TEAM-PRAVAHA

[simple crisp docs]

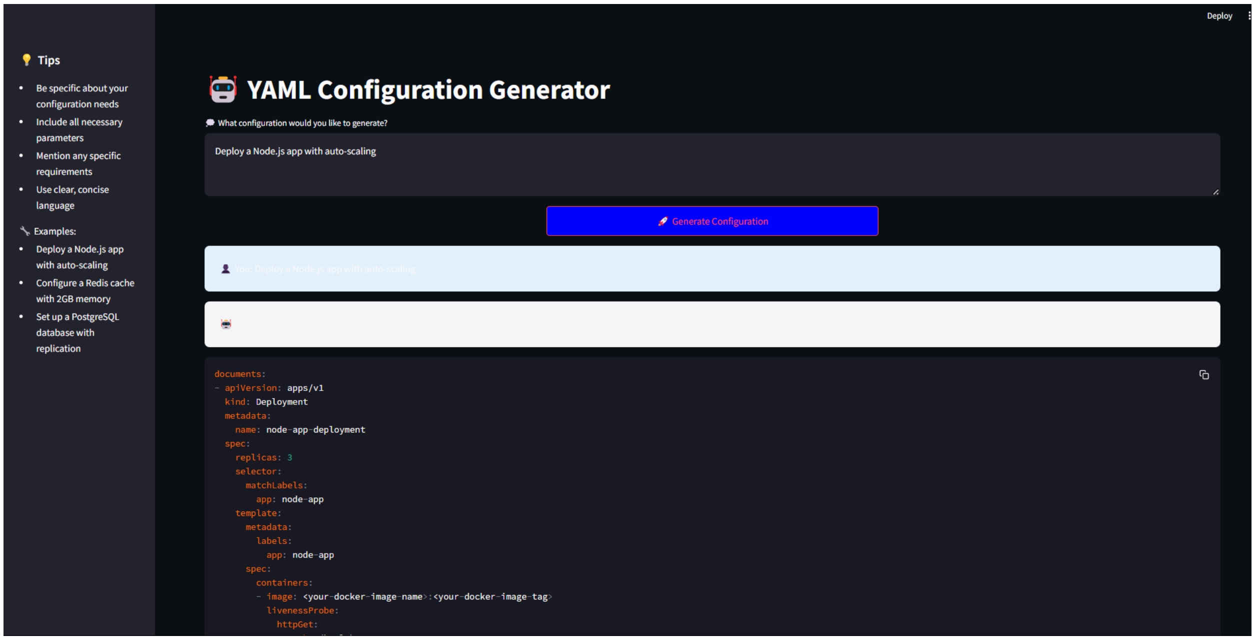
YAML Configuration Generator

A conversational interface for generating Spheron Infrastructure Composition Language (ICL) configurations.

Overview

YAML Configuration Generator simplifies the process of creating Spheron ICL configurations by letting developers describe what they want in plain English. No more memorizing YAML syntax or struggling with configuration structures – just tell the bot what you need, and it generates valid YAML instantly.

DEMO



💡 Tips

- Be specific about your configuration needs
- Include all necessary parameters
- Mention any specific requirements
- Use clear, concise language

🔧 Examples:

- Deploy a Node.js app with auto-scaling
- Configure a Redis cache with 2GB memory
- Set up a PostgreSQL database with replication

YAML Configuration Generator

💭 What configuration would you like to generate?

Deploy a Node.js app with auto-scaling

👤 You: Deploy a Node.js app with auto-scaling

```
documents:
- apiVersion: apps/v1
  kind: Deployment
  metadata:
    name: node-app-deployment
  spec:
    replicas: 3
    selector:
      matchLabels:
        app: node-app
    template:
      metadata:
        labels:
          app: node-app
      spec:
        containers:
        - image: <your-docker-image-name>:<your-docker-image-tag>
          livenessProbe:
            httpGet:
```

Key Features

- **Natural Language Input**: Simply describe your infrastructure needs in plain English
- **Instant YAML Generation**: Get valid Spheron ICL configurations in seconds
- **One-Click Downloads**: Export your configurations directly as YAML files
- **Interactive Refinement**: Easily modify configurations through conversation

Getting Started

Installation

bash

Copy

```
# Install required packages
pip install streamlit google-generativeai pypdf pyyaml

# Run the application
streamlit run app.py
```

Basic Usage

1. Enter your infrastructure requirements in plain English
2. Click "Generate Configuration" to get your YAML
3. Download the generated configuration or refine it further

Example Prompts

- "Deploy a Node.js app with 2GB RAM and auto-scaling"
- "Set up a Python service with Redis cache and PostgreSQL"
- "Configure a static website with CDN and custom domain"

Configuration Examples

Basic Node.js Deployment

yaml

Copy

```yaml
# User prompt: "Deploy a Node.js app with 2GB RAM"
compute:
  type: "container"
  runtime: "nodejs"
  resources:
    memory: "2GB"
  scaling:
    min: 1
    max: 1
```

Auto-scaling Service

yaml

Copy

```yaml
# User prompt: "I need an auto-scaling Node.js service with min 2 and max 5 instances"
compute:
  type: "container"
  runtime: "nodejs"
  scaling:
    min: 2
    max: 5
  resources:
    memory: "1GB"
    cpu: "1x"
```

Best Practices

1. **Be Specific**: Include key details like memory, CPU, or scaling requirements
2. **Use Plain Language**: No need for technical YAML terminology
3. **Iterate Gradually**: Start simple and add complexity through conversation

Common Use Cases

- Deploying web applications
- Setting up microservices
- Configuring databases and caches
- Managing static websites
- Setting up development environments

Limitations

- Complex networking configurations may need manual refinement
- Custom scripts or init commands might require specific syntax
- Some advanced features may need direct YAML editing