

Lab 7

AWS RDS

Software Studio
Datalab, CS, NTHU
2023

Gitlab

lab-weathermood-server-db-todo

The screenshot displays the GitLab web interface for a repository named 'lab-weathermood-server-db-todo'. The interface includes a sidebar with navigation options such as 'Project information', 'Repository', 'Files', 'Commits', 'Branches', 'Tags', 'Contributors', 'Graph', 'Compare', 'Issues', 'Merge requests', 'Security & Compliance', 'Deployments', 'Monitor', 'Infrastructure', 'Packages & Registries', 'Analytics', 'Wiki', 'Snippets', and 'Settings'. The main content area shows the repository's file structure and a table of files. A recent commit titled 'Update README.md' by Sheng-ya Chiu is highlighted. Below the table, the content of the 'README.md' file is displayed, including a title 'Weathermood Server BD TODO' and a list of requirements for an assignment.

courses > ... > 2023-spring > lab-weathermood-server-db-todo > Repository

master lab-weathermood-server-db-todo / + History Find file Web IDE 3196bcd8 Clone

Update README.md
Sheng-ya Chiu authored just now

Name	Last commit	Last update
client	initial commit	3 minutes ago
server	initial commit	3 minutes ago
.gitignore	initial commit	3 minutes ago
README.md	Update README.md	just now

README.md

Weathermood Server BD TODO

In this assignment, you are asked to improve your todo function on the "Forecast" page using relational database.

Requirement

On the "Forecast" page:

1. Deploy to AWS (Must!!! or you will get a 0. You may need to use AWS RDS)
2. (80%) Store and get data from DB (So you maybe need to design the DB schema).
3. (10%) The pagination function.
4. (10%) The "Unaccomplished" function and the "Accomplished" function should be server-side not simulated server-side like this

From local - file - db system

```
function list(searchText = '') {
  return new Promise((resolve, reject) => {
    if (!fs.existsSync('data-posts.json')) {
      fs.writeFileSync('data-posts.json', '');
    }

    fs.readFile('data-posts.json', 'utf8', (err, data) => {
      if (err) reject(err);

      let posts = data ? JSON.parse(data) : [];
      if (posts.length > 0 && searchText) {
        posts = posts.filter(p => {
          return p.text.toLowerCase().indexOf(searchText.toLowerCase()) !== -1
        });
      }
      resolve(posts);
    });
  });
}
```

branch file

```
export function listPosts(searchText = '', start) {
  let url = `${postBaseUrl}/posts`;
  let query = [];
  if (searchText)
    query.push(`searchText=${searchText}`);
  if (start)
    query.push(`start=${start}`);
  if (query.length)
    url += '?' + query.join('&');

  console.log(`Making GET request to: ${url}`);

  return axios.get(url).then(function(res) {
    if (res.status !== 200)
      throw new Error(`Unexpected response code: ${res.status}`);
    return res.data;
  });
}
```

branch db

Outline

1. AWS RDS (relational database service)
2. EB / RDS connection
3. Setup weathermood db in RDS
4. Application setting and deploy

Find RDS

Click create database

The screenshot shows the Amazon RDS console interface. At the top, there is a navigation bar with the AWS logo, 'Services', a search bar, and a region dropdown set to 'Oregon'. On the left, a sidebar lists various RDS-related features like 'Dashboard', 'Databases', 'Query Editor', etc. The main content area features a prominent blue information banner at the top with an 'i' icon, titled 'Try the new Amazon RDS Multi-AZ deployment option for MySQL and PostgreSQL'. The banner text describes performance improvements and includes a blue-bordered box containing an orange 'Create database' button. Below the banner, the 'Resources' section shows usage statistics for DB Instances, Clusters, Snapshots, and other resources in the US West (Oregon) region. To the right, a 'Recommended for you' section lists several guides and articles related to RDS, such as 'Test Your DR Strategy in Minutes' and 'Migrate SSRS to RDS for SQL Server'.

Create database

Standard create; PostgreSQL

Amazon RDS ×

RDS > Create database

Create database

Choose a database creation method [Info](#)

- 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](#)

- Aurora (MySQL Compatible)
- Aurora (PostgreSQL Compatible)
- MySQL
- MariaDB
- PostgreSQL
- Oracle
- Microsoft SQL Server

Create database

Engine version (default); Free tier

Amazon RDS ×

- Dashboard
- Databases
- Query Editor
- Performance insights
- Snapshots
- Exports in Amazon S3
- Automated backups
- Reserved instances
- Proxies

- Subnet groups
- Parameter groups
- Option groups
- Custom engine versions

- Events
- Event subscriptions

- Recommendations **0**
- Certificate update

Engine Version

PostgreSQL 14.6-R1 ▼

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](#)

The deployment options below are limited to those supported by the engine you selected above.

- Multi-AZ DB Cluster - new**
Creates a DB cluster with a primary DB instance and two readable standby DB instances, with each DB instance in a different Availability Zone (AZ). Provides high availability, data redundancy and increases capacity to serve read workloads.
- Multi-AZ DB instance (not supported for Multi-AZ DB cluster snapshot)**
Creates a primary DB instance and a standby DB instance in a different AZ. Provides high availability and data redundancy, but the standby DB instance doesn't support connections for read workloads.
- Single DB instance (not supported for Multi-AZ DB cluster snapshot)**
Creates a single DB instance with no standby DB instances.

Create database

DB instance (name is up to you);
Master username & password (up to you too but must keep it)

The screenshot shows the Amazon RDS console interface. On the left is a navigation sidebar with the following items: Dashboard (highlighted in orange), Databases, Query Editor, Performance insights, Snapshots, Exports in Amazon S3, Automated backups, Reserved instances, Proxies, Subnet groups, Parameter groups, Option groups, Custom engine versions, Events, Event subscriptions, Recommendations (with a '0' badge), and Certificate update. The main content area is titled 'Settings' and contains the following sections:

- DB instance identifier** [Info](#): A text input field containing 'lab-db-demo'. Below it, a note states: 'The DB instance identifier is case-insensitive, but is stored as all lowercase (as in "mydbinstance"). Constraints: 1 to 60 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](#): A text input field containing 'postgres'. Below it, a note states: '1 to 16 alphanumeric characters. First character must be a letter.'
 - Manage master credentials in AWS Secrets Manager**
Manage master user credentials in Secrets Manager. RDS can generate a password for you and manage it throughout its lifecycle.
- Info** box: 'If you manage the master user credentials in Secrets Manager, some RDS features aren't supported. [Learn more](#)'
- Auto generate a password**
Amazon RDS can generate a password for you, or you can specify your own password.
- Master password** [Info](#): A password input field with masked characters. Below it, a note states: 'Constraints: At least 8 printable ASCII characters. Can't contain any of the following: / (slash), ' (single quote), " (double quote) and @ (at sign).'
- Confirm master password** [Info](#): A second password input field with masked characters.

Create database

Default; Uncheck “Enable storage autoscaling”

The screenshot displays the Amazon RDS console interface. On the left is a navigation sidebar with the following items: Dashboard (highlighted in orange), Databases, Query Editor, Performance insights, Snapshots, Exports in Amazon S3, Automated backups, Reserved instances, Proxies, Subnet groups, Parameter groups, Option groups, Custom engine versions, Events, Event subscriptions, Recommendations (with a notification badge showing '0'), and Certificate update. The main content area is titled 'Instance configuration' and includes a sub-section for 'Storage'. In the 'Instance configuration' section, the 'DB instance class' is set to 'db.t3.micro' (2 vCPUs, 1 GiB RAM, Network: 2,085 Mbps), and the 'Include previous generation classes' checkbox is unchecked. In the 'Storage' section, the 'Storage type' is 'General Purpose SSD (gp2)', and the 'Allocated storage' is set to '200 GiB'. The 'Storage autoscaling' section has the 'Enable storage autoscaling' checkbox unchecked. A note below this checkbox states: 'Enabling this feature will allow the storage to increase after the specified threshold is exceeded.'

Create database

Public access Yes; other default

The screenshot shows the Amazon RDS console interface. On the left is a navigation sidebar with the following items: Amazon RDS (with a close icon), Dashboard, Databases, Query Editor, Performance insights, Snapshots, Exports in Amazon S3, Automated backups, Reserved instances, Proxies, Subnet groups, Parameter groups, Option groups, Custom engine versions, Events, Event subscriptions, Recommendations (with a '0' notification badge), and Certificate update. The main content area is titled 'Connectivity' and includes an 'Info' link and a refresh icon. It is divided into several sections: 'Compute resource' with two radio button options: 'Don't connect to an EC2 compute resource' (selected) and 'Connect to an EC2 compute resource'; 'Network type' with two radio button options: 'IPv4' (selected) and 'Dual-stack mode'; 'Virtual private cloud (VPC)' with a dropdown menu showing 'Default VPC (vpc-0c549062274c6c2d7)' and a note that 'After a database is created, you can't change its VPC.'; 'DB subnet group' with a dropdown menu showing 'default'; and 'Public access' with two radio button options: 'Yes' (selected and highlighted with a blue box) and 'No'. The 'Yes' option description states: '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.' The 'No' option description states: '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.' At the bottom, there is a 'VPC security group (firewall)' section with an 'Info' link and a note: 'Choose one or more VPC security groups to allow access to your database. Make sure that the security group rules allow the...'. The footer of the console shows 'CloudShell', 'Feedback', 'Language', '© 2023, Amazon Web Services, Inc. or its affiliates.', 'Privacy', 'Terms', and 'Cookie preferences'.

Create database

Password authentication

Amazon RDS ×

- Dashboard
- Databases
- Query Editor
- Performance insights
- Snapshots
- Exports in Amazon S3
- Automated backups
- Reserved instances
- Proxies

- Subnet groups
- Parameter groups
- Option groups
- Custom engine versions

- Events
- Event subscriptions

- Recommendations **0**
- Certificate update

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

Turn on Performance Insights

▼ **Additional configuration**
Enhanced Monitoring

Monitoring

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

► **Additional configuration**
Database options, encryption turned on, backup turned on, backtrack turned off, maintenance, CloudWatch Logs, delete protection turned off.

Create database

Make sure it is free tier; create database

Proxies

Subnet groups

Parameter groups

Option groups

Custom engine versions

Events

Event subscriptions

Recommendations 0

Certificate update

► Additional configuration

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

Estimated monthly costs

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:

- 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).
- 20 GB for automated backup storage and any user-initiated DB Snapshots.

[Learn more about AWS Free Tier.](#)

When your free usage expires or if your application use exceeds the free usage tiers, you simply pay standard, pay-as-you-go service rates as described in the [Amazon RDS Pricing page.](#)

i You are responsible for ensuring that you have all of the necessary rights for any third-party products or services that you use with AWS services.

Cancel **Create database**

CloudShell Feedback Language © 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Database

Amazon RDS



Dashboard

- Databases
- Query Editor
- Performance insights
- Snapshots
- Exports in Amazon S3
- Automated backups
- Reserved instances
- Proxies

- Subnet groups
- Parameter groups
- Option groups
- Custom engine versions

- Events
- Event subscriptions

- Recommendations 0
- Certificate update

RDS > Databases > lab-db-demo

lab-db-demo

Modify

Actions ▼

Summary

DB identifier lab-db-demo	CPU -	Status ⌚ Backing-up	Class db.t3.micro
Role Instance	Current activity 0 Connections	Engine PostgreSQL	Region & AZ us-west-2c

Connectivity & security

Monitoring

Logs & events

Configuration

Maintenance & backups

Tags

Connectivity & security

Endpoint & port

Endpoint
lab-db-demo.cvak
west-2.rds.amazonaws.com

Port
5432

Networking

Availability Zone
us-west-2c

VPC
[vpc-0c5490](#)

Subnet group
default-vpc-0c549062274c6c2d7

Subnets
[subnet-03ffc85](#)
[subnet-0a6b0e](#)
subnet-

Security

VPC security groups
[default \(sg-0114f93538\)](#)
✔ Active

Publicly accessible
No

Certificate authority [Info](#)
rds-ca-2019

Certificate authority date
August 23, 2024, 01:08
(UTC+08:00)

Outline

1. AWS RDS (relational database service)
2. EB / RDS connection
3. Setup weathermood db in RDS
4. Application setting and deploy

EB + RDS

Find EB - configuration - instances traffic edit

Elastic Beanstalk ×

Applications
Environments
Change history

▼ Application: weathermood-server-db-todo

- Application versions
- Saved configurations

▼ Environment: weathermood-server-db-todo-dev

- Go to environment [↗](#)
- Configuration**
- Events
- Health
- Logs
- Monitoring
- Alarms
- Managed updates
- Tags

▼ Recent environments

- weathermood-server-db-todo-dev

Elastic Beanstalk > Environments > weathermood-server-db-todo-dev > Configuration ⓘ

Configuration [Info](#)

[Cancel](#) [Review changes](#) [Apply changes](#)

Service access [Info](#)
Configure the service role and EC2 instance profile that Elastic Beanstalk uses to manage your environment. Choose an EC2 key pair to securely log in to your EC2 instances.

[Edit](#)

Service role	EC2 instance profile
aws-elasticbeanstalk-service-role	aws-elasticbeanstalk-ec2-role

Networking, database, and tags [Info](#)
Configure VPC settings, and subnets for your environment's EC2 instances and load balancer. Set up an Amazon RDS database that's integrated with your environment.

[Edit](#)

No options configured

Instance traffic and scaling [Info](#)
Customize the capacity and scaling for your environment's instances. Select security groups to control instance traffic. Configure the software that runs on your environment's instances by setting platform-specific options.

[Edit](#)

Instances

IMDSv1	EC2 Security Groups
--------	---------------------

Database

Setup RDS security groups to allow ingress from machines in the same group. Go check out your DB security group first.

The screenshot displays the Amazon RDS console interface for a database instance named 'lab-db-demo'. The left sidebar shows the navigation menu with 'Amazon RDS' at the top and various options like 'Dashboard', 'Databases', 'Query Editor', etc. The main content area shows the instance details under the 'Connectivity & security' tab. A summary table provides key information about the instance, and the 'Connectivity & security' section details the endpoint, networking, and security configurations.

Summary			
DB identifier	CPU	Status	Class
lab-db-demo	-	⌚ Backing-up	db.t3.micro
Role	Current activity	Engine	Region & AZ
Instance	0 Connections	PostgreSQL	us-west-2c

Connectivity & security		
Endpoint & port	Networking	Security
Endpoint lab-db-demo.cvak west-2.rds.amazonaws.com	Availability Zone us-west-2c VPC vpc-0c5490	VPC security groups default (sg-0114f93538) ✔ Active Publicly accessible

EB + RDS

Find EB - configuration - instances traffic edit

Elastic Beanstalk ×

Applications
Environments
Change history

▼ Application: weathermood-server-db-todo

- Application versions
- Saved configurations

▼ Environment: weathermood-server-db-todo-dev

- Go to environment [↗](#)
- Configuration**
- Events
- Health
- Logs
- Monitoring
- Alarms
- Managed updates
- Tags

▼ Recent environments

- weathermood-server-db-todo-dev

Elastic Beanstalk > Environments > weathermood-server-db-todo-dev > Configuration ⓘ

Configuration [Info](#)

Service access [Info](#)
Configure the service role and EC2 instance profile that Elastic Beanstalk uses to manage your environment. Choose an EC2 key pair to securely log in to your EC2 instances.

Service role	EC2 instance profile
aws-elasticbeanstalk-service-role	aws-elasticbeanstalk-ec2-role

Networking, database, and tags [Info](#)
Configure VPC settings, and subnets for your environment's EC2 instances and load balancer. Set up an Amazon RDS database that's integrated with your environment.

No options configured

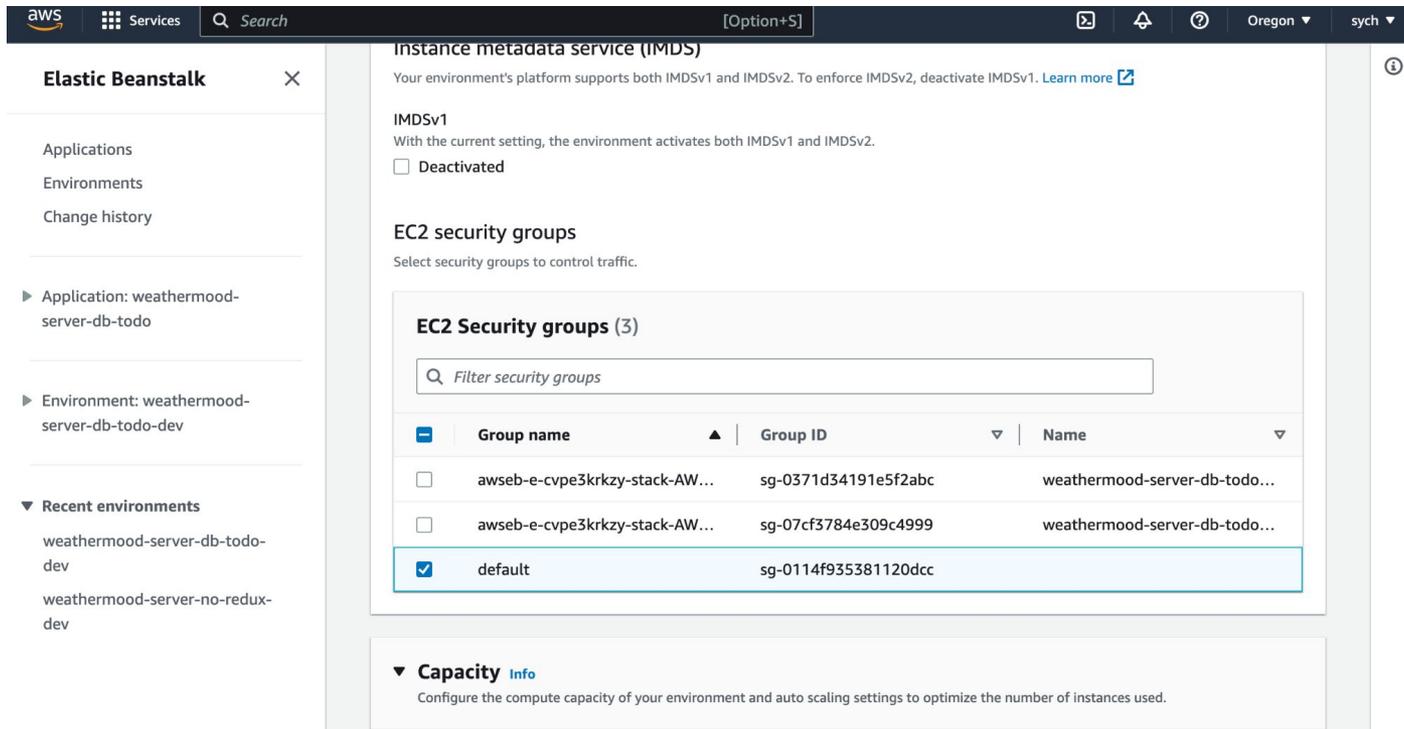
Instance traffic and scaling [Info](#)
Customize the capacity and scaling for your environment's instances. Select security groups to control instance traffic. Configure the software that runs on your environment's instances by setting platform-specific options.

Instances

IMDSv1	EC2 Security Groups
--------	---------------------

EB + RDS

Click the security group from RDS then apply (button is at the bottom)



The screenshot shows the AWS Elastic Beanstalk console. The left sidebar is titled "Elastic Beanstalk" and contains a search icon, "Applications", "Environments", "Change history", and two application/environment entries: "Application: weathermood-server-db-todo" and "Environment: weathermood-server-db-todo-dev". Under "Recent environments", there are two entries: "weathermood-server-db-todo-dev" and "weathermood-server-no-redux-dev".

The main content area is titled "Instance metadata service (IMDS)". It contains a section for "IMDSv1" with a "Deactivated" checkbox. Below that is the "EC2 security groups" section, which includes a search box "Filter security groups" and a table of security groups. The table has columns for "Group name", "Group ID", and "Name". The "default" security group (Group ID: sg-0114f935381120dcc) is selected with a checked checkbox. Below the table is a "Capacity" section with an "Info" link and a description: "Configure the compute capacity of your environment and auto scaling settings to optimize the number of instances used."

	Group name	Group ID	Name
<input type="checkbox"/>	awseb-e-cvpe3krkzy-stack-AW...	sg-0371d34191e5f2abc	weathermood-server-db-todo...
<input type="checkbox"/>	awseb-e-cvpe3krkzy-stack-AW...	sg-07cf3784e309c4999	weathermood-server-db-todo...
<input checked="" type="checkbox"/>	default	sg-0114f935381120dcc	

Configure security group

RDS -> click VPC security groups

The screenshot shows the Amazon RDS console interface. On the left is a navigation sidebar with 'Amazon RDS' at the top and a list of options including 'Dashboard', 'Databases', 'Query Editor', 'Performance insights', 'Snapshots', 'Exports in Amazon S3', 'Automated backups', 'Reserved instances', 'Proxies', 'Subnet groups', 'Parameter groups', 'Option groups', 'Custom engine versions', 'Events', 'Event subscriptions', and 'Recommendations'. The main content area shows the 'lab-db-demo' database instance page. At the top right of this page are 'Modify' and 'Actions' buttons. Below the instance title is a 'Summary' section with a table of key attributes:

DB identifier	CPU	Status	Class
lab-db-demo	-	Backing-up	db.t3.micro
Role	Current activity	Engine	Region & AZ
Instance	0 Connections	PostgreSQL	us-west-2c

Below the summary is a horizontal menu with tabs: 'Connectivity & security' (selected), 'Monitoring', 'Logs & events', 'Configuration', 'Maintenance & backups', and 'Tags'. The 'Connectivity & security' section contains three columns: 'Endpoint & port', 'Networking', and 'Security'. The 'Security' column is highlighted with a blue box and shows 'VPC security groups' with a link to 'default (sg-0114f93538)' and a green 'Active' status icon. Below this, it indicates 'Publicly accessible'.

Configure security group

Inbound rules -> edit inbound rules

The screenshot displays the AWS IAM console interface for configuring a security group. The top section, titled "Security Groups (1/1)", includes a search bar with the filter "search: sg-0114f935381120dcc" and a table listing the security group. The table has columns for Name, Security group ID, Security group name, VPC ID, Description, and Owner. The "Inbound rules" tab is selected and highlighted with a blue box. Below this, a notification banner for Reachability Analyzer is visible. The "Inbound rules (1/1)" section contains a search bar and a table of rules. The "Edit inbound rules" button is highlighted with a blue box.

Name	Security group ID	Security group name	VPC ID	Description	Owner
-	sg-0114f935381120dcc	default	vpc-0c549062274c6c2d7	default VPC security gr...	785490623

Name	Security group rule...	IP version	Type	Protocol	Port range
-	sgr-0225eae5fbe30c7ac	-	All traffic	All	All

Configure security group

1. Type postgresQL; source custom -> choose the security group from RDS
Add rule
2. Type postgresQL; source my ip
Save rules

EC2 > Security Groups > sg-0114f935381120dcc - default > Edit inbound rules

Edit inbound rules [Info](#)

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

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info	Description - optional Info
sgr-0225eae5fbe30c7ac	All traffic	All	All	Custom <input type="text" value="Q"/>	<input type="text"/>

EB + RDS

RDS > database > public accessibility should be Yes, modify if No:
From RDS > Databases > modify > public access

Amazon RDS ×

- Dashboard
- Databases**
- Query Editor
- Performance insights
- Snapshots
- Exports in Amazon S3
- Automated backups
- Reserved instances
- Proxies

- Subnet groups
- Parameter groups
- Option groups
- Custom engine versions

- Events
- Event subscriptions

Security group
List of DB security groups to associate with this DB instance.

Choose security groups ▼

default ×

Certificate authority [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-2019 ▼

▼ **Additional configuration**

Public access

Publicly accessible
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.

Not publicly accessible
No IP address is assigned to the DB instance. EC2 instances and devices outside the VPC can't connect.

Database port
Specify the TCP/IP port that the DB instance will use for application connections. The application connection string must specify the port number. The DB security group and your firewall must allow connections to the port. [Learn more](#) [↗](#)

5432

Outline

1. AWS RDS (relational database service)
2. EB / RDS connection
3. **Setup weathermood db in RDS**
4. Application setting and deploy

Create database in RDS

Server cmd: `createdb -h <RDS_endpoint> -p 5432 -U <RDS_master_name> weathermood`

Then enter password
To create weathermood db

```
(base) sasaya@sasayadeMBP server % createdb -h lab-db-demo.cvakngwhzvjq.us-west-2.rds.amazonaws.com -p 5432 -U postgres weathermood
Password:
Password:
```

The screenshot shows the AWS Management Console for an RDS instance. The left sidebar contains navigation options: Subnet groups, Parameter groups, Option groups, Custom engine versions, Events, and Event subscriptions. The main content area is titled 'Connectivity & security' and includes tabs for Monitoring, Logs & events, Configuration, Maintenance & backups, and Tags. The 'Connectivity & security' section is expanded, showing three columns: Endpoint & port, Networking, and Security. The 'Endpoint & port' column is highlighted with a blue box and contains the endpoint 'lab-db-demo.cvakngwhzvjq.us-west-2.rds.amazonaws.com'. The 'Networking' column shows 'Availability Zone: us-west-2c' and 'VPC: vpc-0c5490'. The 'Security' column shows 'VPC security groups: default (sg-0114f93538)' and 'Active'.

Find endpoint in RDS

Create database in RDS

Server cmd: `psql -h <RDS_endpoint> -U <RDS_master_name>`

Then enter password

Then `\c weathermood` if you are in other db

Make sure you are connected to weathermood db

```
postgres=> \dt
          List of relations
 Schema | Name  | Type  | Owner
-----+-----+-----+-----
 public | posts | table | postgres
 public | todos | table | postgres
(2 rows)
```

```
postgres=> \q
```

Options:

`-h: host`

`-p: port`

`-U: username`

`-d: dbname`

The one you connected to

Create database in RDS

Define schema / create table

- Method 1: Migrate schema

- `pg_dump -h <dev-server> -U <dev-user> --no-owner --schema-only -c weathermood > db.dump`
- `psql -h <rds-endpoint> -U <res-user> weathermood < db.dump`

- Method 2: Connect to remote psql server first and manually create

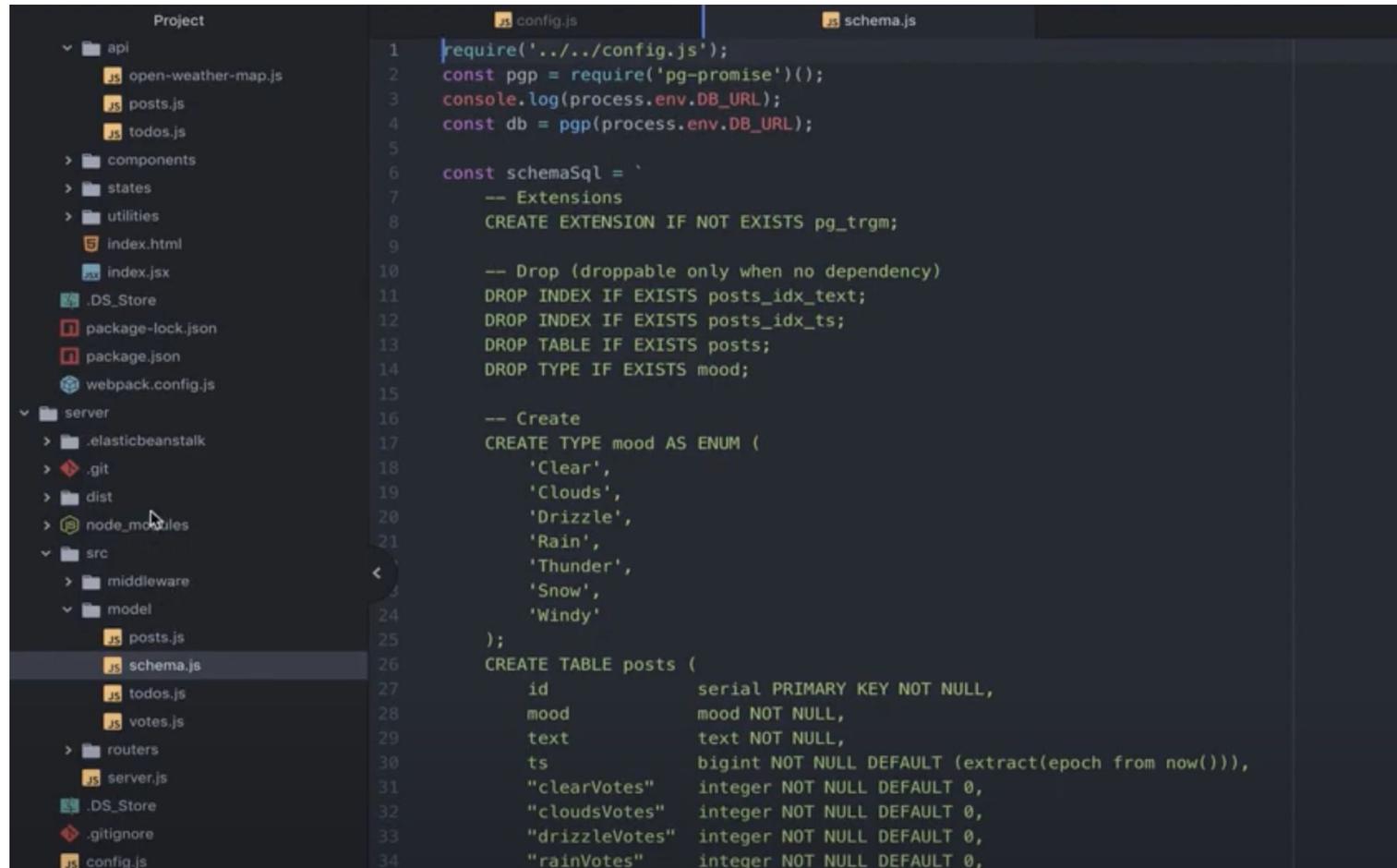
```
weathermood=> CREATE TABLE posts (  
weathermood(>     id          serial PRIMARY KEY NOT NULL,  
weathermood(>     mood        mood NOT NULL,  
weathermood(>     text         text NOT NULL,  
weathermood(>     ts           bigint NOT NULL DEFAULT (extract(epoch from now())),  
weathermood(>     "clearVotes" integer NOT NULL DEFAULT 0,  
weathermood(>     "cloudsVotes" integer NOT NULL DEFAULT 0,  
weathermood(>     "drizzleVotes" integer NOT NULL DEFAULT 0,  
weathermood(>     "rainVotes"   integer NOT NULL DEFAULT 0,  
weathermood(>     "thunderVotes" integer NOT NULL DEFAULT 0,  
weathermood(>     "snowVotes"   integer NOT NULL DEFAULT 0,  
weathermood(>     "windyVotes"  integer NOT NULL DEFAULT 0  
weathermood(> );
```

Create database in RDS

Generate dummy data

```
weathermood=# INSERT INTO posts (mood, text)
weathermood-#     SELECT
weathermood-#         'Clear',
weathermood-#         'word' || i || ' word' || (i+1) || ' word' || (i+2)
weathermood-#     FROM generate_series(1, 100) AS s(i);
INSERT 0 100
```

Create database in RDS



```
1 require('../../config.js');
2 const pgp = require('pg-promise')();
3 console.log(process.env.DB_URL);
4 const db = pgp(process.env.DB_URL);
5
6 const schemaSql = `
7   -- Extensions
8   CREATE EXTENSION IF NOT EXISTS pg_trgm;
9
10  -- Drop (droppable only when no dependency)
11  DROP INDEX IF EXISTS posts_idx_text;
12  DROP INDEX IF EXISTS posts_idx_ts;
13  DROP TABLE IF EXISTS posts;
14  DROP TYPE IF EXISTS mood;
15
16  -- Create
17  CREATE TYPE mood AS ENUM (
18    'Clear',
19    'Clouds',
20    'Drizzle',
21    'Rain',
22    'Thunder',
23    'Snow',
24    'Windy'
25  );
26  CREATE TABLE posts (
27    id          serial PRIMARY KEY NOT NULL,
28    mood       mood NOT NULL,
29    text       text NOT NULL,
30    ts         bigint NOT NULL DEFAULT (extract(epoch from now())),
31    "clearVotes" integer NOT NULL DEFAULT 0,
32    "cloudsVotes" integer NOT NULL DEFAULT 0,
33    "drizzleVotes" integer NOT NULL DEFAULT 0,
34    "rainVotes" integer NOT NULL DEFAULT 0,
```

Outline

1. AWS RDS (relational database service)
2. EB / RDS connection
3. Setup weathermood db in RDS
4. Application setting and deploy

Application setting and deploy

Step 1: Add environments variables on EB environments

- Method 1: Using EB CLI
 - **eb setenv** NODE_ENV=production, RDS_HOSTNAME=...
 - **Reminder:** you also need **RDS_PASSWORD** for the DB_URL, check *server/config.js*
- Method 2: Using AWS console: EB -> Environments -> Configuration -> Software

The screenshot shows the AWS Management Console interface for an Elastic Beanstalk environment named 'weathermood-server-dev'. On the left, a navigation menu includes 'Go to environment', 'Configuration' (highlighted in orange), 'Logs', 'Health', 'Monitoring', 'Alarms', 'Managed updates', 'Events', and 'Tags'. The main content area displays the configuration for the 'Software' category. A search bar at the top of the configuration section contains the text 'Search for an option name or value'. Below the search bar, a table lists environment options. The 'Software' category is selected, showing the following options: 'Environment properties: NODE_ENV, RDS_DB_NAME, RDS_HOSTNAME, RDS_PASSWORD, RDS_PORT, RDS_USERNAME', 'Log streaming: disabled', 'Proxy server: nginx', 'Rotate logs: disabled', and 'X-Ray daemon: disabled'. An 'Edit' button is located in the 'Actions' column for the 'Software' category and is highlighted with a red rectangular border.

Category	Options	Actions
Software	Environment properties: NODE_ENV, RDS_DB_NAME, RDS_HOSTNAME, RDS_PASSWORD, RDS_PORT, RDS_USERNAME Log streaming: disabled Proxy server: nginx Rotate logs: disabled X-Ray daemon: disabled	Edit

Application setting and deploy

```
config.js
1  require('dotenv').config();
2
3  try {
4    switch (process.env.NODE_ENV) {
5      case 'development':
6        process.env.DB_URL = `postgres://${process.env.PG_USERNAME}@${process.env.PG_HOST}`;
7        break;
8      default:
9        // 'staging' or 'production'
10       process.env.DB_URL = `postgres://${process.env.RDS_USERNAME}:${process.env.RDS_PASSWORD}@${process.env.RDS_HOST}`;
11       break;
12     }
13     // only used for debugging
14     console.log(`==DEBUG== process.env.DB_URL = ${process.env.DB_URL}`);
15   } catch (err) {
16     console.log(
17       err,
18       '\n\nError configuring the project. Have you set the environment variables?'
19     );
20   }
21 }
```

Application setting and deploy

Method 1: cmd

```
$ eb setenv NODE_ENV=production \
  RDS_HOSTNAME=<rds-endpoint> RDS_PORT=5432 \
  RDS_USERNAME=<user> RDS_PASSWORD=<password> \
  RDS_DB_NAME=weathermood
```

Application setting and deploy

Method 2: EB > config > edit > add properties

The screenshot displays the AWS Elastic Beanstalk console interface. On the left, a navigation sidebar shows the path: Environment: s-weathermood-server-2021-db-dev > Configuration. The main content area is titled 'Load balancer' and shows the following settings:

Property	Value	Property	Value
Load balancer visibility	public	Load balancer type	application
Store logs	Deactivated	Load balancer is shared	false

Below the load balancer settings, there is a section for 'Updates, monitoring, and logging' with an 'Edit' button highlighted by a blue box. This section includes a description: 'Define when and how Elastic Beanstalk deploys changes to your environment. Manage your application's monitoring and logging settings, instances, and other environment resources.'

The 'Monitoring' section shows the following settings:

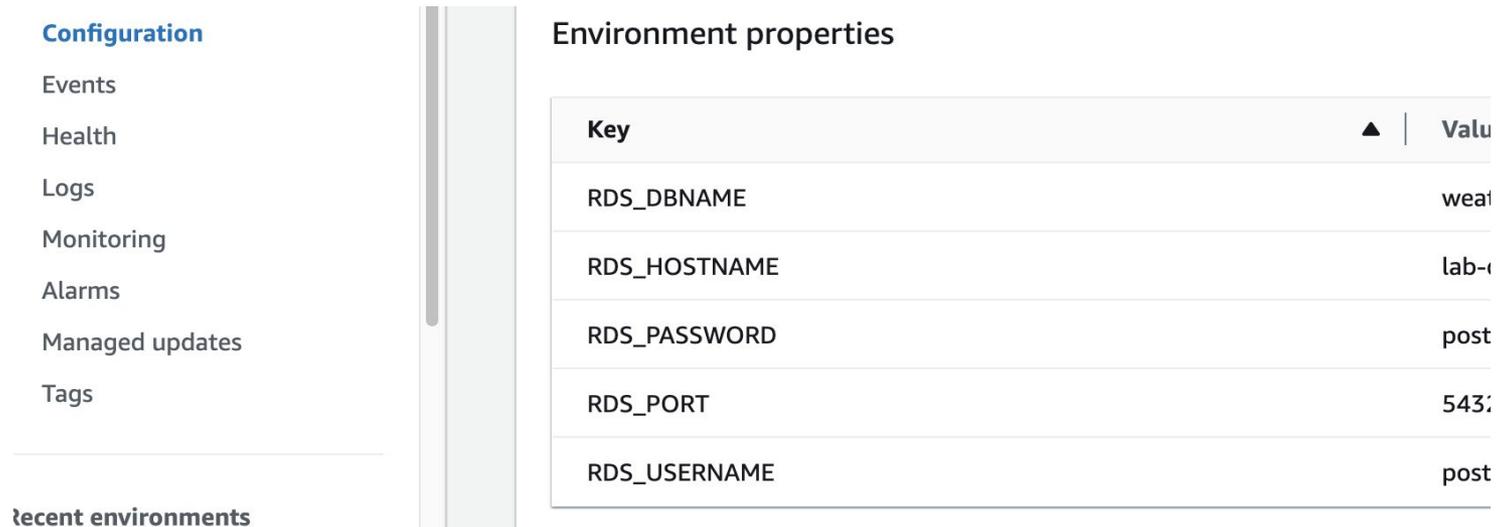
Property	Value	Property	Value
System	enhanced	Cloudwatch custom metrics - instance	—
Log streaming	Deactivated	Cloudwatch custom metrics - environment	—
		Retention	7
		Lifecycle	false

The 'Updates' section shows the following settings:

Property	Value	Property	Value
Managed updates	Deactivated	Update batch size	1
		Deployment batch size	30

Application setting and deploy

Method 2: EB > config > edit > add properties



The screenshot displays the AWS Elastic Beanstalk configuration interface. On the left, a sidebar menu lists various configuration options: Configuration (highlighted in blue), Events, Health, Logs, Monitoring, Alarms, Managed updates, and Tags. Below the menu, the text 'Recent environments' is visible. The main content area is titled 'Environment properties' and contains a table with the following data:

Key	Value
RDS_DBNAME	weat
RDS_HOSTNAME	lab-r
RDS_PASSWORD	post
RDS_PORT	543
RDS_USERNAME	post

Application setting and deploy

Step 2: Configuration before deploy

- Change `postBaseUrl` to your server
- Change `OpenWeatherAPI Key` to your key
- Build client project and copy *dist* to the server project

Step 3: Deploy to EB

- Commit before deploy
- `eb deploy <environment>`
- **Reminder:** You need to specify the environment this time

Fork then clone project

1. Client side code in the client folder
2. Server side code in the server folder
3. **npm install** both first to get all the packages

// if you have other instances on EB, you may consider terminate it