Terraform Module Breakdown

1. VPC Module (`modules/vpc/`)

- **`main.tf`**: Defines the AWS VPC, subnets, and networking components.
- **`variables.tf`**: Contains input variables such as CIDR blocks and tags.
- **`outputs.tf`**: Exports important values like `vpc_id` and `subnet_ids`.

2. EC2 Module (`modules/ec2/`)

- **`main.tf`**: Defines the EC2 instance, security groups, and related resources.
- **`variables.tf`**: Contains input variables such as instance type and AMI ID.
- **`outputs.tf`**: Exports values like `ec2_public_ip`.

3. RDS Module (`modules/rds/`)

- **`main.tf`**: Defines the AWS RDS instance and security settings.
- **`variables.tf`**: Contains input variables like database username and password.
- **`outputs.tf`**: Exports values such as `rds_endpoint`.

Backend Configuration (`backend/`)

1. State Storage and Locking

- **S3**: Used to store the Terraform state file centrally.
- **DynamoDB**: Used for state locking to prevent conflicts.

2. Backend Files

- **`main.tf`**: Configures S3 backend and DynamoDB state locking.
- **`providers.tf`**: Defines the AWS provider and authentication methods.
- **`terraform.tf`**: Contains the root Terraform configuration settings.

Root Files

1. `main.tf`

- Calls the VPC, EC2, and RDS modules.
- Passes necessary input variables to modules.

2. `variables.tf`

- Stores globally used variables, such as region, instance type, and tags.

3. `outputs.tf`

- Defines output values for resources, including the EC2 public IP and RDS endpoint.

4. `providers.tf`

- Configures the AWS provider for Terraform.

5. `terraform.tf`

- Contains root-level Terraform settings and backend configurations.

Deployment Steps

Initialize Terraform:

```
terraform init
```

Plan Infrastructure Changes:

```
terraform plan
```

Apply Configuration:

```
terraform apply
```

Destroy Resources:

```
terraform destroy
```

This Terraform project ensures modular and scalable infrastructure management using AWS. The backend configuration leverages **S3** for centralized state storage and **DynamoDB** for state locking, providing security and reliability in Terraform operations.