 | Instances | EC2 | us-east-1 | ModifyInboundSecurityGroupRules:securityGroupId=sg-0e327b23154fb980f

us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#ModifyInboundSecurityGroupRules:securityGroupId=sg-0e327b23154fb980f

aws Search [Alt+S] Account ID: 5332-6705-7426 United States (N. Virginia) NAYAN MISHRA

EC2 > Security Groups > sg-0e327b23154fb980f - MY-RDS-SG > Edit inbound rules

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-0835f2caa422b8929	PostgreSQL	TCP	5432	Custom 171.61.84.0/32	
-	PostgreSQL	TCP	5432	Custom sg-01945d3629624caa4	

Add rule

Cancel Preview changes Save rules

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

2 Run Prisma Commands: In EC2 run these commands

npx prisma generate

npx prisma migrate dev

npm run seed

Result:

- Database schema created
 - Initial data inserted successfully
-

 Final Connectivity Test

Backend API was accessed via EC2 public IP →

Data retrieved from RDS confirming successful:

Check	Status
EC2 → RDS Connectivity	<input checked="" type="checkbox"/>
Prisma Migration	<input checked="" type="checkbox"/>
Application Data Response	<input checked="" type="checkbox"/>

RDS Deployment Completed!

You now have:

- ◆ Secure PostgreSQL running inside private AWS network
 - ◆ HA-ready architecture using multi-AZ design
 - ◆ Fully connected backend storing real data
-

 Backend Public Access Test (EC2)

Once the backend deployment is complete:

Copy the **Public IPv4 Address** of EC2

→ Paste it into any browser

→ Expected Result:



If the Backend is NOT Loading (Troubleshooting)

Sometimes the backend may fail to start on **Port 80** due to a conflicting service already using it.

Step-by-Step Fix

1 Check Which Process is Using Port 80

Run either command:

```
sudo lsof -i :80
```

or

```
sudo netstat -tulnp | grep :80
```

Sample response:

```
nginx 1234 root 6u IPv4 123456 0t0 TCP *:80 (LISTEN)
```

→ Another web server such as **nginx**, **Apache**, or a previous **Node.js process** is using port 80

2 Resolve the Port Conflict

You have **two safe options**:

— Free up Port 80 (Recommended)

If the service running on 80 is not needed: **sudo kill -9 <PID>**

Replace <PID> with the ID identified above.

Then restart the backend:

```
pm2 restart project-management
```

Check logs to confirm the app is running:

```
pm2 logs project-management
```

Expected logs: Server started on port 80 

EXPECTED SCREENSHOT



The screenshot shows a browser window with the URL `98.94.58.142/projects`. The page displays a JSON array of project objects. Each project has an ID, name, description, start date, and end date. The projects listed are Apollo, Beacon, Catalyst, Delta, Echo, Foxtrot, and Golf.

```
[{"id": 1, "name": "Apollo", "description": "A space exploration project.", "startDate": "2023-01-01T00:00:00.000Z", "endDate": "2023-12-31T00:00:00.000Z"}, {"id": 2, "name": "Beacon", "description": "Developing advanced navigation systems.", "startDate": "2023-02-01T00:00:00.000Z", "endDate": "2023-10-15T00:00:00.000Z"}, {"id": 3, "name": "Catalyst", "description": "A project to boost renewable energy use.", "startDate": "2023-03-05T00:00:00.000Z", "endDate": "2024-03-05T00:00:00.000Z"}, {"id": 4, "name": "Delta", "description": "Delta project for new software development techniques.", "startDate": "2023-01-20T00:00:00.000Z", "endDate": "2023-09-20T00:00:00.000Z"}, {"id": 5, "name": "Echo", "description": "Echo project focused on AI advancements.", "startDate": "2023-04-15T00:00:00.000Z", "endDate": "2023-11-30T00:00:00.000Z"}, {"id": 6, "name": "Foxtrot", "description": "Exploring cutting-edge biotechnology.", "startDate": "2023-02-25T00:00:00.000Z", "endDate": "2023-08-25T00:00:00.000Z"}, {"id": 7, "name": "Golf"}]
```

REFERENCE SCREENSHOT

The terminal session shows the following commands and output:

- `sudo lsof -i :80` - Lists listening ports, showing a connection for MainThread on port 80.
- `sudo kill -9 28452` - Kills the process with ID 28452.
- `pm2 restart project-management` - Restarts the PM2 process named `project-management`.
- `pm2 list` - Displays the current PM2 processes. The table shows one process named `project-management` with ID 0, running in fork mode, using 0% CPU, and 25.2mb memory.

id	name	namespace	version	mode	pid	uptime	⌚	status	cpu	mem	user	watching
0	project-management	default	N/A	fork	29873	0s	0	online	0%	25.2mb	root	disabled

✓ Phase 3: Frontend Deployment & Secure API Access (Amplify + API Gateway)

Objective: Deploy the Next.js frontend application through AWS Amplify and ensure secure, HTTPS-encrypted communication with the backend API, resolving browser mixed-content issues.

1 Frontend Deployment using AWS Amplify Hosting

- Application Setup:**
A new Amplify Hosting application was provisioned to manage frontend deployments.
- GitHub Integration:**
Amplify was configured to pull source code from the GitHub monorepo containing both frontend and backend components.
- Monorepo Build Configuration:**
The build root was explicitly set to the client/ directory to ensure only the Next.js frontend is built and deployed.
- CI/CD Automation:**
Amplify established an automated deployment pipeline where every Git commit triggers a new build and release process.
- Initial Deployment Success:**
The frontend was successfully launched and made publicly accessible over **HTTPS** with global CDN support.
- Identified Problem — Mixed Content Error:**
Since the frontend was served over **HTTPS**, but the backend API on EC2 was only accessible via **HTTP**, browsers blocked API requests due to **mixed-content** security enforcement.
This resulted in failed data fetching and UI rendering disruptions.

Screenshot of the AWS Amplify 'Create new app' wizard Step 1: Choose source code provider.

The page shows a sidebar with steps 1-4: Choose source code provider (selected), Add repository and branch, App settings, Review.

Start building with Amplify
Amplify provides a fully-managed web hosting experience and a backend building service to build fullstack apps. If you need a starter project, please visit the [docs](#).

Deploy your app GitHub, BitBucket, CodeCommit, GitLab

To deploy an app from a Git provider, select one of the options below:

GitHub BitBucket CodeCommit GitLab

Amplify requires read-only access to your repository.

To manually deploy an app from Amazon S3 or a Zip file, select "Deploy without Git"

Deploy without Git

▶ Start with a template N V A W

Cancel Previous Next

Screenshot of the AWS Amplify 'Create new app' wizard Step 2: Add repository and branch.

The page shows a sidebar with steps 1-4: Choose source code provider (selected), Add repository and branch (selected), App settings, Review.

Add repository and branch

Repository dropdown: ITS-NAYAN/project-management

Info message: If you don't see your repository in the dropdown above, ensure the Amplify GitHub App has permissions to the repository. If your repository still doesn't appear, push a commit and click the refresh button. [Update GitHub permissions](#)

Branch dropdown: main

Monorepo root directory: client

My app is a monorepo checkbox: checked

Cancel Previous Next

Screenshot of the AWS Amplify 'Create new app' wizard Step 3 - App settings.

App settings

App name: project-management

Build settings:

- Auto-detected frameworks: Next.js
- Frontend build command: npm run build
- Build output directory: .next

Monorepo Environments:

- My monorepo uses Amplify Gen2 Backend

Review: 4 steps completed

Screenshot of the AWS Amplify 'Create new app' wizard Step 3 - App settings.

Advanced settings

Build instance type: Standard (8 GiB Memory | 4 vCPUs | 128 GB Disk Space)

Build image: Amazon Linux 2023 (default)

Environment variables:

Key	Value
AMPLIFY_DIFF_DEPLOY	false
AMPLIFY_MONOREPO_APP_ROOT	client
NEXT_PUBLIC_API_BASE_URL	http://98.94.58.142

Review: 4 steps completed

Screenshot of the AWS Amplify 'Create new app' Step 3 - App Settings page.

The sidebar shows completed steps: Choose source code provider, Add repository and branch, and App settings (selected). The Review step is shown as step 4.

App settings

Key	Value
AMPLIFY_DIFF_DEPLOY	false
AMPLIFY_MONOREPO_APP_ROOT	client
NEXT_PUBLIC_API_BASE_URL	http://98.94.58.142

Keep cookies in cache key
Enabled
⚠️ Changing this setting can impact your app's performance. [Learn more](#)

Live package updates
Override the default installed versions of packages or tools during the build.
+ Add new

Buttons: Cancel, Previous, Next

Screenshot of the AWS Amplify 'Create new app' Step 4 - Review page.

The sidebar shows completed steps: Choose source code provider, Add repository and branch, App settings (selected), and Review (selected).

App settings

App name: project-management	Frontend build command: npm run build
Framework: Next.js	

Advanced settings

Build instance type: Standard 8 GiB Memory 4 vCPUs 128 GB Disk Space	Build image: Using default image
Keep cookies in cache key: Enabled	Live package updates
Environment variables: AMPLIFY_DIFF_DEPLOY: false AMPLIFY_MONOREPO_APP_ROOT: client NEXT_PUBLIC_API_BASE_URL: http://98.94.58.142	

Buttons: Cancel, Previous, Save and deploy

Screenshot of the AWS Amplify console showing the project-management app overview. The sidebar includes sections for Overview, Hosting, Monitoring, and App settings. The main area displays steps to get to production (Add a custom domain, Enable firewall protections, Connect new branches), a branch list (main, Deployed), and deployment details (Domain: https://main.d3ognm63o73b0p.amplifyapp.com, Last deployment: 0 minutes ago, Last commit: Auto-build / project-management:main). Buttons for Manage sandboxes and Visit deployed URL are present.

Screenshot of a browser window showing the sign-in page for the Amplify app at main.d3ognm63o73b0p.amplifyapp.com. The page features fields for Username (Nayan), Email (study458@gmail.com), Password (Nayan@123), and Confirm Password (Nayan@123). A "Create Account" button is at the bottom right. A "Create Next App" link is visible in the top right corner of the browser header.

The screenshot shows a browser window with multiple tabs open, including VPC Console, Instances | EC2, App Overview, and Create Next App. The main content area displays an error message: "Error fetching data". On the left, there is a sidebar titled "EDLIST" with a "EDROH TEAM" section and a "Private" folder. Below this are links for Home, Timeline, Search, Settings, Users, Teams, Projects, Priority, Urgent (High, Medium, Low), and Backlog.

Identified Problem — Mixed Content Error:

Since the frontend was served over **HTTPS**, but the backend API on EC2 was only accessible via **HTTP**, browsers blocked API requests due to **mixed-content** security enforcement.

This resulted in failed data fetching and UI rendering disruptions.

Also note user created in Cognito

The screenshot shows the AWS Cognito User Pools console. The left sidebar includes sections for Current user pool (User pool - w1luxuq), Applications (App clients), User management (Users, Groups), Authentication (Authentication methods, Sign-in, Sign-up, Social and external providers, Extensions), Security (AWS WAF, Threat protection, Log streaming), and Branding (CloudShell, Feedback). The main content area shows the "Users" section with one user listed: "nayan" (studyy458@gmail.com). The user is confirmed and enabled. Below this is the "Import users" section, which shows no user import jobs found.

2 Solution: Securing Backend Access Using API Gateway

To eliminate mixed content issues and enable encrypted API communication, the EC2 backend was routed through Amazon API Gateway.

- **Purpose & Benefit:**

API Gateway acts as a secure HTTPS entry point, providing **SSL/TLS certificate management**, **API routing**, and **network security** without modifying the EC2 instance.

- **API Gateway Type:**

A **REST API** was created for full flexibility and ease of integration with HTTP backends.

- **Proxy Setup {{proxy+} Resource}:**

A catch-all proxy resource was created to forward all request paths dynamically to the EC2 backend — avoiding manual endpoint-by-endpoint setup.

- **CORS Enablement:**

Cross-Origin Resource Sharing was enabled to allow browser-based clients to access the API.

- **HTTP Proxy Integration:**

The proxy resource was configured to forward traffic directly to the backend EC2 instance's public HTTP endpoint:

`http://<EC2-Public-IP>/{{proxy}}`

- **Deployment to prod Stage:**

The API configuration was deployed to a production stage, automatically generating a **public HTTPS Invoke URL**.

- **Frontend Environment Update:**

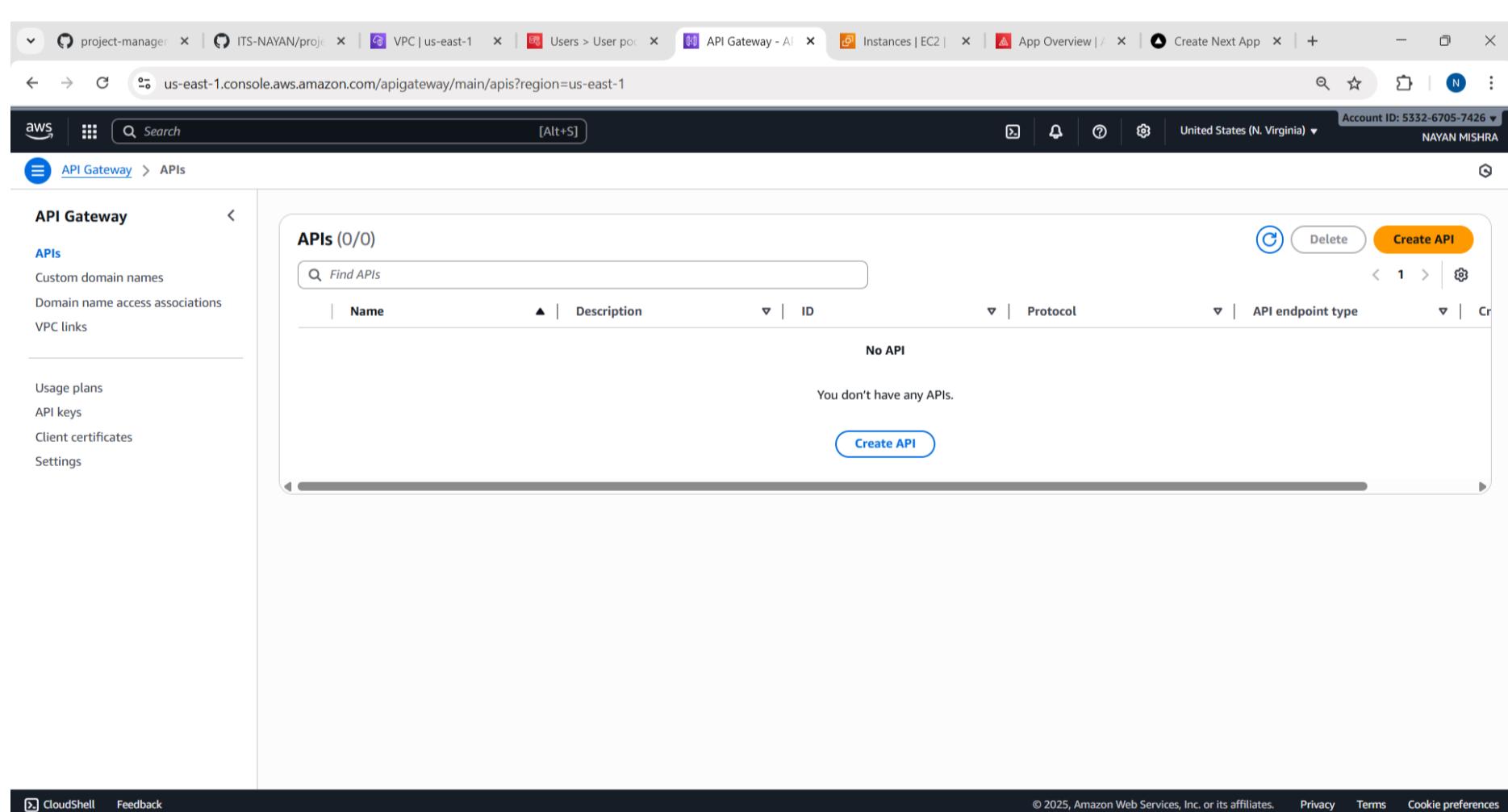
In Amplify, the `NEXT_PUBLIC_API_BASE_URL` variable was updated to use the new **HTTPS endpoint**.

- **Redeployment:**

Amplify was redeployed so the updated secure API endpoint was applied to the live application.

- **Final Outcome — Mixed Content Resolved:**

The frontend successfully communicated with the backend over **fully secure HTTPS**, eliminating security blocking and restoring full application functionality.



The screenshot shows the AWS API Gateway console with the URL `us-east-1.console.aws.amazon.com/apigateway/main/precreate?region=us-east-1`. The page displays two main sections: 'REST API' and 'REST API Private'. Both sections provide a brief description, supported services (Lambda, HTTP, AWS Services), and 'Import' and 'Build' buttons. The 'REST API' section also includes a 'Works with the following:' note.

REST API

Develop a REST API where you gain complete control over the request and response along with API management capabilities.

Works with the following:
Lambda, HTTP, AWS Services

REST API Private

Create a REST API that is only accessible from within a VPC.

Works with the following:
Lambda, HTTP, AWS Services

Import **Build**

The screenshot shows the 'Create REST API' configuration page. The 'API details' section is active, showing options for 'New API' (selected), 'Clone existing API', 'Import API', and 'Example API'. The 'API name' field is set to 'pm_api-gateway'. The 'Description - optional' field is empty. The 'API endpoint type' section indicates 'Regional' is selected. Under 'IP address type', 'IPv4' is selected. At the bottom right are 'Cancel' and 'Create API' buttons.

Create REST API Info

API details

New API
Create a new REST API.

Clone existing API
Create a copy of an API in this AWS account.

Import API
Import an API from an OpenAPI definition.

Example API
Learn about API Gateway with an example API.

API name
pm_api-gateway

Description - optional

API endpoint type
Regional APIs are deployed in the current AWS Region. Edge-optimized APIs route requests to the nearest CloudFront Point of Presence. Private APIs are only accessible from VPCs.
Regional

IP address type Info
Select the type of IP addresses that can invoke the default endpoint for your API.

IPv4
Supports only edge-optimized and Regional API endpoint types.

Dualstack
Supports all API endpoint types.

Create API

Screenshot of the AWS API Gateway console showing a successful REST API creation.

The URL in the browser is <https://us-east-1.console.aws.amazon.com/apigateway/main/apis/mrabaz803c/resources?api=mrabaz803c&experience=rest®ion=us-east-1>.

The main page displays a green success message: "Successfully created REST API 'pm_api-gateway (mrabaz803c)'".

The left sidebar shows the API structure:

- APIs
- Custom domain names
- Domain name access associations
- VPC links
- API: pm_api-gateway**
 - Resources
 - Stages
 - Authorizers
 - Gateway responses
 - Models
 - Resource policy
 - Documentation
 - Dashboard
 - API settings
- Usage plans
- API keys
- Client certificates
- Settings

The "Resources" section shows a single resource path: "/".

Resource details: Path /, Resource ID 7c34cl655g.

Methods (0): No methods defined.

Buttons: API actions, Deploy API, Update documentation, Enable CORS, Delete, Create method.

Screenshot of the AWS API Gateway console showing the creation of a new proxy resource.

The URL in the browser is <https://us-east-1.console.aws.amazon.com/apigateway/main/apis/mrabaz803c/resources/7c34cl655g/create-resource?api=mrabaz803c&experience=rest®ion=us-east-1>.

The page title is "Create resource".

Resource details:

- Proxy resource** Info
Proxy resources handle requests to all sub-resources. To create a proxy resource use a path parameter that ends with a plus sign, for example {proxy+}.
- Resource path**: /
- Resource name**: {proxy+}

CORS (Cross Origin Resource Sharing) Info
Create an OPTIONS method that allows all origins, all methods, and several common headers.

Buttons: Cancel, Create resource.

Screenshot of the AWS API Gateway console showing the creation of a new proxy resource.

The URL in the browser is <https://us-east-1.console.aws.amazon.com/apigateway/main/apis/mrabaz803c/resources?api=mrabaz803c&experience=rest®ion=us-east-1>.

The page shows a green success message: "Successfully created resource '/{proxy+}'".

Resources section details:

- Path: /{proxy+}
- Method type: ANY, OPTIONS
- Integration type: Not setup
- Authorization: None
- API key: Not required

Methods (2) table:

Method type	Integration type	Authorization	API key
ANY	Not setup	None	Not required
OPTIONS	Mock	None	Not required

Actions: API actions, Deploy API.

Left sidebar (API: pm_api-gateway):

- Resources
- Stages
- Authorizers
- Gateway responses
- Models
- Resource policy
- Documentation
- Dashboard
- API settings

Bottom navigation:

- CloudShell
- Feedback
- © 2025, Amazon Web Services, Inc. or its affiliates.
- Privacy
- Terms
- Cookie preferences

Screenshot of the AWS API Gateway console showing the configuration of an integration request for the ANY method of the proxy resource.

The URL in the browser is <https://us-east-1.console.aws.amazon.com/apigateway/main/apis/mrabaz803c/resources/3qf2qa/methods/ANY/edit-integration-request?api=mrabaz803c&experience=rest®ion=us-east-1>.

Method details

Integration type (selected): HTTP

- Lambda function
- AWS service
- VPC link
- Mock

HTTP proxy integration (selected): Send the request to your HTTP endpoint without customizing the integration request or integration response.

HTTP method: ANY

Endpoint URL: http://98.94.58.142/{proxy}

Content handling: Passthrough

Actions: Edit integration request, Save, Cancel.

Bottom navigation:

- CloudShell
- Feedback
- © 2025, Amazon Web Services, Inc. or its affiliates.
- Privacy
- Terms
- Cookie preferences

Screenshot of the AWS API Gateway 'Edit integration request' page.

Endpoint URL: http://98.94.58.142/{proxy}

Content handling: Passthrough

Integration timeout: 29000 ms

Request body passthrough: When there are no templates defined (recommended)

URL path parameters:

URL query string parameters:

URL request headers parameters:

Buttons: Cancel, Save

Screenshot of the AWS API Gateway 'Deploy API' dialog.

Stage: *New stage*

Stage name: prod

Deployment description: (empty)

Buttons: Cancel, Deploy

Background: Shows the API resource configuration with an integration request mapping to an HTTP proxy integration.

ITS-NAYAN/project | VPC | us-east-1 | Users > User pool | API Gateway - Stages | Instances | EC2 | App Overview | Create Next App | mrabaz803c.execute-api.us-east-1.amazonaws.com | +

us-east-1.console.aws.amazon.com/apigateway/main/apis/mrabaz803c/stages?api=mrabaz803c&experience=rest®ion=us-east-1

aws Search [Alt+S] Account ID: 5332-6705-7426 United States (N. Virginia) NAYAN MISHRA

API Gateway > APIs > pm_api-gateway (mrbaz803c) > Stages

Successfully created deployment for pm_api-gateway. This deployment is active for prod.

Stages

prod

Stage details Info

Stage name: prod

Rate Info: 10000

Cache cluster Info: Inactive

Burst Info: 5000

Default method-level caching: Inactive

Web ACL: -

Client certificate: -

Copied

Copy URL: https://mrbaz803c.execute-api.us-east-1.amazonaws.com/prod

Active deployment: 6fnb7s on October 28, 2025, 15:23 (UTC+05:30)

Logs and tracing Info

CloudWatch logs: Inactive

Detailed metrics: Inactive

X-Ray tracing: Inactive

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

ITS-NAYAN/project | VPC | us-east-1 | Users > User pool | API Gateway - Stages | Instances | EC2 | App Overview | Create Next App | mrabaz803c.execute-api.us-east-1.amazonaws.com | +

mrbaz803c.execute-api.us-east-1.amazonaws.com/prod/projects

Pretty-print

```
[{"id": 1, "name": "Apollo", "description": "A space exploration project.", "startDate": "2023-01-01T00:00:00.000Z", "endDate": "2023-12-31T00:00:00.000Z"}, {"id": 2, "name": "Beacon", "description": "Developing advanced navigation systems.", "startDate": "2023-02-01T00:00:00.000Z", "endDate": "2023-10-15T00:00:00.000Z"}, {"id": 3, "name": "Catalyst", "description": "A project to boost renewable energy use.", "startDate": "2023-03-05T00:00:00.000Z", "endDate": "2024-03-05T00:00:00.000Z"}, {"id": 4, "name": "Delta", "description": "Delta project for new software development techniques.", "startDate": "2023-01-20T00:00:00.000Z", "endDate": "2023-09-20T00:00:00.000Z"}, {"id": 5, "name": "Echo", "description": "Echo project focused on AI advancements.", "startDate": "2023-04-15T00:00:00.000Z", "endDate": "2023-11-30T00:00:00.000Z"}, {"id": 6, "name": "Foxtrot", "description": "Exploring cutting-edge biotechnology.", "startDate": "2023-02-25T00:00:00.000Z", "endDate": "2023-08-25T00:00:00.000Z"}, {"id": 7, "name": "Golf"}]
```

ITS Confusing first try with URL only till amazonaws.com, if not then amazonaws.com/prod.

The screenshot shows the AWS Amplify console interface for managing environment variables. The top navigation bar includes tabs for VPC, API Gateway, Instances, Environment Variables, and Create Next App. The main content area is titled "Environment Variables" and displays three defined variables:

Variable	Value	Branch	Actions
AMPLIFY_DIFF_DEPLOY	false	All branches	Actions
AMPLIFY_MONOREPO_APP_ROOT	client	All branches	Actions
NEXT_PUBLIC_API_BASE_URL	https://mrabaz803c.execute-api.us-east-1.amazonaws.com	All branches	Actions

A "Add new" button is available for creating additional variables. The left sidebar lists other hosting-related settings like Access control, Build notifications, and Firewall. The bottom right features "Save" and "Cancel" buttons.

The screenshot shows the AWS Amplify console interface for viewing deployment history. The top navigation bar includes tabs for VPC, API Gateway, Instances, Environment Variables, Deployments, Create Next App, and CloudShell. The main content area is titled "Deployments" and shows a list of deployments for the "main" branch:

Name	Status	Build duration	Commit message	Started at
Deployment 1	Deployed	2 minutes 39 seconds	Auto-build	10/28/2025, 3:01 PM

Below the table, there's a "Deployment history" section with a search bar and a table showing deployment details. The left sidebar lists other project components like Authentication, Data, AI, Storage, and Functions. The bottom right features "CloudShell" and "Feedback" buttons.

RESULT:

The screenshot shows a Project Management Dashboard titled "EDLIST". On the left, there's a sidebar with a logo for "EDROH TEAM" and a "Private" status. The sidebar includes links for Home, Timeline, Search, Settings, Users, Teams, Projects, Apollo, Beacon, Catalyst, Delta, Echo, Foxtrot, and Golf. The main area has three sections: "Task Priority Distribution" (a bar chart with "Urgent" at ~1.0, "High" at ~2.0, and "Medium" at ~1.0), "Project Status" (a large blue circle with a legend indicating "Completed" tasks), and "Your Tasks" (a table with two rows: "Task 1" (Work In Progress, Urgent, due 2023-04-10) and "Task 12" (To Do, High, due 2023-06-10)).

SOME IMAGES MIGHT NOT LOADED SO FOR THAT FOLLOW SOME STEPS:

The screenshot shows the "Create bucket" configuration page in the AWS S3 console. It starts with "General configuration" where the "AWS Region" is set to "US East (N. Virginia) us-east-1" and the "Bucket type" is set to "General purpose". A note says: "Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones." Below this is a "Bucket name" field containing "mn-s3-images". A note says: "Bucket names must be 3 to 63 characters and unique within the global namespace. Bucket names must also begin and end with a letter or number. Valid characters are a-z, 0-9, periods (.), and hyphens (-). [Learn More](#)". There's a "Copy settings from existing bucket - optional" section with a "Choose bucket" button and a note: "Only the bucket settings in the following configuration are copied." At the bottom of this section is a "Format: s3://bucket/prefix" note. The next section is "Object Ownership" with a note: "Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects." It has two options: "ACLs disabled (recommended)" (selected) and "ACLs enabled". A note for "ACLs disabled": "All objects in this bucket are owned by this account. Access to this bucket and its objects is controlled using only policies." A note for "ACLs enabled": "Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs." The footer includes links for CloudShell, Feedback, and navigation icons.

ITS-NAYAN/project | vpcs | VPC Cons... | User pools > Am... | API Gateway - St... | Instances | EC2 | App Overview | Create S3 bucket | Create Next App | +

us-east-1.console.aws.amazon.com/s3/bucket/create?region=us-east-1&bucketType=general

aws Search [Alt+S] United States (N. Virginia) Account ID: 5332-6705-7426 NAYAN MISHRA

Amazon S3 > Buckets > Create bucket

Object Ownership [Info](#)

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

Object Ownership

ACLs disabled (recommended)
All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

ACLs enabled
Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

Object Ownership
Bucket owner enforced

Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

Block all public access
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

Block public access to buckets and objects granted through new access control lists (ACLs)
S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.

Block public access to buckets and objects granted through any access control lists (ACLs)
S3 will ignore all ACLs that grant public access to buckets and objects.

Block public access to buckets and objects granted through new public bucket or access point policies
S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.

Block public and cross-account access to buckets and objects through any public bucket or access point policies
S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Turning off block all public access might result in this bucket and the objects within becoming public

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

ITS-NAYAN/project | vpcs | VPC Cons... | User pools > Am... | API Gateway - St... | Instances | EC2 | App Overview | Create S3 bucket | Create Next App | +

us-east-1.console.aws.amazon.com/s3/bucket/create?region=us-east-1&bucketType=general

aws Search [Alt+S] United States (N. Virginia) Account ID: 5332-6705-7426 NAYAN MISHRA

Amazon S3 > Buckets > Create bucket

Turning off block all public access might result in this bucket and the objects within becoming public

AWS recommends that you turn on block all public access, unless public access is required for specific and verified use cases such as static website hosting.

I acknowledge that the current settings might result in this bucket and the objects within becoming public.

Bucket Versioning

Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. [Learn more](#)

Bucket Versioning

Disable

Enable

Tags - optional (0)

You can use bucket tags to track storage costs and organize buckets. [Learn more](#)

No tags associated with this bucket.

[Add new tag](#)

You can add up to 50 tags.

Default encryption [Info](#)

Server-side encryption is automatically applied to new objects stored in this bucket.

Encryption type [Info](#)

Secure your objects with two separate layers of encryption. For details on pricing, see DSSE-KMS pricing on the Storage tab of the [Amazon S3 pricing page](#).

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

ITS-NAYAN/project x vpcs | VPC Cons... x User pools > Am... x API Gateway - St... x Instances | EC2 | x App Overview | / x Create S3 bucket x Create Next App x +

us-east-1.console.aws.amazon.com/s3/bucket/create?region=us-east-1&bucketType=general

aws Search [Alt+S] Account ID: 5332-6705-7426 United States (N. Virginia) NAYAN MISHRA

Amazon S3 > Buckets > Create bucket

No tags associated with this bucket.

Add new tag You can add up to 50 tags.

Default encryption Info Server-side encryption is automatically applied to new objects stored in this bucket.

Encryption type Info Secure your objects with two separate layers of encryption. For details on pricing, see DSSE-KMS pricing on the Storage tab of the [Amazon S3 pricing page](#).

Server-side encryption with Amazon S3 managed keys (SSE-S3)

Server-side encryption with AWS Key Management Service keys (SSE-KMS)

Dual-layer server-side encryption with AWS Key Management Service keys (DSSE-KMS)

Bucket Key Using an S3 Bucket Key for SSE-KMS reduces encryption costs by lowering calls to AWS KMS. S3 Bucket Keys aren't supported for DSSE-KMS. [Learn more](#)

Disable

Enable

Advanced settings

After creating the bucket, you can upload files and folders to the bucket, and configure additional bucket settings.

Create bucket Cancel

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

THIS IMAGES ARE IN PUBLIC IN SAME REPO

ITS-NAYAN/project x vpcs | VPC Cons... x User pools > Am... x API Gateway - St... x Instances | EC2 | x App Overview | / x Upload objects x Create Next App x +

us-east-1.console.aws.amazon.com/s3/upload/mn--s3-images?region=us-east-1&bucketType=general

aws Search [Alt+S] Account ID: 5332-6705-7426 United States (N. Virginia) NAYAN MISHRA

Amazon S3 > Buckets > mn--s3-images > Upload

Upload Info Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDKs or Amazon S3 REST API. [Learn more](#)

Drag and drop files and folders you want to upload here, or choose Add files or Add folder.

Files and folders (26 total, 16.3 MB) All files and folders in this table will be uploaded.

<input type="checkbox"/>	Name	Folder	Type	Size
<input type="checkbox"/>	i1.jpg	-	image/jpeg	884.8 KB
<input type="checkbox"/>	i2.jpg	-	image/jpeg	301.2 KB
<input type="checkbox"/>	i3.jpg	-	image/jpeg	1.1 MB
<input type="checkbox"/>	i4.jpg	-	image/jpeg	2.8 MB
<input type="checkbox"/>	i5.jpg	-	image/jpeg	2.3 MB
<input type="checkbox"/>	i6.jpg	-	image/jpeg	2.7 MB
<input type="checkbox"/>	i7.jpg	-	image/jpeg	795.7 KB
<input type="checkbox"/>	i8.jpg	-	image/jpeg	1.5 MB
<input type="checkbox"/>	i9.jpg	-	image/jpeg	1.9 MB
<input type="checkbox"/>	i10.jpg	-	image/jpeg	1.2 MB

Remove Add files Add folder

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

The screenshot shows the 'Edit bucket policy' page in the AWS Management Console. The policy document is as follows:

```
1 Version: "2012-10-17",
2 Statement: [
3     {
4         Sid: "PublicReadGetObject",
5         Effect: "Allow",
6         Principal: "*",
7         Action: "s3:GetObject",
8         Resource: "arn:aws:s3:::mn--s3-images/*"
9     }
10 ]
11
12 }
```

The right side of the screen displays a sidebar with options for 'Edit statement', 'Select a statement', and a button to '+ Add new statement'.

ALSO DO SOME CHANGES IN CODE REPO

The screenshot shows a code editor with the file `next.config.mjs` open. The code defines a `nextConfig` object with an `images` property containing a `remotePatterns` array. One pattern is defined with a protocol of `https`, a hostname of `mn--s3-images.s3.us-east-1.amazonaws.com`, and a pathname of `/**`.

```
1 /**
2  * @type {import('next').NextConfig}
3 */
4 const nextConfig = {
5   images: {
6     remotePatterns: [
7       {
8         protocol: "https",
9         hostname: "mn--s3-images.s3.us-east-1.amazonaws.com",
10        port: "",
11        pathname: "/**",
12      }
13    }
14  };
15
16 export default nextConfig;
```

LEFT CLICK ON FOLDER IN VS CODE AND SELECT FIND IN FOLDER AND REPLACE ALL WITH YOUR S3 URL

A screenshot of the Visual Studio Code interface. The left sidebar shows a tree view of files under the 'client' folder, including 'index.tsx', 'page.tsx', and several 'index.tsx' files from subfolders like 'BoardView', 'Sidebar', 'TaskCard', and 'UserCard'. The main editor tab is 'index.tsx' at line 128. A search bar at the top right says 'project-management'. The status bar at the bottom shows 'Ln 177, Col 65'. The code being edited contains several 'Image' components with 'src' attributes pointing to S3 URLs.

```
const Task = ({ task }: TaskProps) => {
  return (
    <div
      ref={(instance) => {
        drag(instance);
      }}
      className={mb-4 rounded-md bg-white shadow dark:bg-dark-secondary ${isDragging ? "opacity-50" : "opacity-100"}}
    >
      {task.attachments && task.attachments.length > 0 && (
        <Image
          src={`https://mn--s3-images.s3.us-east-1.amazonaws.com/${task.attachments[0].fileURL}`}
          alt={task.attachments[0].fileName}
          width={400}
          height={200}
          className="h-auto w-full rounded-t-md"
        />
      )}
      <div className="p-4 md:p-6">
        <div className="flex items-start justify-between">
          <div className="flex flex-1 flex-wrap items-center gap-2">
            {task.priority && <PriorityTag priority={task.priority} />}
            <div className="flex gap-2">
              ...
            </div>
          </div>
        </div>
      </div>
    </div>
  );
}
```

A screenshot of the Visual Studio Code interface, similar to the first one but with a different set of files selected in the sidebar. The tree view shows files like 'index.tsx' in 'BoardView', 'page.tsx', 'index.tsx' in 'Navbar', 'index.tsx' in 'Sidebar', 'index.tsx' in 'TaskCard', and 'index.tsx' in 'UserCard'. The main editor tab is 'index.tsx' at line 14. The status bar at the bottom shows 'Ln 14, Col 65'. The code being edited contains an 'Image' component with a 'src' attribute pointing to an S3 URL.

```
import { User } from "@/state/api";
import Image from "next/image";
import React from "react";

type Props = {
  user: User;
};

const UserCard = ({ user }: Props) => {
  return (
    <div className="flex items-center rounded border p-4 shadow">
      {user.profilePictureUrl &&
        <Image
          src={`https://mn--s3-images.s3.us-east-1.amazonaws.com/p1.jpeg`}
          alt="profile picture"
          width={32}
          height={32}
          className="rounded-full"
        />
      }
      <div>
        <h3>{user.username}</h3>
        <p>{user.email}</p>
      </div>
    </div>
  );
};
```

COMMIT ALL CHANGES TO GIT HUB

The screenshot shows a VS Code interface with multiple tabs open. In the center, a terminal window displays the following command sequence:

```
PS C:\Users\nayan\Desktop\project-management> git add .
warning: in the working copy of 'client/next.config.mjs', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'client/src/app/projects/BoardView/index.tsx', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'client/src/app/users/page.tsx', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'client/src/components/Navbar/index.tsx', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'client/src/components/Sidebar/index.tsx', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'client/src/components/TaskCard/index.tsx', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'client/src/components/UserCard/index.tsx', LF will be replaced by CRLF the next time Git touches it
PS C:\Users\nayan\Desktop\project-management> git commit -m "S3 IMAGES UPDATED"
[main 4163b37] S3 IMAGES UPDATED
 7 files changed, 10 insertions(+), 10 deletions(-)
PS C:\Users\nayan\Desktop\project-management> git push
Enumerating objects: 39, done.
Counting objects: 100% (39/39), done.
Delta compression using up to 8 threads
Compressing objects: 100% (14/14), done.
Writing objects: 100% (20/20), 1.61 KiB | 274.00 KiB/s, done.
Total 20 (delta 11), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (11/11), completed with 11 local objects.
To https://github.com/ITS-NAYAN/project-management.git
  edc063d..4163b37 main -> main
PS C:\Users\nayan\Desktop\project-management>
```

WAIT FOR SOME TIME

The screenshot shows the AWS Amplify console interface. At the top, there's a navigation bar with various AWS services like VPCs, User pools, API Gateway, Instances, All Apps, and Create Next App. Below the navigation bar, the main header reads "AWS Amplify". The "All apps" tab is selected, showing one app named "project-management". The app card provides details: "Deploying", "Prod branch: main", "Last update: No builds", and "Firewall protections: Not enabled". On the right side of the page, there are links for "Give feedback", "Support", "Docs", "Manage sandboxes", and "Create new app". The bottom of the page includes standard footer links for CloudShell, Feedback, Privacy, Terms, and Cookie preferences.

WANTS TO ADD DATABASE IN NEXT PART