========================

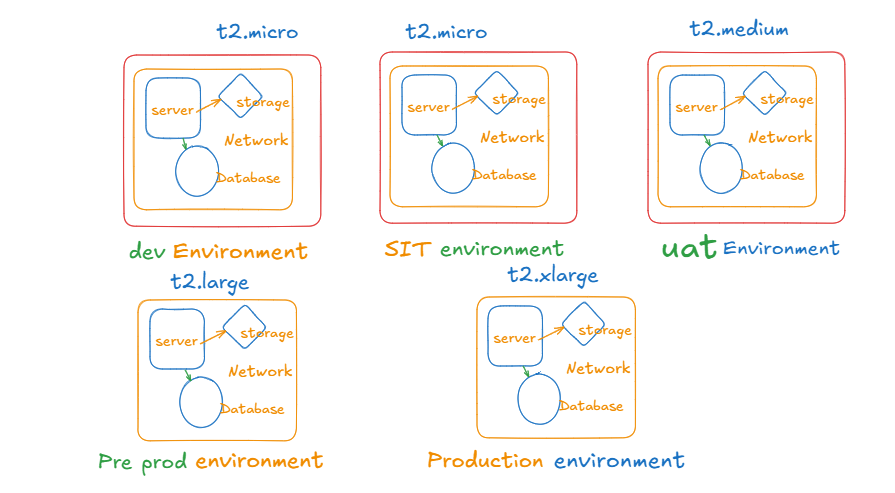
Environment of a project

========================

Environment means the platform that is required to run our application

Ex: Servers, Database, Storage, Network....

One project contains multiple env



E.g.: DEV, SIT or QA, UAT, PILOT, PROD env

Dev Env : Developers will use it for code integration testing

SIT / QA Env : Testers will use it for System Integration Testing

UAT Env: Client will use it for Acceptance testing.

Pilot Env : Pre-Prod testing and Performance testing.

Prod Env : Live Environment - live customers are accessing our application using this environment

Non-PROD Env (DEV, SIT, UAT and PILOT)

PROD Env (Very important & critical)

Note: In real-time from environment-to-environment infrastructure resources configuration might be different

Ex : For Non-PROD : t2.micro or t2.medium instances required

For PROD : t2.xlarge instances required

In order to achieve this requirement, we will maintain environment specific input variable file

inputs-dev.tf - input variables for dev env

inputs-sit.tf - input variables for sit env

inputs-uat.tf - input variables for uat env

inputs-pilot.tf - input variables for pre prod env

inputs-prod.tf - input variables for production env

When we are executing terraform apply command, we can pass inputs variable

create infrastructure for DEV env

==================================

# terraform apply --var-file=inputs-dev.tf

create infrastructure for sit env

==================================

# terraform apply --var-file=inputs-sit.tf

create infrastructure for UAT env

==================================

# terraform apply --var-file=inputs-uat.tf

create infrastructure for pilot env

==================================

# terraform apply --var-file=inputs-pilot.tf

create infrastructure for prod env

==================================

# terraform apply --var-file=inputs-prod.tf

With this approach we can achieve loosely coupling and we can achieve script re-usability.