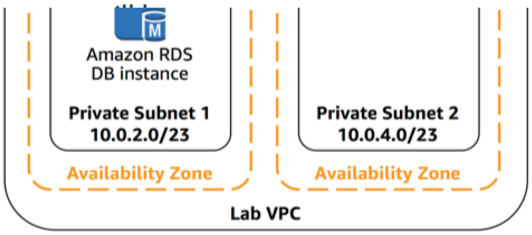
Deploying a Web Application on AWS

Traditional methods of deploying servers and configuring security are complex and often involve multiple teams and long delays. Fortunately, it is quick and easy to deploy secure infrastructure in the cloud.

In this lab you will:

- Launch a database using **Amazon RDS**
- Launch an application server using **Amazon EC2**
- Automatically install an **application**



Duration

This lab will require approximately **30 minutes** to complete.

for this lab. The lab instructions are **not compatible with *Internet Explorer*** due to a difference in the Amazon RDS console.

- Click **Start Lab** to launch the lab.

- Click Open Console

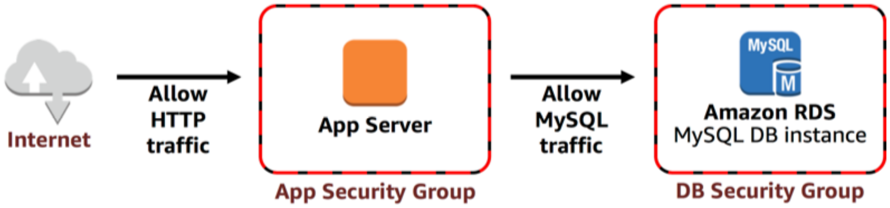
- Sign in to the AWS Management Console using the credentials shown to the left of these instructions.

⚠ Please do not change the Region during this lab.

Task 1: Configure Security

Security should be implemented at every layer of your architecture — in the application, on the server, within the network and when connecting to the internet.

In this task, you will define **Security Groups** for the Amazon EC2 application server and Amazon RDS database instance:



A *security group* acts as a virtual firewall that controls the traffic for one or more instances. When you launch an instance, you associate one or more security groups with the instance. You add *rules* to each security group that allow traffic to or from its associated instances. You can modify the rules for a security group at any time; the new rules are automatically applied to all instances that are associated with the security group.

First, you will create the **App Security Group**. It will be configured to permit incoming HTTP connections from the Internet.

1. In the **AWS Management Console**, on the Services menu, click **EC2**.

2. In the left navigation pane, click **Security Groups**.

Task 2: Create an Amazon RDS Database
Task 3: Launch an Application Server using Amazon EC2
Task 4: Test the Application
Lab Complete
Task 3: Launch an Application Server using Amazon EC2
Task 4: Test the Application
Lab Complete

Task 2: Create an Amazon RDS Database
Task 3: Launch an Application Server using Amazon EC2
Task 4: Test the Application
Lab Complete

Task 2: Create an Amazon RDS Database
Task 3: Launch an Application Server using Amazon EC2
Task 4: Test the Application
Task 3: Launch an Application Server using Amazon EC2

Some existing security groups will be displayed. You will now create a new security group for the application servers.

3. Click **Create Security Group** and configure:

   - **Security group name:** `App-SG`
   - **Description:** `Allow HTTP access`
   - **VPC:** *Lab VPC*

4. Click **Add Rule** then configure:

   - **Type:** *HTTP*
   - **Source:** *Anywhere*
   - Click **Create**

   You will use this Security Group later in the lab when launching an application server.

   Next, you will configure the **Database Security Group**. It will be configured to permit incoming database connections from the App Server.

   First, you will copy the ID of the *App-SG* that you just created.

5. Select ☑ **App-SG**.

6. Copy the **Group ID** displayed in the **Description** tab in the lower half of the page.

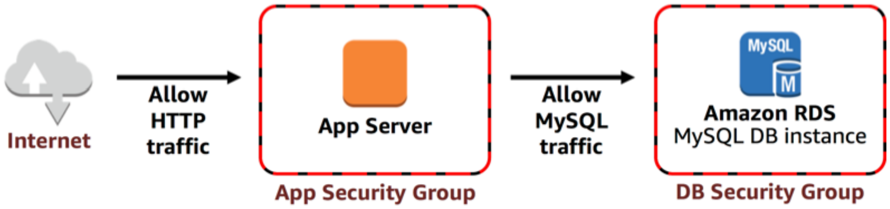
   Tip: Simply hover over the Group ID and click 📋 Copy.

   It will look similar to: *sg-1234abcd*

7. Click **Create Security Group** and configure:

   - **Security group name:** `DB-SG`
   - **Description:** `Allow DB access`
   - **VPC:** *Lab VPC*
   - Click **Add Rule** then configure:
     - **Type:** *MYSQL/Aurora*
     - **Source:** Paste the *Group ID* that you copied in the previous step
   - Click **Create**

This configuration means that *the Database security group (DB-SG) is permitting Inbound access from the Application security group (App-SG)*.



The ability for one security group to refer to another security group is a powerful capability. It means that additional EC2 instances can be granted access to the database by simply associating them with *App-SG*. Any instance associated with the *App-SG* will then be permitted to communicate to the database (or, more accurately, to any database associated with the *DB-SG* security group).

You will use this Database security group in the next task.

Task 2: Create an Amazon RDS Database

Traditionally, creating a database can be a complex process requiring either a Database Administrator or a Systems Administrator. In the cloud, the task is made simple by using Amazon Relational Database Service (Amazon RDS).

In this task, you will create a MySQL database in your VPC. MySQL is a popular open-source relational database management system (RDBMS), so there are no software licensing fees.

⚠ **Windows Users:** Please use Chrome or Firefox as your web browser for this lab. The lab instructions are **not compatible with *Internet Explorer*** due to a difference in the Amazon RDS console.

8. On the **Services ⌄** menu, click **RDS**.

9. In the left navigation pane, click **Databases**.

10. Click **Create database**

11. On the **Select engine** page:

    - Select ◉ **MySQL**
    - Click **Next**

    Several use-cases will be presented, ranging from enterprise-class databases through to Dev/Test systems. You will also see mention of **Amazon Aurora**, which is a MySQL-compatible system that has been re-architected for the cloud. If your company uses large-scale MySQL or PostgreSQL databases, Amazon Aurora can provide enhanced performance.

12. On the **Choose use case** page:

    - Select ◉ **Dev/Test - MySQL**
    - Click **Next**

    You can now select a database configuration, including software version, instance class, storage and login settings. The **Multi-AZ deployment** option can automatically create a replica of the database in a second Availability Zone for High Availability. In this lab, however, you will use a single database instance.

13. On the **Specify DB details** page, configure:

    - **DB instance class:** *db.t2.micro* (at the top of the list)
    - **DB instance identifier:** `inventory-db`
    - **Master username:** `master`
    - **Master password:** `lab-password`
    - **Confirm password:** `lab-password`
    - Click **Next**

    You can next **Configure Advanced Settings**, including networking, security, backup and monitoring.

14. In the **Network & Security** section, configure:

    - **Virtual Private Cloud (VPC):** *Lab VPC*
    - **Subnet group:** *lab-db-subnet-group* (*This was created for you during lab setup*)

- **VPC security groups:**
  - Click ◉ **Choose existing VPC security groups**
  - Add **DB-SG** (which you created earlier)
  - Remove ✕ **default**

15. In the **Database options** section, configure:
    - **Database name:** `inventory`

    This is the logical name of the database that will be used by the application.

    💬 Feel free to look at the many other options displayed on the page, but leave them set to their default values. Options include automatic backups, the ability to export log files and automatic version upgrades. The ability to activate such features via tick-boxes shows the power of using a *fully-managed database solution* rather than having to install, backup and maintain the database yourself.

16. Click **Create database** (at the bottom of the page).

    You will receive a message that Your DB instance is being created.

    This will take a few minutes but **there is no need to wait**. Please continue with the next task.

Task 3: Launch an Application Server using Amazon EC2

You are now ready to launch an **Amazon EC2 instance** to run the application. As part of the launch, you will provide a *configuration script* that will automatically install an application. You will also associate the instance with the *Application security group* that you created earlier in the lab. This will permit HTTP (web) access from the Internet.

17. On the **Services ⌄** menu, click **EC2**.

18. Click **Launch Instance**

Step 1: Choose an Amazon Machine Image (AMI)

This step allows you to choose an AMI, which contains a copy of the disk volume that will be used to launch the instance.

Examine the list of AMIs that are displayed, showing many versions of Microsoft Windows and Linux. These disk images are regularly updated to incorporate security patches and software that helps you use AWS services. You can also create your own AMI that includes your own data and applications, or you can select pre-built commercial applications from the **AWS Marketplace**.

Your application will use *Amazon Linux 2*.

19. Beside the **Amazon Linux 2 AMI** in the top row, click **Select**

Step 2: Choose an Instance Type

This step allows you to choose an **Instance Type**, which determines the resources that will be allocated to your EC2 instance. Each Instance Type allocates a combination of virtual CPUs, memory, disk storage and network performance.

Instance Types are divided into **families** such as Compute-optimized, Memory-optimized and Storage-Optimized. The name of the Instance Type includes a family identifier, such as **t2** and **m4**. The number indicates the *generation* of the instance, so **m5** is newer than **m4**.

Your application will use a **t2.micro** Instance Type, which is a small instance that can burst above baseline performance when it is busy. It is ideal for development, testing and for applications that have bursty workloads.

20. Select ☑ **t2.micro**

21. Click **Next: Configure Instance Details**

Step 3: Configure Instance Details

Step 3: Configure Instance Details

This step allows you to configure instance details, such as the number of instances to launch and the network configuration. You can hover over the ℹ icons to view a description of each field.

You will launch the instance in a public subnet within the *Lab VPC* network.

22. Configure these settings:

    - **Network:** *Lab VPC*
    - **Subnet:** *Public Subnet 1* (Make sure it says *Public!*)
    - **IAM role:** *Inventory-App-Role*

    The *Inventory-App-Role* grants permissions to applications running on the instance to make requests to AWS services. The role is:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": "ssm:*",
            "Resource": "arn:aws:ssm:*:*:parameter/inventory-
app/*",
            "Effect": "Allow"
        }
```

23. Scroll down, then expand ▸ **Advanced Details**.

    A field for **User data** will appear.

    ℹ When you launch an instance, you can pass a configuration script via the *User data* field. The script can be used to perform configuration tasks and install software.

    Your instance is running Amazon Linux, so you will provide a *shell script* that will run when the instance starts.

24. Copy the following script and paste it into the **User data** field:

```
#!/bin/bash
# Install Apache Web Server and PHP
yum install -y httpd mysql
amazon-linux-extras install -y php7.2
# Download Lab files
wget https://us-west-2-tcprod.s3.amazonaws.com/courses/ILT-
TF-100-ARCHIT/v6.3.3/lab-2-webapp/scripts/inventory-app.zip
unzip inventory-app.zip -d /var/www/html/
```

Task 3: Launch an Application Server using Amazon EC2

Task 4: Test the Application

Task 3: Launch an Application Server using Amazon EC2

Task 4: Test the Application

Lab Complete

Task 2: Create an Amazon RDS Database

Task 3: Launch an Application Server using Amazon EC2

Task 4: Test the Application

using Amazon EC2

Task 4: Test the Application

Lab Complete

Task 2: Create an Amazon RDS Database

Task 3: Launch an Application Server using Amazon EC2

Task 4: Test the Application

Lab Complete

Task 1: Configure Security

Task 2: Create an Amazon RDS Database

Task 3: Launch an Application Server using Amazon EC2

Task 4: Test the Application

Lab Complete

Task 3: Launch an Application Server using Amazon EC2

using Amazon EC2

Task 4: Test the Application

Lab Complete

Task 2: Create an Amazon RDS Database

Task 3: Launch an Application Server using Amazon EC2

using Amazon EC2

Task 4: Test the Application

```
# Download and install the AWS SDK for PHP
wget https://github.com/aws/aws-sdk-
php/releases/download/3.62.3/aws.zip
unzip aws -d /var/www/html
# Turn on web server
chkconfig httpd on
service httpd start
```

The script will:

- Install an Apache web server (httpd) and the PHP language
- Download the Inventory application and the AWS SDK
- Activate the Web server and configure it to automatically start on boot

This type of script is an excellent way to configure a new instance without having to login and manually configure software. It also makes automation easy because a new instance can be launched and fully configured without requiring intervention from technical staff. You could similarly use Powershell scripts on Windows instances for configuration.

25. Click **Next: Add Storage**

Step 4: Add Storage

This step can be used to add additional Amazon Elastic Block Store (EBS) disk volumes and configure their size and performance.

both of which you define.

27. Click **Add Tag** then configure:

- **Key:** `Name`
- **Value:** `App Server`

This Name will appear on the instance in the EC2 management console.

28. Click **Next: Configure Security Group**

Step 6: Configure Security Group

You will associate the instance with the Application Security Group (*App-SG*) you created earlier in the lab.

29. Click ◉ **Select an existing security group**.

30. Select ☑ **App-SG**

    The inbound rules show that they will permit HTTP traffic from anywhere on the internet (0.0.0.0/0).

31. Click **Review and Launch**

    💬 If you receive a warning that *You will not be able to connect to the instance*, click **Continue**

Step 7: Review Instance Launch

This step displays a summary of the configuration for the instance you are about to launch.

32. Click **Launch**

    A **Select an existing key pair or create a new key pair** window will appear.

33. Select ☑ **I acknowledge that....**

34. Click **Launch Instances**

    Your instance will now be launched.

35. Click **View Instances**

    The instance will appear in a *pending* state, which means it is being launched. It will then change to *running*, which indicates that the instance has started booting. There will be a short time before you can access the instance, during which the configuration script will install and configure the application.

    Review the information displayed in the **Description** tab. It includes information about the instance type, security settings and network settings.

36. Wait for your instance to display the following a status of ● **running**.

💬 The instance will actually be up and running quite quickly, but the status check will take a couple of minutes longer.

Your instance has now started. The configuration script will automatically install and configure the application.

Task 4: Test the Application

You are now ready to test that the application is operational.

You can access the web application via the IP address of the EC2 instance.

37. Select ☑ **App Server**.

38. In the **Description** tab, copy the **IPv4 Public IP** to your clipboard.

    Tip: If you hover over the IP address, a copy ⧉ icon will appear. Click the icon to copy the displayed value.

39. Open a new web browser tab, paste the IP address into the address bar, then hit Enter.

    The web application should appear. Very little information is displayed because the application is not yet connected to the database.

40. Click ⚙ **Settings**.

    You can now configure the application to use the Amazon RDS DB instance you created earlier. You will first retrieve the **Database Endpoint** so that the application knows how to connect to a database.

41. Return to the **AWS Management Console** but do not close the application tab. (You will return to it soon.)

42. On the **Services ▾** menu, click **RDS**.

43. In the left navigation pane, click **Databases**.

44. Click inventory-db.

45. Scroll down to the **Connectivity** section and copy the **Endpoint** to your clipboard.

    It should look similar to: *inventory-db.crwxbgqad61a.rds.amazonaws.com*

46. Return to the browser tab with the Inventory application and enter:

    - **Endpoint:** Paste the endpoint you copied earlier
    - **Database:** `inventory`
    - **Username:** `master`
    - **Password:** `lab-password`
    - Click **Save**

    The application will now connect to the database, load some initial data and display information.

47. ✚ Add Inventory, ✎ edit and 🗑 delete inventory information using the web application.

    The inventory information is stored in the Amazon RDS MySQL database you created earlier in the lab. This means that any failure in the application server will *not* lose any data. It also means that multiple application servers can access the same data.

    👍 You have now successfully launched the application and connected it to the database!

    Optional: You can view the saved parameters in the **Systems Manager** console, under *Parameter Store*.

Lab Complete

Congratulations! You have completed the lab.

Click **End Lab** at the top of this page to clean up your lab environment

environment.