**Setting Up Zappa on Your Local System**

**Prerequisites**

1. **AWS Account**: Ensure you have an AWS account. If not, [sign up for AWS](https://aws.amazon.com/free/).
2. **Python and pip**: Make sure Python (3.8 or higher) and pip are installed on your system.
3. **Git**: Ensure Git is installed to clone the repository.
4. **AWS CLI**: Install the AWS CLI and configure it with your AWS credentials.

**Step 1: Clone the Repository**

First, clone the repository and switch to the feature/zappa branch.

git clone <your-repository-url>

cd <your-repository-directory>

git checkout feature/zappa

**Step 2: Install Zappa**

1. **Create a Virtual Environment**: It’s a good practice to use a virtual environment to manage dependencies.

python -m venv venv

source venv/bin/activate # For Windows: venv\Scripts\activate

1. **Install Zappa and Other Dependencies**: Install Zappa and any other required packages using pip.

pip install zappa

pip install -r requirements.txt

**Step 3: Configure AWS CLI**

1. **Install AWS CLI**: Download and install the AWS CLI from the [official AWS CLI installation guide](https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2.html).
2. **Configure AWS CLI**: Configure the AWS CLI with your credentials.

aws configure

You will be prompted to enter your AWS Access Key ID, Secret Access Key, region, and output format. Make sure to use the same region specified in your zappa\_settings.json file.

**Step 4: Check zappa\_settings.json**

Open the zappa\_settings.json file and identify your collaborator environment. The environments are defined as keys in the JSON file, such as dev, collaborator1, collaborator2, etc.

Example snippet from zappa\_settings.json:

{

"dev": {

"aws\_region": "us-east-1",

"django\_settings": "myproject.settings",

"profile\_name": "default",

"project\_name": "myproject",

"runtime": "python3.8",

"s3\_bucket": "my-zappa-bucket",

"environment\_variables": {

"DB\_NAME": "your\_db\_name",

"DB\_USER": "your\_db\_user",

"DB\_PASSWORD": "your\_db\_password",

"DB\_HOST": "your\_db\_host",

"DB\_PORT": "5432"

}

},

"collaborator1": {

"aws\_region": "us-east-1",

"django\_settings": "myproject.settings",

"profile\_name": "default",

"project\_name": "myproject-collaborator1",

"runtime": "python3.8",

"s3\_bucket": "my-zappa-bucket-collaborator1",

"environment\_variables": {

"DB\_NAME": "your\_db\_name",

"DB\_USER": "your\_db\_user",

"DB\_PASSWORD": "your\_db\_password",

"DB\_HOST": "your\_db\_host",

"DB\_PORT": "5432"

}

}

}

**Note – the zappa\_settings.json file will be available in the repository with all needed data.**

**Step 5: Deploying with Zappa**

1. **Deploy to Your Environment**: Use the zappa deploy command with your environment name to deploy the application.

zappa deploy <your-environment-name>

For example, if your environment is collaborator1, run:

zappa deploy collaborator1

1. **Update Deployment**: If you make changes and need to update the deployment, use the zappa update command.

zappa update <your-environment-name>

**Step 6: Managing Migrations**

1. **Create Migrations**: If you need to create migrations, run:

zappa manage <your-environment-name> "makemigrations"

1. **Apply Migrations**: Apply the migrations to the database.

zappa manage <your-environment-name> "migrate"

**Additional Zappa Commands**

* **Check Status**: To check the status of your deployment, use:

zappa status <your-environment-name>

* **Tail Logs**: To tail the logs of your Lambda function:

zappa tail <your-environment-name>

* **Rollback Deployment**: To rollback to a previous deployment:

zappa rollback <your-environment-name>

**Troubleshooting Tips**

* **AWS Permissions**: Ensure that your AWS IAM user has the necessary permissions to deploy Lambda functions, manage S3 buckets, and configure API Gateway.
* **Environment Variables**: Double-check the environment variables in zappa\_settings.json to ensure they are correct.
* **Logs**: Use the zappa tail command to view logs and debug any issues.