forgeahead
Your Vision. Our Execution.

PyDocStrGen Deployment Guide

PyDocStrGen: Context-Aware Python Docstring Generator using GenAI

– AWS Marketplace AMI Based Solution

800 West El Camino Real
Suite 180 Mountain View, CA 94040, USA
T : +1-650-948-1787
F : +1-650-948-1789

Suite 218, 'B1' Cerebrum IT Park,
Level 2, Kalyani Nagar, Pune 411014, India
T : +91 20-6734-5900
F : +91 20-6734-5901

www.forgeahead.io

forgeahead
Your Vision. Our Execution.

1. Introduction

1.1 What is PyDocStrGen?

PyDocStrGen is a CLI-only Python documentation generation tool deployed via an EC2 AMI. It uses **Amazon Bedrock foundation models** (e.g., Anthropic Claude) to generate and update **PEP 257-compliant docstrings** across Python codebases. It supports multiple styles (Google, NumPy, PEP 257) and allows contextual enhancement via README.md files or Amazon Bedrock Knowledge Base IDs.

1.2 Key Use Cases

- **Modernize Legacy Code:** Introduce high-quality documentation to aging codebases.
- **Enforce Consistency:** Ensure uniform documentation styles across development teams.
- **Improve Onboarding:** Enhance code readability to reduce the learning curve for new engineers.
- **Release Readiness:** Guarantee documentation coverage for SDKs, public releases, or audit compliance.
- **Automate Research Code:** Automatically document machine learning or research code during early development stages.

1.3 Overview of Typical Deployment

Upon subscribing to the AMI from AWS Marketplace, a CloudFormation template orchestrates the deployment of a single EC2 instance into your AWS account. This instance comes pre-configured with:

- PyDocStrGen tool (CLI-only)
- Python 3.13 environment
- AWS SDK and Amazon Bedrock SDK for seamless AWS service integration
- A pre-configured **IAM Role** to grant secure, least-privilege access to necessary AWS services (e.g., S3, Bedrock).

The solution operates on a simple **"S3 in, S3 out" model**. You provide your source archive (e.g., .zip, .tar, .tar.gz, .tar.bz2, .tar.xz) in an S3 bucket. The tool processes it and uploads the documented version back to the same bucket in the same format and path.

1.4 Deployment Options

- **Single-Instance, Single-AZ:** The standard deployment model is a single EC2 instance running in one Availability Zone. This is suitable for the tool's on-demand, batch-processing nature.
- **Multi-AZ / Multi-Region:** Not applicable. PyDocStrGen is a CLI-based tool designed for single-instance operation.

1.5 Supported Regions

PyDocStrGen is supported in any AWS Region where Amazon Bedrock is available. As of this writing, key supported regions include:

- US East (N. Virginia) (us-east-1)
- US West (Oregon) (us-west-2)
- US East (Ohio) (us-east-2)

For an up-to-date list, please refer to the official AWS Regional Services List.

1.6 Deployment Time

The end-to-end deployment typically takes **5-10 minutes**.

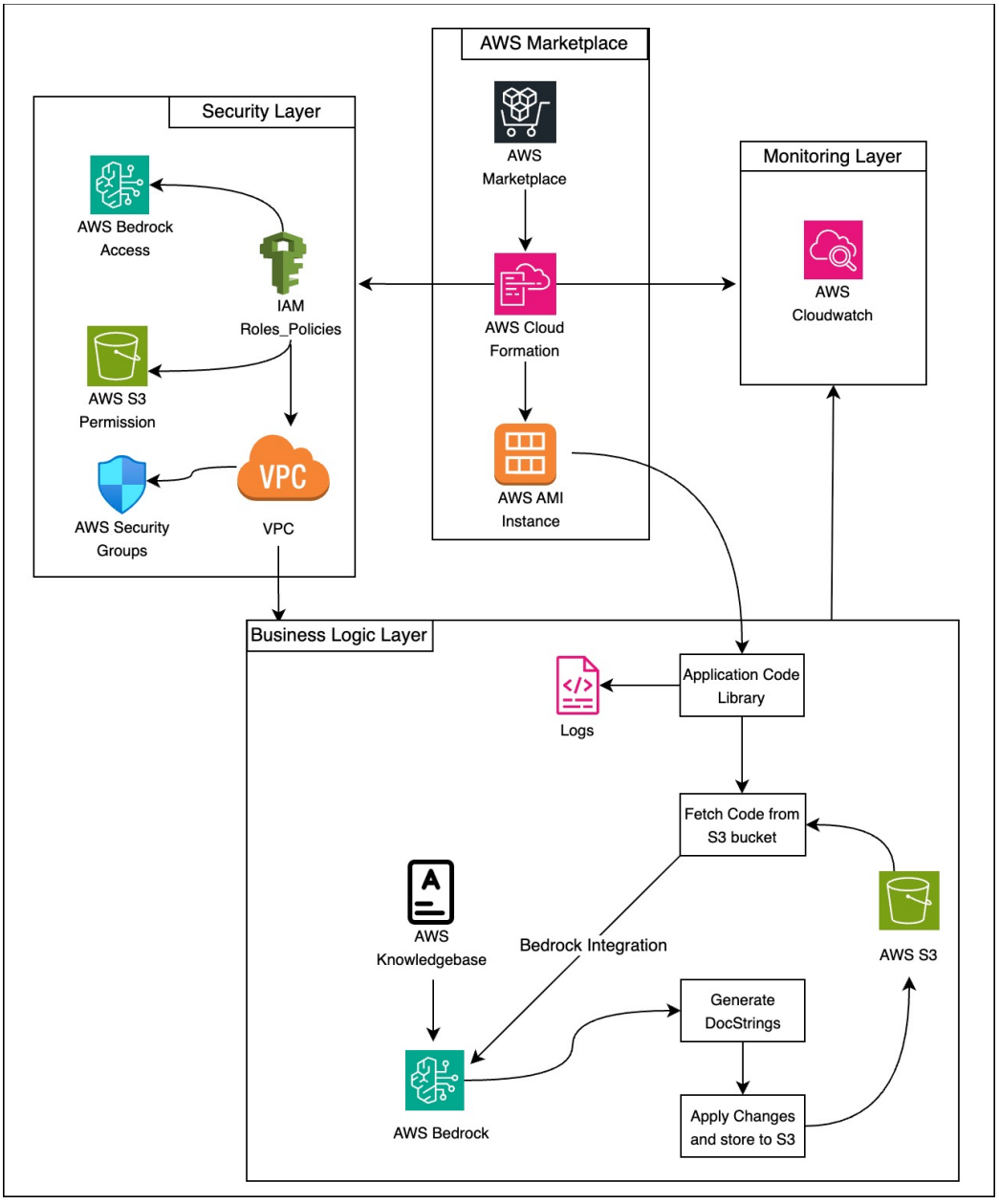
- CloudFormation Stack Launch: ~5-7 minutes
- Initial Test Run: ~2-5 minutes

2. Architecture

2.1 High-Level Architecture

The solution deploys all resources within a customer's VPC. The user provides an S3 bucket location that serves as both input and output. The PyDocStrGen EC2 instance downloads the source code (zip, tar, tar.gz etc.) from the source S3 bucket, uses Amazon

Bedrock to generate docstrings, and uploads the processed code to the same S3 bucket and path in the same archive format.



AWS Marketplace Layer

This layer represents the product listing and deployment mechanism:

- **AWS Marketplace:** Where customers find and subscribe to PyDocGen.
- **AWS CloudFormation:** Automates the provisioning of the solution (infrastructure + app).
- **AWS AMI Instance:** Launches pre-built EC2 instances containing PyDocGen backend logic and configurations.

Security Layer

This layer secures access to AWS services and isolates the environment:

- **IAM Roles/Policies:** Controls access to AWS services like Bedrock, S3. Role is assigned to EC2.
- **AWS Bedrock Access:** Authorized interaction with foundational models like Claude or Titan for docstring generation.
- **S3 Permission:** Enables secure reading/writing of project docs and output files.
- **VPC:** The virtual network containing all AWS resources.
- **AWS Security Groups:** Define traffic rules for compute instances and endpoints.

Business Logic Layer

This is the operational core of PyDocGen.

- **Application Code Library:** Library which will invoke AI Agent to perform Docstring generation and other operations.
- **Fetch Code from S3 bucket:** The application running on the EC2 instance will likely retrieve the user's Python codebase from S3 bucket.

Context Integration

- **Knowledgebase:** PyDocStrGen utilizes customer-provided Amazon Bedrock Knowledge Bases to enrich context for docstring generation.

AI Inference

- **AWS Bedrock:** Understands the Context from the knowledgebase, and based on this generates high-quality, contextual docstrings on the ingested Code.

Output

- **Generate DocStrings:** Creates docstrings using bedrock domain understanding capability.
- **Apply Changes and store to S3:** After generation, your application applies these new/updated docstrings to the code. The modified code (with new docstrings) is then stored back into an S3 bucket.

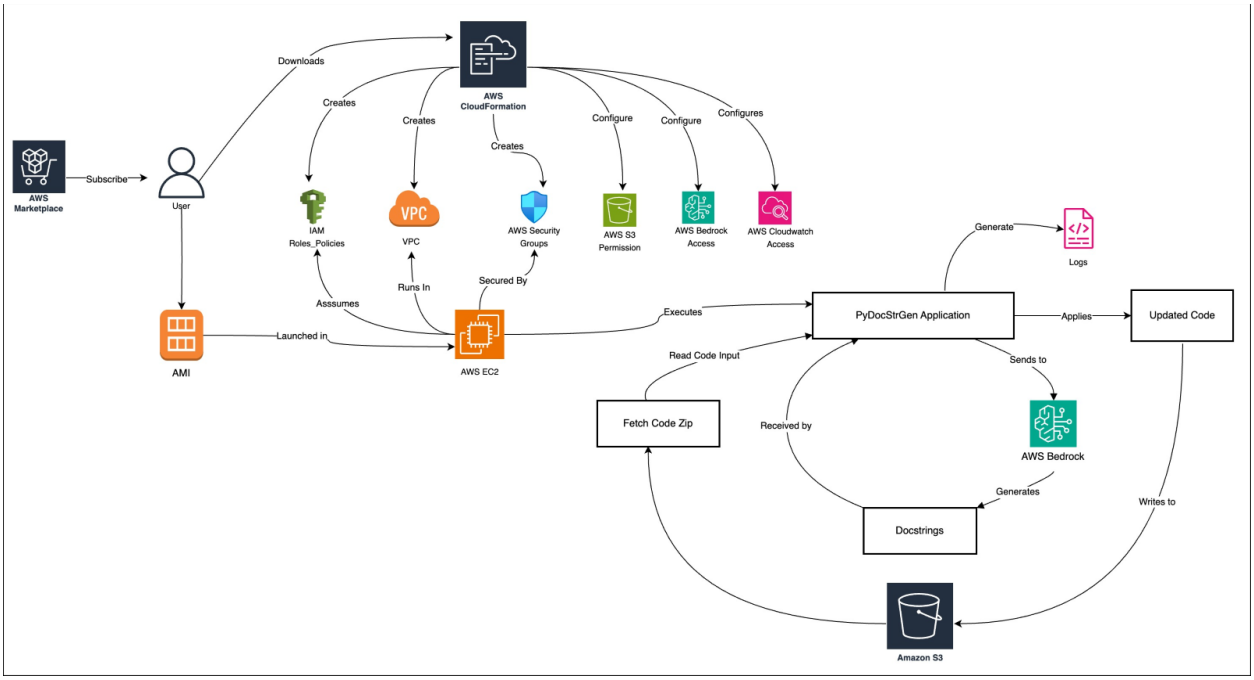
Monitoring Layer

This new layer provides observability, compliance, and auditing.

- **Amazon CloudWatch:** Monitors logs, system health, errors, and performance metrics (e.g., inference duration, S3 activity).

2.2 Network Diagram

The CloudFormation template provisions a VPC with public and private subnets. The EC2 instance is placed in a private subnet and uses a NAT Gateway in the public subnet for outbound internet access to Amazon Bedrock and Amazon S3 service endpoints.



The user subscribes to the product listed on AWS Marketplace.

The user downloads and runs a CloudFormation template.

This template automates infrastructure provisioning:

- Creates a VPC, security groups, and necessary IAM roles/policies.
- Launches a single EC2 instance using the AMI.
- Configures access permissions for:
    - Amazon S3 (to fetch/store code archives)
    - Amazon Bedrock (for AI inference)
    - CloudWatch (for monitoring/logging) The EC2 instance is launched inside the VPC, secured by security groups. It assumes the IAM role for controlled service access. The PyDocStrGen application is executed on the EC2 instance. It reads zipped Python code files from Amazon S3 (e.g. zip, .tar, .tar.gz, .tar.bz2, .tar.xz). Extracts and analyzes the Python code.

PyDocStrGen uses Amazon Bedrock (likely via Claude, Nova, or similar foundation models) to:

- Analyze the source code
- Generate accurate docstrings
- Review the generated docstrings and regenerate if needed The generated docstrings are returned to the application. The tool inserts the generated docstrings back into the source code. The updated code archive is written back to Amazon S3 — typically in the same location and format as the input. PyDocStrGen also generates logs, which are stored locally on the EC2 instance at /var/log/pydocstrgen/app.log and streamed to Amazon CloudWatch Logs for centralized monitoring.

3. Prerequisites and Requirements

3.1 AWS Account and Environment

- An active AWS account.
- Permissions to subscribe to AWS Marketplace AMIs and create CloudFormation stacks.
- An S3 bucket must be provided, or a new one will be created during deployment. This single S3 bucket will be used by PyDocStrGen for both accessing source archive (e.g., .zip, .tar, .tar.gz, .tar.bz2, .tar.xz) and storing processed output. User will need permissions to upload files to this bucket.
- **User IAM Permissions for Instance Access:** Your IAM user or role needs permissions to start and manage sessions with the EC2 instance via AWS Systems Manager Session Manager (e.g., ssm:StartSession, ssm:TerminateSession, ec2:DescribeInstances). The deployed EC2 instance itself will be configured with the necessary IAM instance profile to allow SSM Agent communication.
- PyDocStrGen uses inline agents with AWS Bedrock foundation models. To ensure seamless operation, you must confirm you have access to your chosen foundation model in the desired AWS region. Note that AWS requires you to request access to some models manually through the Amazon Bedrock console.

3.2 Required Skills

- **AWS:** Basic familiarity with the AWS Management Console, specifically with EC2, IAM, S3, SSM and CloudFormation.
- **Python:** Basic familiarity with Python is helpful but not required for operation.

3.3 AMI Configuration

- **Operating System:** Ubuntu LTS (latest)
- **Installed Software:**
    - PyDocStrGen CLI
    - Python 3.13
    - AWS SDKs (boto3, botocore)
    - Amazon Bedrock Runtime SDK
    - Required Python libraries

4. Security

4.1 IAM and Least Privilege

The solution follows the principle of **least privilege**. The CloudFormation template creates a highly-scoped IAM role for the EC2 instance that grants only the minimal permissions required:

- s3:GetObject: Scoped to the specified S3 bucket and prefix.
- s3:PutObject: Scoped to the specified S3 bucket and prefix.
- s3:ListBucket: To list objects in the bucket for processing.
- Standard permissions for SSM and CloudWatch logging.
- bedrock:InvokeModel:
- bedrock:ListFoundationModels
- bedrock:InvokeInlineAgent
- bedrock:InvokeAgent
- bedrock:GetAgent
- bedrock:CreateAgent

- bedrock:UpdateAgent
- bedrock:GetFoundationModel
- bedrock:InvokeModelWithResponseStream
- bedrock:GetInferenceProfile

All these bedrock permissions scoped to wildcard ("Resource": "*") because the PyDocStrGen solution needs flexibility to access various foundation models as they become available, without requiring template updates.

An example of the FTR-compliant IAM policy is available in Appendix B.

4.2 Data Encryption

- **Encryption in Transit:** All data in transit between the user, AWS services (SSM, Bedrock, S3) is encrypted using TLS 1.2 or higher.
- **Encryption at Rest:**
  - The root EBS volume is not encrypted by default to allow flexibility.
  - Customers can enable encryption using AWS default or custom KMS keys.
  - Default SSE: AES-256 encryption is enabled for S3 buckets.

4.3 Network Configuration

The solution deploys a new, secure VPC by default.

- **Security Groups:** A security group is attached to the EC2 instance that denies all inbound traffic. All outbound HTTPS (port 443) traffic is allowed to AWS service endpoints.
- **Public Resources:** No public resources or open inbound ports are created. SSH access is disabled by default.

4.4 IMDSv2 Enforcement

For enhanced security, the EC2 instance provisioned by this CloudFormation template automatically enforces **IMDSv2**.

5. Cost and Licensing

5.1 Licensing Model

PyDocStrGen itself is available via the AWS Marketplace at no charge. However, please note that you will still be billed for any underlying AWS services utilized to run it.

5.2 Billable AWS Services

The following AWS services are provisioned by the PyDocStrGen CloudFormation template and will incur costs in the customer's AWS account.

- **Amazon EC2**
  - **Description:** Billed for the running instance (e.g., t3.medium) that hosts the PyDocStrGen application.
  - **Status:** Mandatory. An EC2 instance is the core compute resource for PyDocStrGen.
- **Amazon S3**
  - **Description:** Standard storage and data transfer costs for your input and output buckets. This includes storing your source code archives and the processed, documented output.
  - **Status:** Mandatory. S3 is fundamental to the "S3 in, S3 out" operational model for code processing.
- **Amazon Bedrock**
  - **Description:** AWS Bedrock will bill you on a pay-per-use basis for model inference during docstring generation. You'll need to be mindful about the amount of code you process, as your billing is based on the number of tokens consumed, which is directly proportional to the amount of code processed.
  - **Status:** Mandatory. Bedrock provides the core AI/ML capabilities for docstring generation.
- **NAT Gateway**
  - **Description:** Billed per hour and for data processing. It provides outbound internet access from the private subnet to AWS service endpoints.

- **Status:** Conditional/Mandatory (if creating new VPC). It is mandatory if the customer chooses to create a new VPC with private subnets via the CloudFormation template. If the customer uses an existing VPC that already provides outbound internet access from the private subnet (e.g., through VPC Endpoints, or an existing NAT Gateway), then a new NAT Gateway might not be provisioned by the template for the existing VPC scenario, but some form of outbound connectivity from the private subnet is required.

- **Amazon CloudWatch**
  - **Description:** Incurs minimal costs for logs and metrics. This includes storing application logs in CloudWatch Logs and monitoring performance metrics.
  - **Status:** Mandatory. CloudWatch is provisioned by the template for logging and monitoring functionality.

5.3 Cost Monitoring and Control

Effective cost management is crucial. We strongly recommend customers proactively monitor and control their AWS expenditure for PyDocStrGen.

Key Tools:

- **AWS Budgets:** Set up budgets to monitor costs and receive alerts for various services (EC2, S3, Bedrock, NAT Gateway), helping prevent unexpected charges.
- **AWS Cost Explorer:** Use for visualizing, understanding, and analyzing your costs by service, instance type, and tags, identifying trends and anomalies.

Cost Optimization Strategies:

PyDocStrGen's main cost drivers are Amazon EC2, Amazon S3, Amazon Bedrock, and the NAT Gateway.

- **Amazon EC2 Instance:**
  - **Stop/Terminate When Not in Use:** As a CLI tool, the EC2 instance may not need to run 24/7. Stopping the instance when inactive significantly reduces EC2 compute costs. Consider terminating and redeploying for maximum savings on infrequent usage. Instance itself is stateless.

    - **Right-Sizing:** Select an InstanceType that precisely matches your workload performance needs to avoid over-provisioning.
- **Amazon S3 Storage:**
    - Implement S3 Lifecycle Policies or Intelligent-Tiering to automatically move less-accessed data to more cost-effective storage classes.
- **Amazon Bedrock API:**
    - Be aware that Bedrock costs are token-based. Monitor API usage and consider the cost-effectiveness of your chosen foundation model.
- **NAT Gateway:**
    - This is a significant, constant cost driver, billed hourly and for data processed. It's essential for outbound connections from private subnets (e.g., to Bedrock API).

By actively monitoring these components and applying these strategies, you can maintain efficient and predictable operating costs for your PyDocStrGen deployments.

6. Launch and Deployment

6.1 Sizing and Instance Selection

- **Recommended Minimum:** t3.medium (2 vCPUs, 4 GB RAM). This is sufficient for most small to medium-sized codebases.
- **For Larger Codebases:** Consider a larger instance type (e.g., t3.large) if you experience memory or performance issues.

6.2 Launching via CloudFormation (Recommended)

- **Step 1: Subscribe in AWS Marketplace**
    - Go to the PyDocStrGen listing in the AWS Marketplace.
    - Click **Continue to Subscribe**, review, and accept the terms.
    - Click **Continue to Configuration**, select CloudFormation as the fulfillment option, choose your region, and click **Continue to Launch**.
    - On the Launch page, under Choose Action, select **Launch CloudFormation**.

- **Step 2: Launch the CloudFormation Stack**
  - You will be redirected to the AWS CloudFormation console.
  - Provide a **Stack name** (e.g., pydocstrgen-stack).
  - In the **Parameters** section, provide the name of your S3 bucket (which will be used for both source code input and processed output). For a detailed description of each parameter, see Appendix A.
  - Acknowledge that IAM resources will be created and click **Create stack**.

6.3 Post-Deployment Verification

After launching the CloudFormation stack, perform the following steps to verify the deployment:

1. Wait for the CloudFormation stack status to become CREATE_COMPLETE.
2. Navigate to the EC2 Console and find your newly created instance (it will have the name you provided, or default to PyDocStrGen-Instance).
3. Note down the **Instance ID** and the **AWS Region** where your instance is deployed. These will be needed for connecting via AWS CLI.

7. Accessing the EC2 Instance and Using the PyDocStrGen Tool

To interact with the PyDocStrGen CLI tool, you will connect to the EC2 instance using **AWS Systems Manager (SSM) Session Manager**. This is the recommended and most secure method. The pydocstrgen tool is accessible only by the ssm-user on the instance.

7.1 Prerequisites for Accessing the Instance

To connect to your EC2 instance via SSM Session Manager using the AWS CLI, ensure you have the following installed on your local machine:

1. **AWS Command Line Interface (CLI):**

a. If you don't have it installed, follow the official AWS CLI installation guide: https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html
b. After installation, configure your AWS CLI with your credentials: aws configure

2. **Session Manager Plugin for AWS CLI:**
   a. This plugin is required for aws ssm start-session to function.
   b. Follow the installation guide for your operating system: https://docs.aws.amazon.com/systems-manager/latest/userguide/session-manager-working-with-install-plugin.html
   c. **Verify Plugin Installation:** After installation, you can verify it by running: https://docs.aws.amazon.com/systems-manager/latest/userguide/session-manager-working-with-install-plugin.html#verify-plugin-installation

7.2 Connecting to the EC2 Instance

You can connect to the instance using either the AWS Management Console or the AWS CLI.

7.2.1 Connecting via AWS Management Console (Session Manager Tab)

1. Navigate to the EC2 Console.
2. Select your PyDocStrGen instance.
3. Click the **Connect** button.
4. Choose the **Session Manager** tab.
5. Click **Connect**.
6. A new browser tab will open with a terminal session. By default, this session might start in the sh shell.
7. To switch to a bash shell, which offers better command-line features, run the following command:

```
[ -z "$BASH" ] && exec /bin/bash
```

7.2.2 Connecting via AWS CLI (Session Manager)

1. Open your local terminal or command prompt.

2. Use the aws ssm start-session command, replacing {instance-id} with your instance's ID and {your-instance-region} with the AWS region where your instance is located (e.g., us-east-1):

```
aws ssm start-session --target i-0abcdef1234567890 --document-name AWS-StartInteractiveCommand --parameters 'command=["/bin/bash"]' --region us-east-1
```

   a. **Note:** The --document-name AWS-StartInteractiveCommand --parameters 'command=["/bin/bash"]' part attempts to start directly in bash. If for any reason you find yourself in an sh shell, run [-z "$BASH"] && exec /bin/bash as described above.

7.2.3 Exiting the Session

To safely exit your Session Manager session, simply type:

```
exit
```

and press Enter.

7.3 Initial Tool Verification

Once connected to the instance via Session Manager (and ensured you are in a bash shell as ssm-user), you can verify the PyDocStrGen installation by running the help command:

```
pydocstrgen --help
```

You should see the command usage and available arguments.

7.4 Note on SSH Access (Not Recommended)

We strongly recommend using SSM Session Manager for accessing the instance as it provides a secure, auditable, and keyless method without requiring inbound ports to be open. SSH access is not enabled by default.

If you choose to use SSH (not recommended):

- You will need to manually configure the security group attached to the EC2 instance to allow incoming traffic on port 22 (SSH) from your IP address.
- You will need to use an EC2 Key Pair associated with the instance during launch (which is an optional parameter in the CloudFormation template).

- The SSH command would typically look like:

```
ssh -i /path/to/your-key-pair.pem ubuntu@{instance-public-dns-or-ip}
```

  - **Important:** After connecting as the ubuntu user, you **must switch to the ssm-user** to access the pydocstrgen tool, as it is only accessible by ssm-user. You can do this by running sudo su - ssm-user or similar.

7.5 Using the PyDocStrGen CLI Tool

To use the tool, connect to the instance via AWS Systems Manager Session Manager (as described in Section 7.2). Only "ssm-user" can access the CLI tool.

7.5.1 Key Command-Line Arguments

Argument	Description	Example
--docstr-style	The desired docstring style.	--docstr-style numpy
--s3-url	S3 URI to the .zip, .tar, .tar.gz, .tar.bz2, .tar.xz file. e.g. s3://pydocstrgen-source/docstrtest/mycode.tar.gz [default: None]	--s3-url s3://pydocstrgen-source/docstrtest/mycode.tar.gz
--kb-id	Optional: The ID of an Amazon Bedrock Knowledge Base to provide additional context.	--kb-id A1B2C3D4E5

--local-path	Local path used for code processing. If not provided, the current directory is used. [default: ./]	--local-path /tmp/test/
--exclude-dirs	A comma-separated list of directories to exclude from processing.	--exclude-dirs tests,dist
--help	Display command usage	

7.5.2 Example Command

Once connected to the instance, you can start the processing job with a simple command:

```
pydocstrgen --s3-url s3://pydocstrgen-source/docstrtest/mycode.tar.gz --docstr-style numpy --kb-id A1B2C3D4E5 --local-path /tmp/test/
```

7.5.3 Configuration File

PyDocStrGen uses a configuration file for its settings, including logging and AWS Bedrock integration.

Location:

You'll find the main configuration file here: /etc/default/pydocstrgen.json

Usage:

This JSON file lets you customize PyDocStrGen's behavior. Here's what it looks like:

```
JSON
{
  "log_level": "info",
  "bedrock": {
    "model_inference_profile_id": "us.anthropic.claude-3-5-haiku-20241022-v1:0",
    "region_name": "us-east-1"
  }
}
```

- log_level: Sets how verbose the application's logs are. Common options include "info", "debug", "warning", "error", and "critical".
- bedrock: This section handles AWS Bedrock settings for model inference.
    - model_inference_profile_id: This critical parameter defines the Inference Profile ID for the large language model (LLM) used to generate docstrings. For instance, "us.anthropic.claude-3-5-haiku-20241022-v1:0" points to the Anthropic Claude 3.5 Haiku model. You can find other available Inference Profile IDs in the AWS Bedrock documentation on Base Model IDs.
        - **Important:** Before using a specific model here, ensure you have access to it in your AWS account and region. Some AWS Bedrock foundation models require a manual access request through the Amazon Bedrock console.
    - region_name: Specifies the AWS region where your Bedrock model is deployed (e.g., "us-east-1").

By changing this file, you can control PyDocStrGen's operation and choose the Bedrock model for docstring generation.

Important Note: The quality of generated docstrings, processing time, and cost will vary based on the LLM selected via model_inference_profile_id. While PyDocStrGen may work with various recent models, we've only extensively tested it with Claude 3.5 Haiku.

8. Operations and Management

8.1 Monitoring and Health Checks

The health of the application can be monitored through **Amazon CloudWatch**:

- **Logs:** Raw application logs are generated on the EC2 instance at /var/log/pydocstrgen/app.log. These logs, along with system logs, are also streamed to Amazon CloudWatch Logs for centralized monitoring.
    - **CloudWatch Log Group Names:**
        - System logs are sent to: /aws/marketplace/pydocstrgen-system-logs
        - Application logs are sent to: /aws/marketplace/pydocstrgen-application-logs
- Customers can create alarms and configure retention policies for CloudWatch Logs. By default, the log retention period is set to two weeks.

8.2 Software Upgrades and Patching

- To upgrade to a new version of PyDocStrGen: Subscribe to the new version of the AMI from the Marketplace and update your CloudFormation stack with the new AMI ID.
- **Automated OS Patching (Recommended):** Use AWS Systems Manager Patch Manager to automate OS-level security patching.

8.3 Managing Service Limits

The solution is unlikely to hit AWS service limits under normal use. Be aware of your account limits for EC2 instances and potential Amazon Bedrock throughput quotas for very large processing jobs.

9. Backup and Recovery

9.1 Backup Strategy

- **Source Code:** Your source code is safely stored in your own S3 bucket. It is your responsibility to manage versioning and backups for that bucket according to your organization's policies.
- **EC2 Instance:** The EC2 instance is stateless. If it is terminated or fails, you can launch a new one from the CloudFormation template without any data loss.

9.2 Recovery Procedures

- **Instance Failure:** Terminate the failed instance and launch a new one from the CloudFormation template.
- **Deployment Failure:** CloudFormation will automatically roll back to the last known good state. Check the Events tab in the CloudFormation console for error messages.

10. Troubleshooting and Support

10.1 Common Issues and Resolutions

Issue	Resolution Steps
Bedrock Invocation Fails	1. Verify the EC2 instance's IAM role has bedrock:InvokeModel permissions. 2. Check that the instance has a route to the internet via the NAT Gateway.
S3 Acc	1. Ensure the IAM role on the EC2 instance has s3:GetObject, s3:ListBucket, and s3:PutObject permissions. 2. Verify the permissions are correctly scoped

ess Denied	to the correct bucket specified during launch. 3. Check for any bucket policies on your S3 bucket that might be denying access.
SSM Session Fails to Start	1. Confirm the SSM Agent is running on the instance. 2. Check that the instance IAM role has the required permissions.

10.2 Getting Support

For issues related to the PyDocStrGen software, please contact our support team.

- **Email:** support-pydocstrgen@forgeahead.io

10.3 Support Tiers and SLAs

Tier	Response Time SLA	Description
Standard	24 hours business hours	Included with all subscriptions. Support via email.

11. Appendices

Appendix A: PyDocStrGen CloudFormation Template Deployment Guide

This guide provides instructions for deploying the PyDocStrGen AI Solution using the CloudFormation template.

Prerequisites

- AWS account with permissions to create CloudFormation stacks

- Access to AWS Marketplace to subscribe to the PyDocStrGen AMI
- AMI ID for the PyDocStrGen product

Deployment Options

The template supports two main deployment scenarios:

- **New VPC Deployment:** Creates a new VPC with public and private subnets
- **Existing VPC Deployment:** Uses your existing VPC infrastructure

Parameter Requirements

Required Parameters (All Deployments)

- AMIId: The PyDocStrGen Product AMI ID from AWS Marketplace

Network Configuration

- **Option 1: Create New VPC (NetworkConfiguration = CreateNewVPC)**
  - Required parameters:
    - NewVPCCIDR: CIDR block for the new VPC (e.g., 172.16.0.0/16)
    - PublicSubnetCIDR: CIDR block for the public subnet (e.g., 172.16.1.0/24)
    - NewPrivateSubnetCIDR: CIDR block for the private subnet (e.g., 172.16.2.0/24)
- **Option 2: Use Existing VPC (NetworkConfiguration = UseExistingVPC)**
  - Required parameters:
    - ExistingVPCId: ID of your existing VPC
    - ExistingPrivateSubnetId: ID of the private subnet where the instance will be deployed

Storage Configuration

- **Option 1: Create New S3 Bucket (UseExistingS3Bucket = false)**
  - No additional parameters required
  - A new S3 bucket will be created with the naming pattern: pydocstrgen-{account-id}-{region}-{unique-id}
- **Option 2: Use Existing S3 Bucket (UseExistingS3Bucket = true)**
  - Required parameters:

- S3BucketName: Name of your existing S3 bucket

Optional Parameters

- InstanceType: EC2 instance type (default: t3.medium)
- RootVolumeSize: Size of the root EBS volume in GiB (default: 20)
- InstanceName: Name tag for the EC2 instance (default: PyDocStrGen-Instance)
- KeyPairName: EC2 Key Pair for SSH access (optional)
- Application: Application tag (default: PyDocStrGen-AI-Agent)
- CostCenter: Cost center tag (default: PyDocStrGen)

Deployment Steps

1. Navigate to AWS CloudFormation in the AWS Management Console
2. Click "Create stack" > "With new resources (standard)"
3. Upload the template file or specify the S3 URL
4. Fill in the required parameters based on your deployment scenario
5. Review and create the stack

Notes

- When creating a new VPC, ensure the CIDR blocks don't overlap with existing networks
- S3 bucket names must be globally unique and follow AWS naming conventions
- The template creates minimal security groups that allow only outbound HTTPS traffic

Appendix B: FTR-Compliant IAM Policy Example

This is an example of the least-privilege policy created by the CloudFormation template for the EC2 instance role.

```
JSON
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "bedrock:InvokeModel",
```

```
        "bedrock:ListFoundationModels",
        "bedrock:InvokeInlineAgent",
        "bedrock:InvokeAgent",
        "bedrock:GetAgent",
        "bedrock:CreateAgent",
        "bedrock:UpdateAgent",
        "bedrock:GetFoundationModel",
        "bedrock:InvokeModelWithResponseStream",
        "bedrock:GetInferenceProfile",
        "ssm:StartSession",
        "ssm:SendCommand",
        "ssm:GetParameters",
        "logs:CreateLogGroup",
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
    {
      "Action": [
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::{SourceS3Bucket}"
      ],
      "Effect": "Allow"
    },
    {
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::{SourceS3Bucket}/*"
      ],
      "Effect": "Allow"
    },
```

```
    {
      "Condition": {
        "Bool": {
          "aws:SecureTransport": "true"
        }
      },
      "Action": [
        "s3:PutObject"
      ],
      "Resource": [
        "arn:aws:s3:::{SourceS3Bucket}/*"
      ],
      "Effect": "Allow"
    }
  ]
}
```

Appendix C: Bucket Policy

```
{
   "Version": "2008-10-17",
   "Statement": [
     {
       "Sid": "DenyInsecureConnections",
       "Effect": "Deny",
       "Principal": "*",
       "Action": "s3:*",
       "Resource": [
         "arn:aws:s3::: {SourceS3Bucket} ",
         "arn:aws:s3::: {SourceS3Bucket} /*"
       ],
       "Condition": {
         "Bool": {
           "aws:SecureTransport": "false"
         }
       }
     }
   ]
}
```