RoostGPT Installer

- RoostGPT CLI Installation Guide - EC2 Linux
- EC2 Requirements for Docker based Roost
- Roost on Windows

RoostGPT CLI Installation Guide - EC2 Linux

Overview

RoostGPT is an AI-powered testing tool. This guide will help you install RoostGPT on your EC2 (RHEL or Ubuntu) instance in just a few minutes.

Pilot Prerequisites

Before installing RoostGPT, ensure your system meets the following requirements:

Requirement	Specification
Operating System	RHEL 9.5 or Ubuntu 22.04
CPU (or vCPUs)	8 or more
RAM	16 GB or more
Instance Type	c5a.2xlarge or better
Root Disk Space	Minimum 100 GB available
Additional Volume	Minimum 150 GB
Mount Point for additional volume	/var/tmp/Roost (with full permissions granted to SSH user)
Network	Internet connection enabled
Permissions	sudo/root access required (SSH user should have sudo permissions)
AWS Key & Secret	Set in environment or in .profile for accessing AWS Services

Requirement	Specification
AWS IAM User Role	Policy: AmazonBedrockFullAccess or Custom Bedrock Policy as defined below
AWS Bedrock Inference Profiles (one or more of these listed profile id)	global.anthropic.claude-sonnet-4-5-20250929-v1:0 global.anthropic.claude-sonnet-4-20250514-v1:0 eu.anthropic.claude-sonnet-4-5-20250929-v1:0 eu.anthropic.claude-sonnet-4-20250514-v1:0 eu.anthropic.claude-3-7-sonnet-20250219-v1:0
RoostGPT License & Installer	Provided by Roost Team
POC 1 - Input Sample	1. Word document or PDF with User Story or System Analysis Document 2. Optional: API Specification that is representative of current features
POC 2 - Input Sample	1. Swagger API Spec

AWS Bedrock Model Access to one or more of the anthropic (cross-region) models is fine.

Custom Bedrock Policy for AWS IAM User

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "BedrockAPIs",
         "Effect": "Allow",
         "Action": [
            "bedrock:Get*",
            "bedrock:List*",
            "bedrock:CallWithBearerToken",
            "bedrock:BatchDeleteEvaluationJob",
            "bedrock:CreateEvaluationJob",
            "bedrock:CreateGuardrail",
            "bedrock:CreateGuardrailVersion",
            "bedrock:CreateInferenceProfile",
            "bedrock:CreateModelCopyJob",
            "bedrock:CreateModelCustomizationJob",
```

```
            "bedrock:CreateModelImportJob",
            "bedrock:CreateModelInvocationJob",
            "bedrock:CreatePromptRouter",
            "bedrock:CreateProvisionedModelThroughput",
            "bedrock:DeleteCustomModel",
            "bedrock:DeleteGuardrail",
            "bedrock:DeleteImportedModel",
            "bedrock:DeleteInferenceProfile",
            "bedrock:DeletePromptRouter",
            "bedrock:DeleteProvisionedModelThroughput",
            "bedrock:StopEvaluationJob",
            "bedrock:StopModelCustomizationJob",
            "bedrock:StopModelInvocationJob",
            "bedrock:TagResource",
            "bedrock:UntagResource",
            "bedrock:UpdateGuardrail",
            "bedrock:UpdateProvisionedModelThroughput",
            "bedrock:ApplyGuardrail",
            "bedrock:InvokeModel",
            "bedrock:InvokeModelWithResponseStream"
        ],
        "Resource": "*"
      }
    ]
}
```

- Configure AWS profile into launched server, either by
  - setting **AWS_ACCESS_KEY_ID** and **AWS_SECRET_ACCESS_KEY**, in $HOME/.profile
  - Or stage $HOME/.aws/credentials
- Transfer roostgpt-installer executable into the EC2 instance
- Ensure the below is installed on the instance to support troubleshooting.
  - Python3 (version 3.3 or later)
  - AWS CLI (version aws-cli/2.31.29 or later)

Details of Roost Installer

Bundled artifact in the Roost installer

Name	Purpose / Description	Location
roostgpt-linux	CLI executable for roostGPT	Bundled binary Reference: v1.1.18/roostgpt-linux
roostenc-linux	Decryt/Encrypt binary for troubleshooting RoostGPT behaviour	Bundled in the roostgpt-installer (not available publicly)
license/license.ral	Roost License File	Bundled in the roostgpt-installer (not available publicly)
workspace/postman-api-test	Sample swagger API spec file and env to generate postman collection with tests	API Spec: weather,yaml Reference: https://openweathermap.org/api
workspace/functional-test	Sample Functional Test project with env	Sample word document written by Roost Team - (not publicly available)

Installation Steps

Mount EBS Volume (optional, check if it is done already)

1. Verify that /var/tmp/Roost does not exist already
2. Check the read/write/execute permissions on the mount point
3. Extract the volume ID
4. Mount the disk, volume as Roost folder
5. Add fstab entry for handling m/c reboot

```
# Verify if the Roost folder exists
df -h /var/tmp/Roost

# if not create directory
sudo mkdir $ROOST_DIR
sudo chown `id -u`:`id -g` ${ROOST_DIR}

# Check if volume and disk exists
# Typically this is the
export DISK="nvme0n1"
export EBS_VOLUME="/dev/nvme0n1"
blkid | grep "${DISK}:"
```

```
lsblk | grep -w "${EBS_VOLUME}"

# Run below snippet to mount disk and add entry to fstab file
    lsblk -f | grep -w "${DISK}" | grep "ext4"
    if [ $? -ne 0 ]; then
      sudo mkfs -t ext4 ${EBS_VOLUME}
    fi
    if [ ! -d "${ROOST_DIR}" ]; then
      sudo mkdir ${ROOST_DIR}
    fi
    sudo mount ${EBS_VOLUME} ${ROOST_DIR}
    mount | grep "$EBS_VOLUME"
    if [ $? -eq 0 ]; then
      sudo chown `id -u`:`id -g` ${ROOST_DIR}

      grep -v "${ROOST_DIR}" /etc/fstab  | sudo tee /etc/fstab.noroost
      # Plan for reboot
      epoch=$(date +%s)
      sudo cp /etc/fstab /etc/fstab.orig.${epoch}
      which blkid
      if [ $? -eq 0 ]; then
        uuid=$(blkid | grep "${EBS_VOLUME}:" | awk '{print $2}'|sed -e 's/"//g')
      fi
      if [ -z $uuid ]; then
        id=$(lsblk -f -o NAME,UUID | grep -w "${DISK}" | awk '{print $2}'|sed -e 's/"//g')
        uuid="UUID=${id}"
      fi
      if [ ! -z $uuid ]; then
        echo "${uuid}     ${ROOST_DIR}     ext4    defaults        0       2" | sudo tee -a /etc/fstab.noroost
        sudo cp /etc/fstab.noroost /etc/fstab
        sudo umount ${ROOST_DIR}
        sudo mount -a
        if [ $? -eq 0 ]; then
          mount | grep "$EBS_VOLUME"
          if [ $? -ne 0 ]; then
           sudo chown `id -u`:`id -g` ${ROOST_DIR}
          fi
        fi
      fi
```

Copy the installer and license file

1. Download Installer from Shared Google Drive <LOCATION SENT OVER EMAIL>
   - You have to download the RoostGPT installer (Roost License file will be bundled) from a shared Google Drive.
2. Transfer Installer using scp
3. Make Installer executable

```
# From your local machine, copy installer to Ubuntu instance
scp roostgpt-installer.sh ubuntu@<your-instance-ip>:~/

# Then SSH into your instance
ssh ubuntu@<your-instance-ip>



# Add execute permissions to installer
chmod +x roostgpt-installer.sh

# Execute Roost Installer
sudo ./roostgpt-installer.sh
```

Installation interface:

```
╔═══════════════════════════════════════════════════════════════════════╗
║                                       ║
║            RoostGPT Installer                ║
║                                       ║
╚═══════════════════════════════════════════════════════════════════════╝


[→] Checking system requirements...
[✓] System requirements met


Continue with installation? [Y/n]
```

Press **Y** and **Enter** to continue.

The installer will:

- Extract files
- Install RoostGPT binary

- Unpack the code parsers
- Copy workspace with pre-canned examples
- Configure all necessary permissions

Installation typically completes in **under 1 minute**.

Verify Installation

After successful installation, you'll see:

```
╔═══════════════════════════════════════════════════════╗
║                                                       ║
║     ✓ RoostGPT Installation Completed Successfully!   ║
║                                                       ║
╚═══════════════════════════════════════════════════════╝


Now you can run: roostgpt
```

Installed Samples

```
# Show the installed sample workspaces


find ~/workspace -ls
```

Structure - Roost Sample Workspace

The workspace contains example projects to help you get started:

```
~/workspace/
├── functional-test/
   └── functional-test.env          # Configuration for POC 1
   └── run.sh                     # Run script for POC 1
   └── UserStory.docx             # Sample user requirements doc for POC 1
└── postman-api-test/
   └── postman-api-test.env         # Configuration for POC 2
   └── run.sh                     # Run script for POC 2
   └── weather.yaml               # Sample Swagger Spec for POC 2
```

Configuring AWS Credentials

Adding AWS Credentials

```
# SSH into your instance
ssh ubuntu@<your-instance-ip>

# Edit $HOME/.bash_profile
vim $HOME/.bash_profile

# Add export AWS keys to your $HOME/.profile and save the file
export AWS_ACCESS_KEY_ID=xxx
export AWS_SECRET_ACCESS_KEY=xxx
export AWS_DEFAULT_REGION=eu-west-1
export AWS_BEDROCK_MODEL_TYPE=cross-region # Supports [foundation, cross-region]
export AWS_BEDROCK_MODEL=global.anthropic.claude-sonnet-4-20250514-v1:0

# Source the file .profile to make the AWS creds available in the same session
source $HOME/.bash_profile
```

Using RoostGPT

Running RoostGPT

You can now access RoostGPT from anywhere on your system, using the command roostgpt

Quick Start Example

```
# 1. SSH into your Ubuntu instance
ssh ubuntu@<your-instance-ip>

# 2. Navigate to workspace
cd ~/workspace/functional-test

# 3. Run RoostGPT
./run.sh
```

```
# OR you can run using the full command
roostgpt test create -c functional-test.env
```

Viewing Roost Test Results

```
# Roostgpt test results can be viewed using the below command
roostgpt test result

# A sample Postman Collection with tests from a swagger spec will be generated under a folder
postman_collections/
 |_  modified_postman_weather.json
 |_  postman-api-test/
      |_   Realtime_API.json
      |_   Forecast_API.json
      |_   Future_API.json
      |_   History_API.json
      |_   Marine_Weather_API.json
      |_   Search_Autocomplete_API.json
      |_   IP_Lookup_API.json
      |_   Time_Zone_API.json
      |_   Astronomy_API.json
```

Here's a complete workflow from login to running RoostGPT:

Configure a new test

Each project folder in your workspace contains a `<UNIQUE-TEST-NAME>.env` file for configuration. You can copy this env and modify the contents for a new test run. Or create a new test env file.

Execute roostgpt with this new env.

```
# Create a new Roost Test Configuration File
vim new_test.env

####################################
# Modify the below attributes
```

```
TEST_NAME=ANY_UNIQUE_NAME    # Any unique_name to represent the test configuration
TEST_TYPE=functional         # Default is unit; Supports [unit, functional, api-spec-test, integration]
AI_TYPE=bedrock_ai           # Default is openai; Supports [openai, azure_open_ai, claude_ai, dbrx, bedrock_ai, gemini]

ROOST_USER_INPUT_TYPE=file               # Roost User input Type for test generation; Supports [text, file]
ROOST_USER_INPUT_FILE=USER_FILE_PATH     # The user input file-path will be used to generate tests

# GIT env vars
GIT_TYPE=local               # Default is github; Supports [github, gitlab, azure, bitbucket, local]
LOCAL_PROJECT_PATH=PATH_TO_WORKSPACE_FOLDER       # Required if GIT_TYPE is local, Path to your workspace.

# AWS Bedrock env vars
AWS_ACCESS_KEY_ID=                       # Required: If not defined here, it will be picked from environment
AWS_SECRET_ACCESS_KEY=                   # Required: If not defined here, it will be picked from environment
AWS_SESSION_TOKEN=                       # Optional; AWS Session token.
AWS_DEFAULT_REGION=eu-west-1             # Required if AI_TYPE=aws_bedrock.
AWS_BEDROCK_MODEL_TYPE=cross-region      # Required if AI_TYPE=aws_bedrock.
AWS_BEDROCK_MODEL=global.anthropic.claude-sonnet-4-20250514-v1:0       # Required if AI_TYPE=aws_bedrock.

# Api Spec env vars
API_SPEC_TYPE=swagger                    # Optional; Supports [swagger]
API_SPEC_SOURCE=file                     # Optional; Supports [file, gitpath, url]
API_SPEC_FILE_PATH="$HOME/FILE"          # Optional; path of the source file.
API_SPEC_URL=                            # Optional; URL of the source file, if source is url

####################################

# Run the new test configuration
roostgpt test -c new_test.env
```

Troubleshooting

Installation Issues

Problem: Permission denied when running installer

```
# Solution: Make sure you're using sudo
sudo ./roostgpt-installer.sh
```

Problem: Command not found: roostgpt

```
# Solution: Verify installation completed successfully
which roostgpt
# Should output: /usr/local/bin/roostgpt

# Check executable permissions on roostgpt and grant if missing
chmod +x $(which roostgpt)

# If not found, re-run the installer
sudo ./roostgpt-installer.sh
```

Problem: Insufficient disk space

```
# Solution: Check available space
df -h $HOME /var/tmp/Roost

# Free up space or use an instance with more storage
```

Runtime Issues

Problem: Can't find workspace

```
# Solution: Workspace is in your home directory
cd ~/workspace
ls -la
```

Problem: Configuration not loading

```
# Solution: Ensure .env files exist and are readable
find ~/workspace/ -name '*.env' -ls
find ~/workspace/ -name '*.env' -exec chmod 644 {} \;
```

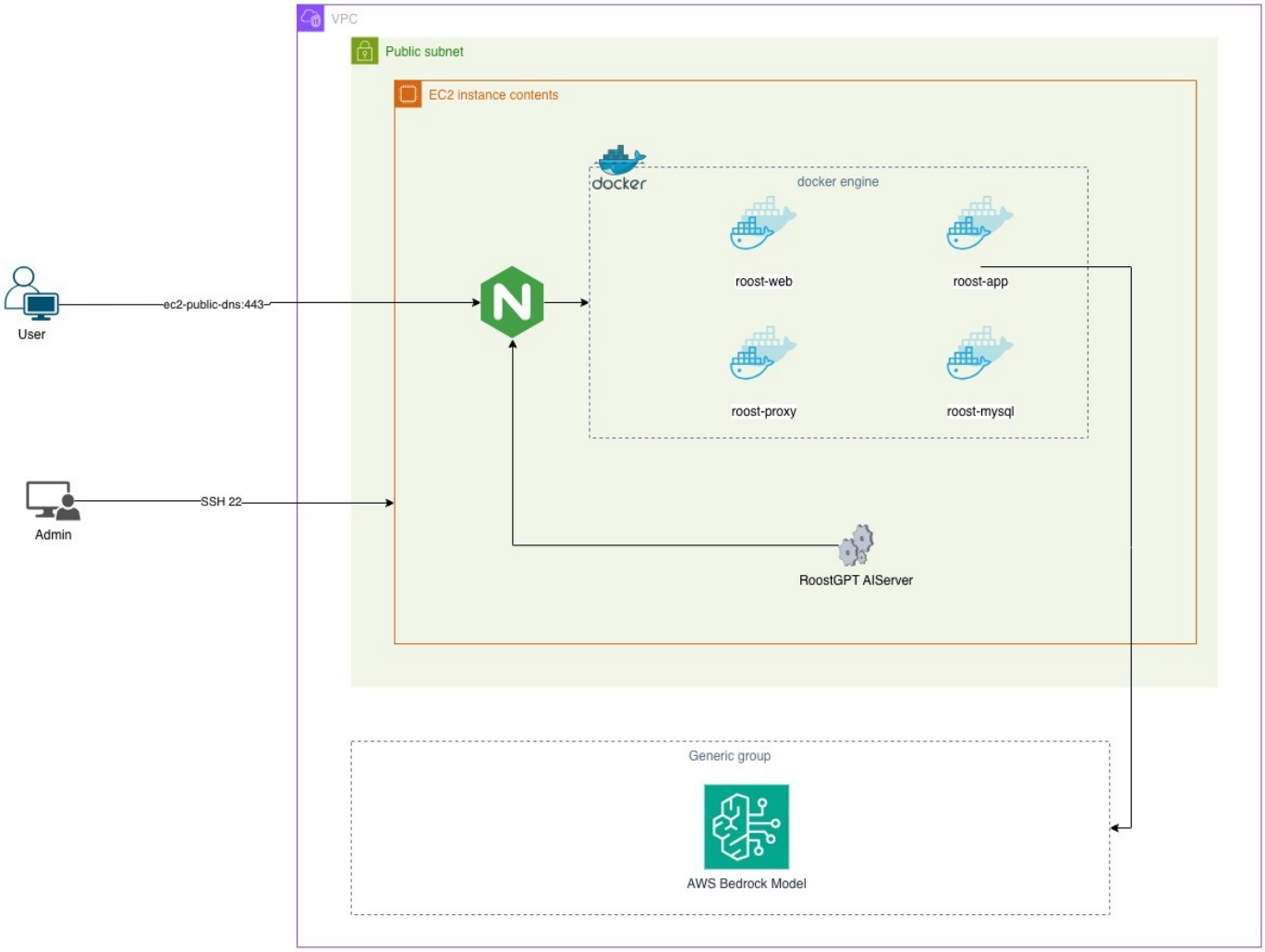
Verify AWS Bedrock Access

```
aws bedrock list-inference-profiles --region eu-west-1
```

EC2 Requirements for Docker based Roost

Roost Pilot Setup (Single EC2 Instance)

Architecture of RoostGPT stack running in single instance:



Using Terraform:

To accomplish POC from single EC2 instance where RoostGPT can be setup, you can run below terraform scripts:

Link: https://github.com/roost-io/install/tree/demo/terraform/aws/bedrockdemo

Prerequisites:

- VPC ID where instance would be created
- Public Subnet ID.
- Configure AWS profile into launched server, either by setting AWS_ACCESS_KEY and AWS_SECRET_KEY_ID, in $HOME/.profile or staging $HOME/.aws/credentials
- Allowed SSH CIDR range (Could be individual IP, company network etc)

Instance Details

Whether using terraform or provisioning EC2 externally, Roost expects following configurations

- Region (eu-west-1 or any)
- Instance_size: c5a.2xlarge (16 GB Memory, 8 vCPUs) or bigger
- Instance root disk size: Minimum 100 GB
- Additional EBS disk size: Minimum 150 GB
- Image: Ubuntu 22.04 HVM base, SSD Volume Type
- Network Configuration:
  - Accepts CIDR range to allow SSH (port 22)
  - Allow SSH from 4.247.149.66/32 and 40.112.174.40/32 (Roost Support)
  - HTTPS traffic is enabled on EC2 at port 443

- Python and AWS CLI are already installed.
- If provisioned using terraform, a new ssh key-pair is created and kept under terraform-root-dir/data dir. (SSH keypair to be shared with Roost Support team)
- Default SSH user is Ubuntu which must have sudo permission
- To access RoostGPT over HTTPS (optional), we will need -
  - Domain certs
  - DNS Name.

Packages that will be installed on EC2 Linux (by roostGPT installer):

1. curl
2. jq
3. pkill
4. shasum
5. gzip
6. docker-ce
7. docker-cli

8. docker-compose
9. nginx
10. nginx-extras
11. Entry into crontab
12. Script into init.d

Configurations to run RoostGPT:

- SSH user should have sudo permissions
- AWS IAM User should have **AmazonBedrockFullAccess** or the below permission -

```
{
  Version = "2012-10-17"
  Statement = [
    {
      Effect = "Allow"
      Action = [
        "bedrock:InvokeModel",
        "bedrock:InvokeModelWithResponseStream",
        "bedrock:ListFoundationModels",
        "bedrock:GetFoundationModel"
      ]
      Resource = "*"
    },
    {
      "Effect": "Allow",
      "Action": [
          "bedrock:GetInferenceProfile",
          "bedrock:ListInferenceProfiles",
      ],
      "Resource": [
          "arn:aws:bedrock:*:*:inference-profile/*",
          "arn:aws:bedrock:*:*:application-inference-profile/*"
      ]
    }
  ]
}
```

- Configure AWS profile into launched server, either by
  - setting AWS_ACCESS_KEY_ID and AWS_SECRET_ACCESS_KEY, in $HOME/.profile
  - Or stage $HOME/.aws/credentials
- Transfer roost license file (.ral) into EC2 instance.

- Ensure python3 is installed on the instance

Why does RoostGPT need to run with a privileged user?

- RoostGPT installs docker and nginx like services on the EC2
- RoostGPT docker containers run as root.
- RoostGPT adds crontab entry for the current user and also adds an init.d script to handle Roost processes on a m/c reboot.

Roost on Windows

Steps to install

Use the Windows installer provided by Roost team.

Upon a successful installation, you will see the files copied on your system

File Name	Type	Location
roostgpt-win.exe	Windows executable	C:\Program Files (x86)\RoostGPT [Default]
RoostJavaASTParser.jar	Java archive	$USER_PROFILE\var\tmp\Roost\bin
RoostUITestGenerator-win.exe	Windows executable	$USER_PROFILE\var\tmp\Roost\bin
license.ral	Encrypted data file	$USER_PROFILE/.roost
roost-workspace (folder)	RoostGPT Pre-canned Samples	$USERPROFILE/roost-workspace

Pre-requisites to run Roost Generated Tests

To run the Roost generated tests successfully on the local system, you will need pre-requisites installed on the system

Pre-requisites for Java Unit Test:

- JDK/JRE
- Maven/Gradle

Pre-requisites for API Spec Test:

- Postman/Newman

Pre-requisites for UI Test:

- Playwright

Configure AWS Credentials

```
# Add AWS keys to your $HOME/.profile and save the file OR
setx AWS_ACCESS_KEY_ID "xxx"
setx AWS_SECRET_ACCESS_KEY "xxx"
setx AWS_DEFAULT_REGION "eu-west-1"
setx AWS_BEDROCK_MODEL_TYPE "cross-region" # Supports [foundation, cross-region]
setx AWS_BEDROCK_MODEL "global.anthropic.claude-sonnet-4-20250514-v1:0"
```

Roost Samples

Structure - Roost Sample Workspace

The workspace contains example projects to help you get started:

```
~/workspace/
├── functional-test/
    └── functional-test.env            # Configuration for POC 1
    └── prepaid-card-requirement.docx   # Sample user requirements doc for POC 1
    └── prepaid-card-swagger.yaml       # Required yaml for the POC 1
└── postman-api-test/
    └── postman-api-test.env          # Configuration for POC 2
    └── prepaid-card-swagger.yaml     # Sample Swagger Spec for POC 2
    └── weather.yaml                  # Sample Swagger Spec for POC 2
```

Using RoostGPT

Running RoostGPT

You can now access RoostGPT from anywhere on your system, using the command roostgpt

Quick Start Example

```
# 1. Navigate to workspace
cd $env:USERPROFILE\roost-workspace`


# **Ensure the env is updated with latest/required creds.**
```

```
# you can run using the full command
**run the below the command to generate the functional-test.**
`roostgpt.exe test create -c .\functional-test\functional-test.env`

**run the below the command to generate the Postman-api-test.**
`roostgpt.exe test create -c .\postman-api-test\postman-api-test.env`
```

Viewing Roost Test Results

```
# Roostgpt test results can be viewed using the below command
roostgpt test result

# A sample Postman Collection with tests from a swagger spec will be generated under a folder
postman_collections\
  |_ roost-postman\
    |_ prepaid-card-swagger\
      |_ roost_postman_prepaid-card-swagger_<epoch>
        |_ datasets\
         |_  User_login.json
```

After running the test result command successfully.

Below is the example of Functional test.

```
Loading result dashboard...


   ┌──────────── Summary ────────────┐
   │                                 │
   │   Test Name: functional-test    │
   │   Test Type: FUNCTIONAL         │
   │   Status: COMPLETED             │
   │                                 │
```

Git Details

Git Information	Value
Git Type	LOCAL
Project Path	.
Branch	roost-1763108488
Repo Path	.

AI Usage Statistics

AI Usage	Value
AI Type	bedrock_ai
Model	global.anthropic.claude-sonnet-4-20250514-v1:0
API Calls	4
Prompt Tokens	35,698
Completion Tokens	21,596

Generated Files

roost_test_1763108488

File Type / Method	Path
Gherkin Feature	functional_tests/roost_test_1763108488/roost_test_1763108488.feature
Functional Test	functional_tests/roost_test_1763108488/roost_test_1763108488.json
API Specification	functional_tests/roost_test_1763108488/prepaid-card-swagger_enhance.yaml
CSV Report	functional_tests/roost_test_1763108488/roost_test_1763108488.csv
Excel Report	functional_tests/roost_test_1763108488/roost_test_1763108488.xlsx

Configuration

Configuration	Value
User Input File	prepaid-card-requirement.docx
Create API Spec	true

```
──────────────────────────────────────┤
│ Create Excel              │ true                                                                  │
├──────────────────────────┼──────────────────────────────────────────────────────────────────────
──────────────────────────────────────┤
│ API Spec Type             │ swagger                                                               │
├──────────────────────────┼──────────────────────────────────────────────────────────────────────
──────────────────────────────────────┤
│ API Spec Source           │ file                                                                  │
├──────────────────────────┼──────────────────────────────────────────────────────────────────────
──────────────────────────────────────┤
│ API Spec File Path        │ prepaid-card-swagger.yaml                                             │
└──────────────────────────┴──────────────────────────────────────────────────────────────────────
──────────────────────────────────────┘
[14/11/2025, 1:57:14 PM] - [INFO] -
================================================================================
================================
```

After running the Postman test result command successfully.

Below is the example of Postman-api test.

```
Loading result dashboard...

  ┌──────────── Summary ────────────┐
  │                                 │
  │  Test Name: postman-api-test    │
  │  Test Type: API-SPEC-TEST       │
  │  Status: COMPLETED              │
  │                                 │
  └─────────────────────────────────┘


Git Details
┌──────────────────────────┬──────────────────────────────────────────────────────────────────────
──┐
│ Git Information          │ Value                                                                │
```

Git Type	LOCAL
Project Path	.
Branch	roost-1763116002
Repo Path	.

AI Usage Statistics

AI Usage	Value
AI Type	bedrock_ai
Model	global.anthropic.claude-sonnet-4-20250514-v1:0
API Calls	6
Prompt Tokens	4,983
Completion Tokens	2,661

Generated Files

File Type	Path
Postman Collection	postman_collections\roost-postman\prepaid-card-swagger\roost_postman_prepaid-card-swagger_1763116...

Endpoints from: postman-api-test\prepaid-card-swagger.yaml

Endpoint (Method)	Test Data
✔ Register a new user [POST]	postman_collections\roost-postman\prepaid-card-swagger\datasets\Register_a_new_user.json
✔ User login [POST]	postman_collections\roost-postman\prepaid-card-swagger\datasets\User_login.json
✔ Submit KYC documents [POST]	postman_collections\roost-postman\prepaid-card-swagger\datasets\Submit_KYC_documents.json
✔ Issue a new card [POST]	postman_collections\roost-postman\prepaid-card-swagger\datasets\Issue_a_new_card.json
✔ Load funds onto a card [POST]	postman_collections\roost-postman\prepaid-card-swagger\datasets\Load_funds_onto_a_card.json
✔ Get card transaction history [GET]	postman_collections\roost-postman\prepaid-card-swagger\datasets\Get_card_transaction_history.json

Configuration

Configuration	Value

```
├──────────────────────────┼──────────────────────────────────────────────────────────────────────────────
──────────────────────────┤
│ Test Type                │ api-spec-test                                                                │
├──────────────────────────┼──────────────────────────────────────────────────────────────────────────────
──────────────────────────┤
│ Framework                │ postman                                                                      │
├──────────────────────────┼──────────────────────────────────────────────────────────────────────────────
──────────────────────────┤
│ API Spec Type            │ swagger                                                                      │
├──────────────────────────┼──────────────────────────────────────────────────────────────────────────────
──────────────────────────┤
│ API Spec Source          │ file                                                                         │
├──────────────────────────┼──────────────────────────────────────────────────────────────────────────────
──────────────────────────┤
│ HTTP Verbs               │ get,post,put,patch,delete                                                    │
├──────────────────────────┼──────────────────────────────────────────────────────────────────────────────
──────────────────────────┤
│ Endpoint Regex           │ .*                                                                           │
└──────────────────────────┴──────────────────────────────────────────────────────────────────────────────
──────────────────────────┘
[14/11/2025, 3:58:12 PM] - [INFO] -
================================================================================
==================================
```