

# Guided Lab: Securing Applications by Using Amazon Cognito

---

## Lab overview and objectives

---

While building web applications, user authentication and authorization can be challenging. Amazon Cognito makes it convenient for developers to add sign-up, sign-in, and enhanced security functionality.

In this lab, you configure an Amazon Cognito user pool, which you use to manage users and their access to an existing web application. You also create an Amazon Cognito identity pool, which authorizes users when the application makes calls to the Amazon DynamoDB service.

After completing this lab, you should be able to do the following:

- Create an Amazon Cognito user pool.
- Add users to the user pool.
- Update the example application to use the user pool for authentication.
- Configure the Amazon Cognito identity pool.
- Update the example application to use the identity pool for authorization.

## Duration

---

This lab requires approximately **60 minutes** to complete.

## AWS service restrictions

---

In this lab environment, access to AWS services and service actions might be restricted to the ones that are needed to complete the lab instructions. You might encounter errors if you attempt to access other services or perform actions beyond those that this lab describes.

## Scenario

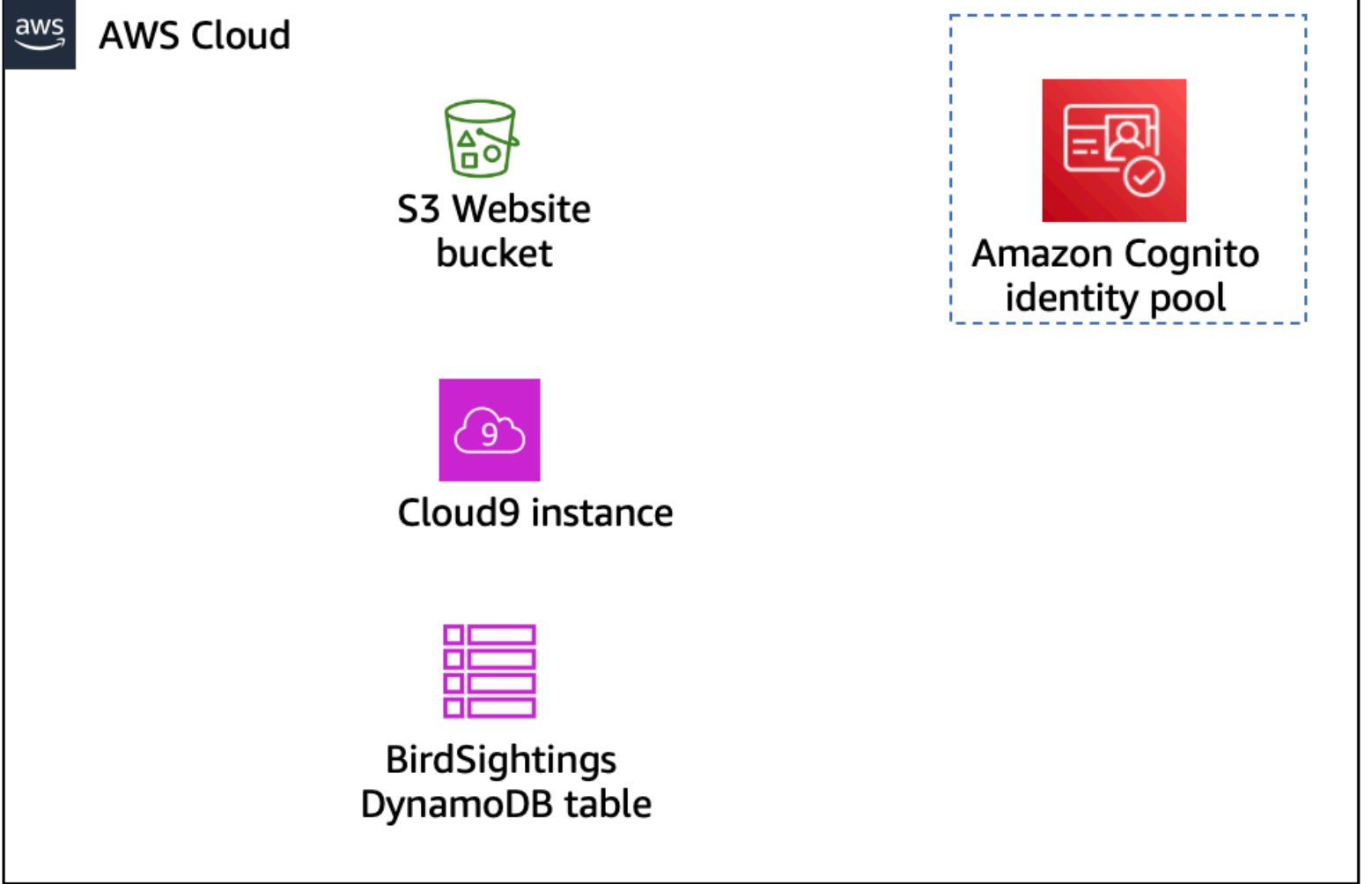
---

You have the Birds web application, which was built by using a NodeJs server running on an AWS Cloud9 instance and an Amazon Simple Storage Service (Amazon S3) bucket with static website hosting capability. The Birds application tracks students' bird sightings by using the following components:

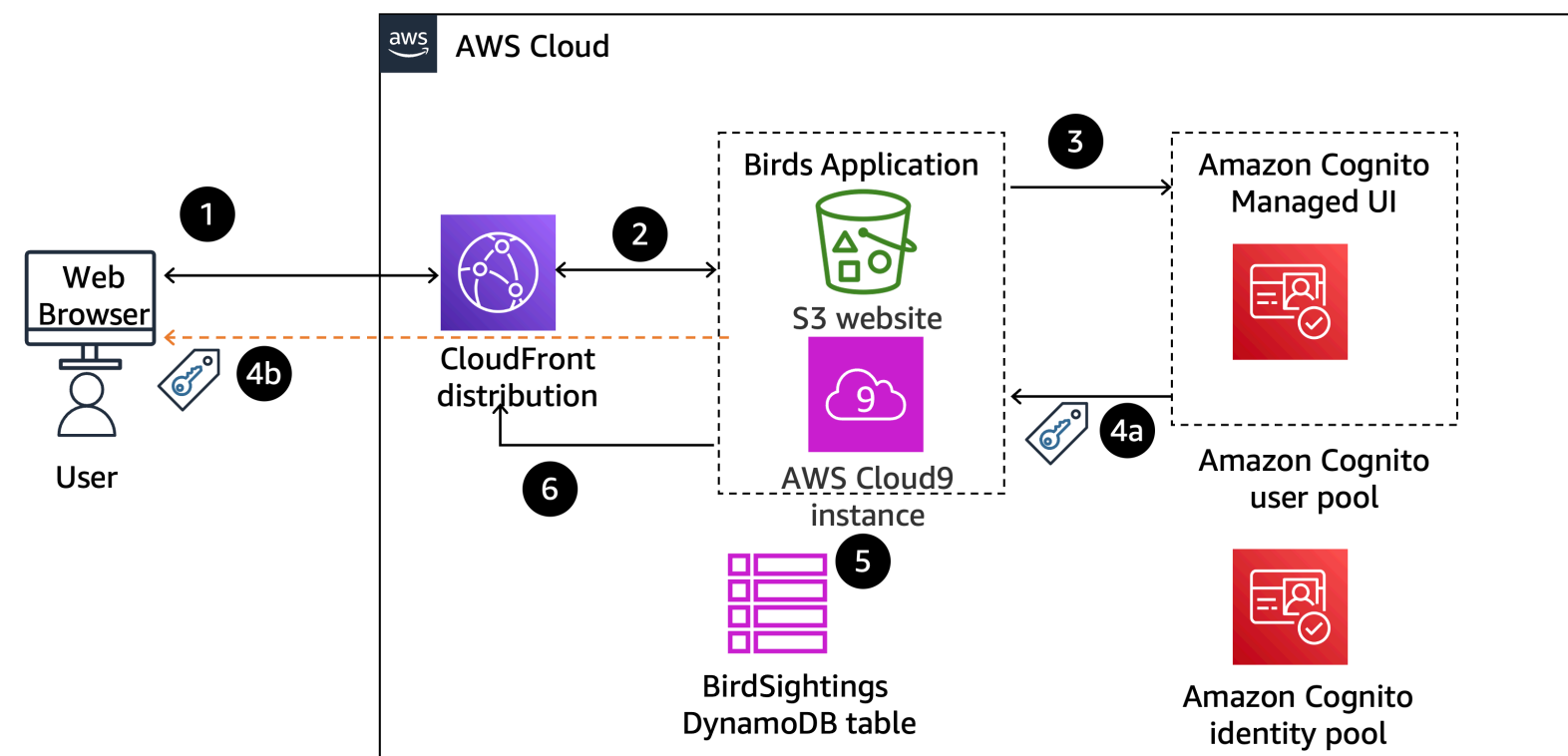
- A home page
- An educational page that teaches students about birds
- The following three protected pages, which students can access only if they have been authenticated:
  - A sightings page where students can view past bird sightings
  - A reporting page where students report new bird sightings
  - An administrator page where site administrators can perform additional operations

You need to add authentication and authorization to the application for the protected pages.

Starting architecture: You begin with the following architectural components. Initially, you use these components to install the application and get it running.

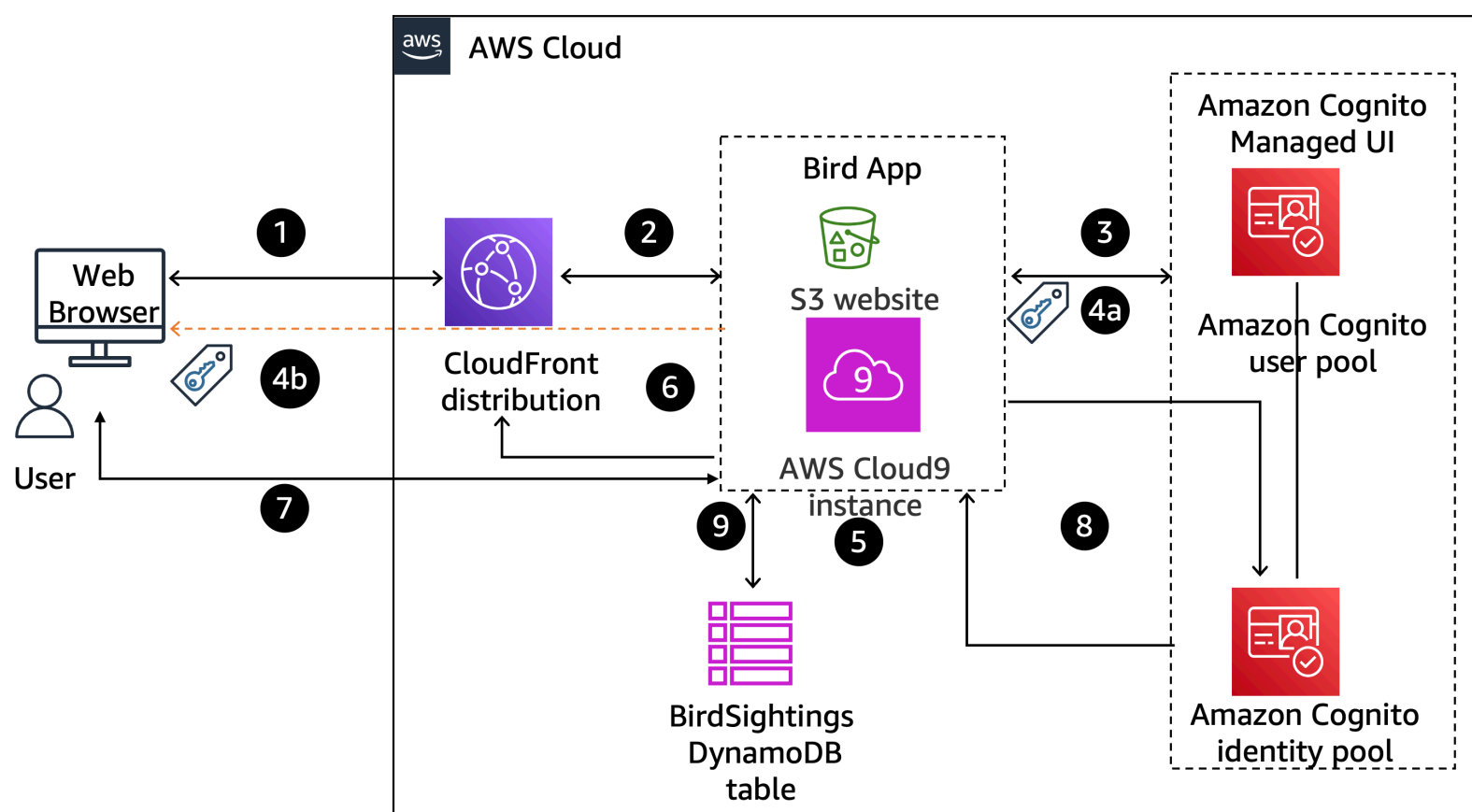


Intermediate architecture: You add the following components to the architecture to implement authentication by using an Amazon Cognito user pool for protected pages.



Step	Explanation
2	The request is routed to the NodeJs application server that is hosting the Birds application.
3	The application redirects the request to the Amazon Cognito managed UI.
4a	The user is authenticated by the Amazon Cognito user pool, and the access token is returned to the application.
4b	The Amazon Cognito SDK also stores the access token in browser's local storage for subsequent use, with the default expiration of 3,600 seconds.
5	The application validates the token and returns the protected page as requested.
6	The page is returned to the user's browser through the Amazon Cloudfront distribution.

Final architecture: Finally, you add additional authentication and authorization by using an Amazon Cognito identity pool to implement administrator access to the site.




Step	Explanation
1	The user requests access to the administrator page from the browser.
2	The request is routed to the NodeJs application server that is hosting the Birds application.
3	The application redirects the request to the Amazon Cognito managed UI.
4a	The user is authenticated by the Amazon Cognito user pool, and the access token is returned to the application.
4b	The Amazon Cognito SDK also stores the access token in browser's local storage for subsequent use, with the default expiration of 3,600 seconds.
5	The application validates the token and returns the administrator page as requested.

Step	Explanation
6	The page is returned to the user's browser through the Cloudfront distribution.
7	The user initiates a query to a DynamoDb table.
8	The application sends the token to the Amazon Cognito identity pool and receives temporary AWS credentials upon validation.
9	The application uses the received credentials to query the DynamoDB table and return data to the protected page. The page is returned to the user's browser through the Cloudfront distribution.

## Accessing the AWS Management Console

- At the top of these instructions, choose ▶ **Start Lab**.
  - The lab session starts.
  - A timer displays at the top of the page and shows the time remaining in the session.
 

💡 **Tip:** To refresh the session length at any time, choose ▶ **Start Lab** again before the timer reaches 0:00.
  - Before you continue, wait until the circle icon to the right of the AWS  link in the upper-left corner turns green. When the lab environment is ready, the **AWS Details** panel displays.
- To connect to the AWS Management Console, choose the **AWS** link in the upper-left corner, above the terminal window.
  - A new browser tab opens and connects you to the console.
 

💡 **Tip:** If a new browser tab does not open, a banner or icon is usually at the top of your browser with a message that your browser is preventing the site from opening pop-up windows. Choose the banner or icon, and then choose **Allow pop-ups**.
- Arrange the AWS Management Console tab so that it displays along side these instructions. Ideally, you will be able to see both browser tabs at the same time so that you can follow the lab steps.

## Task 1: Preparing the lab environment

Before you can work on this lab, you need to download files and run some scripts in the AWS Cloud9 integrated development environment (IDE) that was prepared for you.

- To connect to the AWS Cloud9 IDE, from the top of these instructions, choose **AWS Details**.
- Copy the value for **Cloud9url**, and paste it into a new browser tab to open the AWS Cloud9 IDE.
- In a text editor of your choice (outside the AWS Cloud9 IDE), copy and paste the following text. As you work through the lab, you use this file to store information that you need in later steps:

```
S3 bucket:
CloudFront distribution domain:
User pool ID:
App client ID:
Amazon Cognito domain prefix:
Identity pool ID:
```

- In the AWS Cloud9 IDE, in the terminal window in the bottom pane, at the **voclabs:~/environment \$** prompt, run the following commands to retrieve the code and install the Birds application that you use in this lab:

```
wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-100-RSNEWLABS-3-124627/311-lab-SF-
cognito/code.zip
unzip code.zip
cd resources
. ./setup.sh
```

**Note:** After pasting the commands into the terminal, press Enter to ensure that all of the commands run.

After the setup.sh script completes, you see the last four lines of output that are similar to the following:

```
...
# The S3 bucket name is:
"c42885a57145711365962t1w991727102856-s3bucket-1s4xxypclttq8"
# The CloudFront distribution domain is:
"drhx6krwefmhd.cloudfront.net"
```

8. From this output, copy and paste the following values into your text file. You need these values in later steps:

- Record the S3 bucket name. The value is similar to the following: **c1234567890abcdefghi-s3bucket-123456abcd3**
- Record the CloudFront distribution domain name. The value is similar to the following: **d123456acbdef.cloudfront.net**

Next, you update the web application code to point to the node server's API endpoint.

9. In the AWS Cloud 9 IDE, in the explorer window to the left of the terminal, expand the **website/scripts** folder.

10. Open the **config.js** file.

11. In the file, replace `<cloudfront-domain>` with the CloudFront distribution domain that you recorded into your text editor.

The updated line should be similar to the following:

```
CONFIG.BASE_NODE_SERVER_STR = "https://d123456acbdef.cloudfront.net";
```

12. To save the file, choose **File > Save**.

13. To close the file, on the tab with the file, choose **X**.

Because you are using an S3 bucket to host the website, you need to upload the application files to the S3 bucket:

14. In the following command, replace `<s3-bucket>` with the name of the S3 bucket that you recorded into your text editor, and run the updated command in the AWS Cloud9 terminal:

```
cd /home/ec2-user/environment
aws s3 cp website s3://<s3-bucket>/ --recursive --cache-control "max-age=0"
```

Because you also use the NodeJs server as part of application, you need to start the node server.

15. To start the node server, run the following commands:

```
cd /home/ec2-user/environment/node_server
npm start
```

After the node server starts, you see the following output in the terminal:

```
> start
> REGION_STR=us-west-2 node index.js

Live on port: 8080
```

**Note:** Leave the server running in this terminal window.

As part of the setup script that you ran earlier, the CloudFront distribution is being created. This distribution is used to securely deliver the contents hosted on the S3 bucket.

Next, you check the status of the CloudFront distribution.

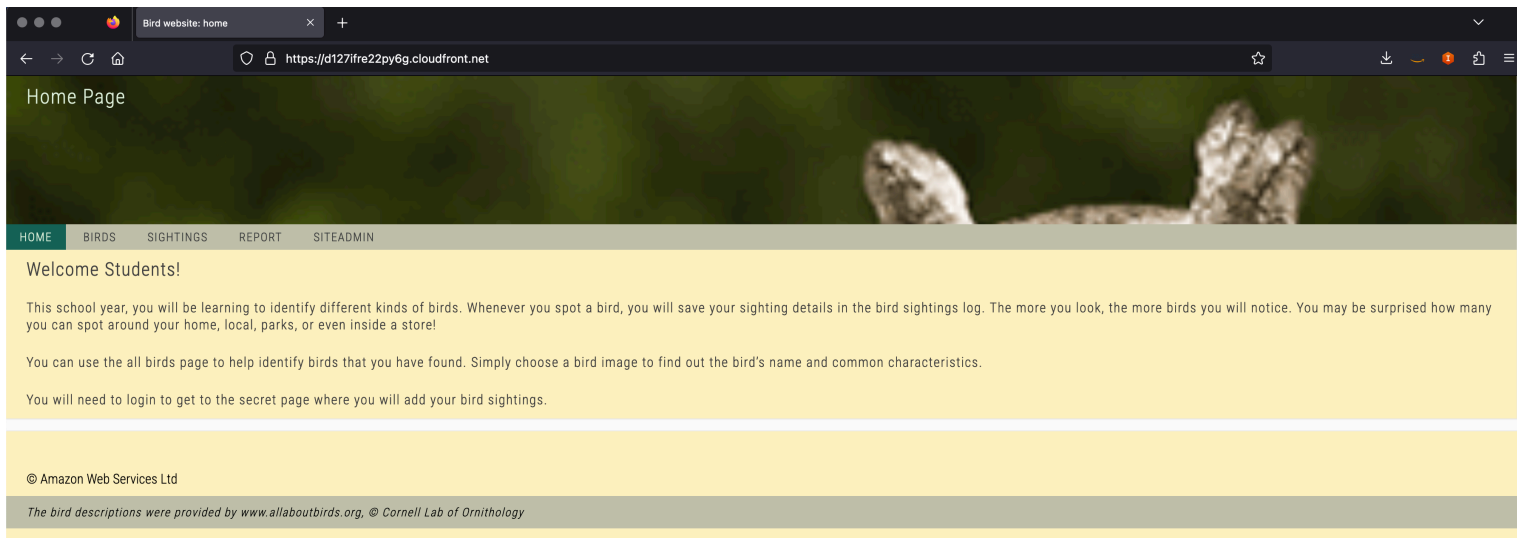
16. On the AWS Management Console, in the search box, enter and choose **CloudFront** to open the CloudFront console.
17. In the **Distributions** list, locate the **Status** column of the distribution, and wait or proceed to the next task based on the status:
  - If the status is *Deploying*, wait until the status is *Enabled*, and then proceed to the next task.
  - If the status is *Enabled*, continue to the next task.

## Task 2: Reviewing the Birds website

In this task, you explore the Birds web application to understand how it behaves before you enable user authentication.

18. In a new browser tab, enter the CloudFront distribution domain that you copied into your text file.

The Birds web application opens. The home page includes a welcome message to students.



19. Navigate to the **BIRDS** page to see pictures of birds.
  - For additional information, choose one of the bird pictures.

**Note:** These pages with bird descriptions are not protected. Anyone can get to this content.
20. To return to the previous page, choose **BACK**.

**Note:** You might need to scroll down to locate the **BACK** button.
21. Choose **SIGHTINGS**.

**Note:** This page requires users to log in before they can see the content.
22. To try the login functionality, choose **LOGIN**.

A message indicates that you do not have access. This message appears because the identity provider, an Amazon Cognito user pool, still needs to be set up for the application.

Choose **Dismiss**.
23. Close the browser tab where the Birds web application is running. Keep the AWS Cloud9 IDE browser tab open, and let the node server run.

## Task 3: Configuring the Amazon Cognito user pool

In this task, you create an Amazon Cognito user pool, create users, and update the application to use the user pool.

## Task 3.1: Creating a user pool

In this task, you create an Amazon Cognito user pool, which is used as the identity provider to create users and manage user passwords. The Birds web application uses the tokens generated by the Amazon Cognito user pool to help ensure that users have authenticated and have a valid session before they can access a protected page or perform a protected action. Tokens are also used to authenticate administrators and provide access to the site administrator page.

24. On the AWS Management Console, in the search box, enter and choose `cognito` to open the Amazon Cognito console.

25. In the navigation pane on the left, choose **User pools**.

26. Choose **Create user pool** button.

27. On the **Set up resources for your application** page, configure the following options:

- For **Application type**, ensure that **Traditional web application** is already selected.
- For **Name your application:** `bird_app_client`
- Under the **Configure options**, choose **Username**
- For **Required attributes for sign-up**, choose **email** from the dropdown.
- For **Add a return URL - optional**, enter `https://<cloudfront-domain>/callback.html`, and replace *<cloudfront-domain>* with the CloudFront distribution domain from your text editor.

**Note:** The updated URL should be similar to the following: <https://d123456acbdef.cloudfront.net/callback.html>

- Choose **Create user directory** button.
- On the next page, under **Check out your sign-in page**, choose **View login page**.
  - Notice the **Sign-in** page which you will configure later, close the browser tab.
- Go back to the **User pools** and choose the pool you created.
  - User pool Name should be similar to *User pool - zzzzzz*
- On the **Overview:** page, you notice the pool details and other configuraiton information.
- Choose **Rename** button and under **User pool name** enter `bird_app`, choose **Save changes**.
- Copy the value for **User pool ID** to the editor.
- From the bottom **Recommendations** pane, choose `bird_app_client` link.
- Go to **Login pages**
  - On the **Managed login pages configuration** choose **edit**.
  - For **OAuth 2.0 grant types**, ensure that **Authorization code grant** is chosen.
  - From the dropdown list, choose **Implicit grant**.
  - For **OpenID Connect scopes**, ensure that **Email** and **OpenID** are chosen, and clear **Phone**.
  - Choose **Save changes** button.
- On the **App client: bird\_app** page, copy the value for **Client ID** to the editor.
- Choose **Edit**.
- For **Authentication flows**, from the dropdown list, choose **ALLOW\_USER\_PASSWORD\_AUTH**, and clear **ALLOW\_USER\_SRP\_AUTH**.
- Choose **Save changes**.
- From the pane on the left, choose **Domain**
- From the **Domain** copy the part of the URL after `//` and before `.auth.us-west-2.amazoncognito.com`, paste it into your editor against *Amazon Cognito domain prefix*.

Note: The Cognito domain should look similar to *us-west-2ozkgdmcoh*

Next you create and add users to the user pool.

## Task 3.2: Adding users to the user pool

28. From the navigation pane on the left, choose **Overview**.

29. Under **User management** choose **Users**

30. Choose **Create user** button.

31. Create a user with the following details:

- For **User name**, enter `testuser`.
- For **Password**, enter `Lab-password1$`.

32. Choose **Create user** button.

33. Create another user with the following details:

- For **User name**, enter `admin`.
- For **Password**, enter `Admin123$`.

34. Choose **Create user** button.

**Note:** This user has administrator permissions.

35. From the navigation pane, under **User management** choose **Groups**.

36. Choose **Create group** button.

37. On the **Create group** page for **Group name**, enter `Administrators`.

38. Choose **Create group** button.

39. Choose the **Administrators** group that you created.

40. Choose **Add user to group** button.

41. From the list of users, select **admin**.

42. Choose **Add** button.

**Note:** You have added the **admin** user to the **Administrators** group.

43. To return to the earlier menu, choose the **Overview** from the navigation pane.

## Task 4: Updating the application to use the user pool for authentication

So far, you have created the Amazon Cognito user pool, configured it with information about the web application, and created a user. However, the Birds application isn't set up with authentication yet because the application doesn't know about Amazon Cognito.

In this task, you update the application to provide the information it requires to interact with Amazon Cognito. This information includes the user pool ID, application client ID, and Amazon Cognito domain prefix.

44. Return to the browser tab with the AWS Cloud9 IDE.

45. In the terminal, to stop the node server, press Ctrl+C.

Next, you update the config.js file to add your Amazon Cognito user pool information to the website code.

46. In the explorer to the left of the terminal, expand the **website/scripts** folder.

47. Open the **config.js** file.

48. In the file, to uncomment the following lines of code, remove the two forward slashes (`//`) from the beginning of each line.



```
//CONFIG.COGNITO_DOMAIN_STR = "<cognito-domain>";
//CONFIG.COGNITO_USER_POOL_ID_STR = "<cognito-user-pool-id>";
//CONFIG.COGNITO_USER_POOL_CLIENT_ID_STR = "<cognito-app-client-id>";
//CONFIG.CLOUDFRONT_DISTRO_STR = "<cloudfront-distribution>";
```

49. In this code, replace each placeholder with the following values that you saved into your text editor:

**Note:** Update the values for only the following parameters. Replace the entire string, including the brackets (<>).

- *<cognito-domain>*: Use the value for **Amazon Cognito domain prefix**.
- *<cognito-user-pool-id>*: Use the value for **User pool ID**.
- *<cognito-app-client-id>*: Use the value for **App client ID**.
- *<cloudfront-distribution>*: In the value for **CloudFront distribution domain**, use only the prefix before **.cloudfront.net** (for example, **d123456acbdef**).

The updated code should be similar to the following:

```
CONFIG.COGNITO_DOMAIN_STR = "abc-10-12-2021";
CONFIG.COGNITO_USER_POOL_ID_STR = "us-west-2_AAAA1111";
CONFIG.COGNITO_USER_POOL_CLIENT_ID_STR = "1a1a1a12b2b2b2b3c3c3c3c";
CONFIG.CLOUDFRONT_DISTRO_STR = "d123456acbdef";
```

50. To save the file, choose **File > Save**.

Next, you push the updated website code to the S3 bucket.

51. In the AWS Cloud9 IDE terminal, enter the following command, and replace *<s3-bucket>* with the name of the S3 bucket. Run your revised command.

```
cd /home/ec2-user/environment
aws s3 cp website s3://<s3-bucket>/ --recursive --cache-control "max-age=0"
```

52. To update the node server files, run the following commands:

**Note:** These commands update the JavaScript packages required for the application to use the Amazon Cognito user pool, along with other configuration changes made earlier.

```
cd /home/ec2-user/environment/node_server
cp package2.json package.json
cp libs/mw2.js libs/mw.js
```

Next, you update the package.json file to add the Amazon Cognito user pool information to the node server.

53. In the explorer window to the left of the terminal, expand the **node\_server** folder.

54. Open the **package.json** file.

55. Replace *<cognito\_user\_pool\_id>* with the **User pool ID** from your text file.

The updated line should be similar to the following:

```
"start": "REGION_STR=us-west-2 USER_POOL_ID_STR=us-west-2_AAAA1111 node index.js"
```

56.. To save the file, choose **File > Save**.

## Task 5: Testing the user pool integration with the application

In this task, you test the updated application. First, you restart the node server so that it uses the updated configuration.

57. To start the node server, run the following command at the prompt:

```
npm start
```

58. Return to the browser tab with the Birds application, and refresh the page.

59. Choose **SIGHTINGS**.

This time, the list of bird sightings from the students in the class is not displayed.

The bird sightings are not displayed because the application is now integrated with Amazon Cognito and requires authentication!.

Note: If you get an error related to *jwt*, start a new browser tab or window.

60. Choose **LOGIN**.

61. On the prompt, provide the credentials for **testuser**.

**Note:** You may be prompted to change password.

62. After successfully logging in, the birds listing is displayed.

**Note:** If you are redirected to the **Home** page, navigate to the **SIGHTINGS** page again.

This test proves that only a user who has been authenticated by the Amazon Cognito user pool can access the protected pages of the application. The user pool centrally stores your application usernames and passwords, which makes your users and their authentication information more secure and convenient to manage.

63. Choose **SITEADMIN**.

You see a message stating that you need to log in as an administrator to access the administration page.

If you see the message *You need Admin credentials to see Admin page*, choose **DISMISS**.

64. On the **SITEADMIN** page, choose **ADMIN LOGIN**.

65. On the login prompt, choose **Sign in as a different user?**

66. Use the **admin** login credentials to log in.

**Note:** You may be prompted to change password.

After successfully logging in, navigate to **SITEADMIN** page. The message *Admin page under construction* is displayed.

This test proves that the Amazon Cognito user pool can manage role-based access control to your application and provide secure access to the protected pages of the application based on the role.

## Task 6: Configuring the identity pool

The Amazon Cognito identity pool was created for you when you launched the lab environment. In this task, you configure the Amazon Cognito identity pool to work with the Birds application.

67. On the Amazon Cognito console, in the navigation pane on the left, choose **Identity pools**.

68. Choose the **bird\_app\_id\_pool** link.

69. Copy and paste the **Identity pool ID** into your text editor as the **Identity pool ID** for use later.

70. From the lower pane, choose the **User access** tab.

71. Choose **Add identity provider**.

72. Choose **Amazon Cognito user pool**.

73. For **User pool ID**, choose the user pool with the name **bird\_app**.

74. For **App client ID**, choose the application with the name **bird\_app\_client**.

75. In the **Role settings** section, notice that **Default authenticated role** is displayed.

76. Choose **Save changes** button.

77. Review the **Authenticated access** section.

Notice that the **Authenticated role** has been configured to use the default role, which is assigned to users when they successfully log in. You could set up additional rules to assign different AWS Identity and Access Management (IAM) roles to different users. For this phase of application development, you only keep one role assigned to the users.

## Task 7: Updating the application to use the identity pool for authorization

As with the user pool, the application needs to be updated so it can interact with the identity pool. In this task, you make the necessary updates to the Birds application.

First, you update the Birds web application.

78. Return to the browser tab where the AWS Cloud9 IDE is open.
79. In the terminal window where the node server is running, press Ctrl+C to stop the Nodejs server:

Next, you update the config.js file.

80. In the Explorer window, expand the **website/scripts** folder.
81. Open the **config.js** file.
82. Remove the two forward slashes (**//**) from the beginning of the last line of code to uncomment the code, and replace `<cognito-identity-pool-id>` with the identity pool ID that you saved earlier.
83. To save the file, choose **File > Save**.

Next, you update the auth.js file.

84. From the **website/scripts** folder, open the **auth.js** file.
85. In the file, replace `<cognito-user-pool-id>` with the user pool ID. The placeholder is on or around line 91 in the file.

**Important:** Ensure that you use the user pool ID here and not the identity pool ID.

```
AWS.config.credentials = new AWS.CognitoIdentityCredentials({
  IdentityPoolId : CONFIG.COGNITO_IDENTITY_POOL_ID_STR,
  Logins : {
    "cognito-idp.us-west-2.amazonaws.com/<cognito-user-pool-id>": token_str_or_null
  }
});
```

The updated code is similar to the following:

```
AWS.config.credentials = new AWS.CognitoIdentityCredentials({
  IdentityPoolId : CONFIG.COGNITO_IDENTITY_POOL_ID_STR,
  Logins : {
    "cognito-idp.us-west-2.amazonaws.com/us-west-2_AAAA1111": token_str_or_null
  }
});
```

**Note:** This section of code uses the COGNITO\_IDENTITY\_POOL\_ID\_STR variable, which you set in the config.js file. The code uses this identifier to request credentials from the identity pool. Notice that this code block also passes the user pool ID and token\_str\_or\_null, which holds the authentication token. The identity pool uses this information to verify the user in the user pool. If the user and token pass the validation, the identity pool sends AWS credentials back to the application.

86. To save the file, choose **File > Save**.

Next, you push the updated website code to the S3 bucket.

87. In the AWS Cloud9 IDE terminal, enter the following command and replace `<s3-bucket>` with the name of the S3 bucket. Run your revised command.

```
cd /home/ec2-user/environment
aws s3 cp website s3://<s3-bucket>/ --recursive --cache-control "max-age=0"
```

88. Ensure that the node server is still running. If it is not, run the following commands to start it again:

```
cd /home/ec2-user/environment/node_server
npm start
```

## Task 8: Testing the identity pool integration with the application

In this task, you test the updated Birds application to ensure that you can access temporary AWS credentials. With these temporary credentials, you are able to access AWS services based on the roles that were defined when you set up the identity pool. Remember that your identity pool is configured to associate authenticated users with an IAM role that allows access to a DynamoDB table.

89. Return to the browser tab where the Birds application is open.

90. Choose **HOME**, and refresh the page to ensure that your browser is using the updated code.

91. Choose **REPORT**.

92. Choose **LOGIN**, and log in with the **admin** credentials that you created in the user pool.

**Note:** You may be already logged-in as **admin** user from the earlier steps.

93. If you were redirected to **HOME** page, choose **REPORT** again.

94. To verify that you now have access to the temporary AWS credentials that you can use to interact with a DynamoDB table, choose **VALIDATE MY TEMPORARY AWS CREDENTIALS**.

The application now uses the identity pool to generate temporary credentials and uses the same credentials to access the **BirdSightings** DynamoDB table. After successfully connecting to the database table, the application attempts to count the number of rows (0) and returns the following message: *Your temporary AWS credentials have been configured.. Connecting to DynamoDB Table..BirdSightings Your Dynamodb Table has 0 rows..*

This test verifies that you have correctly configured the Amazon Cognito identity pool and the application. Users who are logged in are able to access temporary credentials to communicate with the DynamoDB database.

**Note:** You can access the DynamoDB table from the AWS Management Console and verify that the table has 0 rows..

## Conclusion

Congratulations! You now have successfully done the following:

- Created an Amazon Cognito user pool
- Added users to the user pool
- Updated the example application to use the user pool for authentication
- Configured the Amazon Cognito identity pool
- Updated the example application to use the identity pool for authorization

## Submitting your work

95. To record your progress, choose **Submit** at the top of these instructions.

96. When prompted, choose **Yes**.


After a couple of minutes, the **Grades** panel appears and shows you how many points you earned for each task. If the results don't display after a couple of minutes, choose **Grades** at the top of these instructions.

**💡 Tip:** You can submit your work multiple times. After you change your work, choose **Submit** again. Your last submission is recorded for this lab.

97. To find detailed feedback about your work, choose **Submission Report**.

# Lab complete

---

 Congratulations! You have completed the lab.

98. At the top of this page, choose  **End Lab**, and then choose Yes to confirm that you want to end the lab.

The following message appears: **Ended AWS Lab Successfully**

© 2023, Amazon Web Services, Inc. and its affiliates. All rights reserved. This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited.