**Deploy Full Stack Todo Application**

The purpose of this document is to walk through the deployment of Brad Massengale’s Todo application using existing Terraform scripts.

**Prerequisites**

* [AWS Account](https://aws.amazon.com/console/)
* [Terraform](https://learn.hashicorp.com/tutorials/terraform/install-cli?in=terraform/aws-get-started)
* [DBeaver](https://dbeaver.io/download/) or some other database viewer
* [Code](https://github.com/bmassengale/todo) downloaded locally
* Text editor
* Terminal/Console/Powershell access

**Initialize Terraform Environments**

1. Open your terminal to the root of the project

A screen shot of a monitor

Description automatically generated

1. Navigate to *terraform/Phase1* and run t*erraform init*
   1. If successful, the message “Terraform has been successfully initialized!” will appear
2. Open BaseCodeDependencies.tf in your favorite editor
3. Within the “provider” text block starting on line 10, replace the “profile” tag with an “access\_key” and a “secret\_key” tag
   1. **Note:** This is not secure, but is done here for simplicity. Do not save these tags in source control
   2. This is a reference to the keys associated with an AWS IAM user
   3. The resulting code will look similar to the following:

*A picture containing graphical user interface, text

Description automatically generated*

**Create Core Resources (Phase 1)**

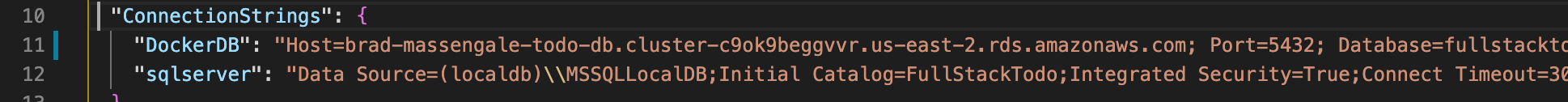
1. Within the same terraform/Phase1 directory, run terraform apply
2. Type “yes” when prompted
3. Wait for terraform to notify you of resource creation
   1. This can take up to 30 minutes
4. Make note of the terminal outputs
   1. Cloudfront\_URL
   2. DB\_Endpoint
   3. DB\_Port
   4. DB\_Username
   5. DB\_Password

A picture containing text

Description automatically generated

**Update and Upload API Code**

1. Email [bradamassengale@gmail.com](mailto:bradamassengale@gmail.com) and ask to have the value of Cloudfront\_URL whitelisted in his Auth0 account
   1. **This is VERY important for the application to work in the end**
2. From the root of the code, navigate to API/TodoApi.Api/ and open appsettings.Development.json
3. Rename the file appsettings.json
4. Update the Host next to the “DockerDB” key to be the DB\_endpoint



1. Save and close the file
2. In the terminal, navigate to the API directory and run the following command:
   1. dotnet publish -c Release -o Production TodoApi.sln
   2. After this command is run, a directory called Production will be created
3. Navigate into the Production directory
4. Compress all of the contained files into a zip file named Production.zip
5. Open a web browser and log into the AWS Console and ensure that the us-east-2 region is selected
6. Navigate to the S3 page
7. Open the S3 bucket called brads-magnificent-backend-api
8. Upload Production.zip

**Initialize Phase 2 and Create API Resource**

1. From the root of the code base, navigate to *terraform/Phas21* and run t*erraform init*
   1. If successful, the message “Terraform has been successfully initialized!” will appear
2. Open API.tf in your favorite editor
3. Within the “provider” text block starting on line 12, replace the “profile” tag with an “access\_key” and a “secret\_key” tag
4. Run *terraform apply*
5. Type “yes” when prompted
6. Wait for terraform to notify you of resource creation
   1. This can take several minutes
7. Make note of the terminal outputs
   1. API\_URL

**Update and Upload Frontend Code**

1. From the root of the code, navigate to client/src
2. Open index.js
3. Update the redirectUri value to the CloudFront\_URL
   1. Be sure to add https:// at the start of the new value

Text

Description automatically generated

1. Open client/src/Components/AppContainer/AppContainer.js
2. Replace all 4 occurrences of http://localhost:8080 with the API\_URL

A screen shot of a television

Description automatically generated

1. Save the file after replacing the text
2. Navigate to the “client” directory in the terminal
3. Run *npm install*
4. Run *npm run build*
5. Open a web browser and log into the AWS Console and ensure that the us-east-2 region is selected
6. Navigate to the S3 page
7. Open the S3 bucket called brads-magnificent-todo-application
8. Upload all contents from within the client/build directory
   1. Be sure that the files and folders have both copied over. This might take to “Uploads”
9. Within the AWS Console, open the CloudFront Service
10. Click the ID of the distribution made by terraform
    1. Hint: Look for a comment similar to the below

Graphical user interface

Description automatically generated

1. Click the “Invalidations” tab
2. Click “Create Invalidation”
3. In the textbox, type “/\*” without the quotes

Graphical user interface, application

Description automatically generated

1. Click Invalidate

Graphical user interface, text, application

Description automatically generated

1. Wait for the Status to change to “Completed”

**Prepare the Database**

1. Open a SQL Database viewer of your choice
2. Create a connection to a new database
3. If prompted, tell the software that the database is a PostgreSQL instance
4. Fill in the database information from the Phase 1 outputs

A screenshot of a cell phone

Description automatically generated

1. After connecting, open a SQL editor within the application
2. Type the following query to create the necessary table for the application

CREATE TABLE public.Todos (

TodoId SERIAL PRIMARY KEY,

Title varchar(400),

IsComplete BOOLEAN,

Something varchar(400),

Username varchar(400)

);

1. Execute the command
2. Ensure that the table has been created

Graphical user interface, text, application

Description automatically generated

**Enjoy the Application**

1. Open a web browser
2. in the search bar, type the Cloudfront\_URL
3. Enjoy!