# AI Developer API Gateway by Code Creator

## Customer Launch and First-Use Instructions

Ubuntu 24.04.4 LTS | LiteLLM, PostgreSQL, Redis, Docker, and Nginx

**Built for a clean first launch.** The server creates fresh administrator and API credentials on its first boot, waits for LiteLLM readiness, then writes the instance-specific FIRST_LOGIN.txt file with the correct public-IP URLs.

| At a glance | What you get |
|---|---|
| Web and API front door | Nginx on TCP 80 with the LiteLLM Admin UI, API documentation, and OpenAI-compatible API base. |
| Private runtime services | LiteLLM on 127.0.0.1:4000 with PostgreSQL and Redis available only inside the Docker network. |
| Secure onboarding | Fresh administrative credentials and a gateway API key are generated for every new instance launch. |
| Your provider accounts | No OpenAI, Anthropic, AWS Bedrock, Azure, or other provider key is included. Add your own after launch. |

# 1. Product overview

AI Developer API Gateway by Code Creator is a ready-to-launch, self-hosted AI API control plane for developers, teams, agencies, and businesses. It centralizes OpenAI-compatible AI requests behind an AWS instance you control, with LiteLLM for provider routing, virtual-key management, spend-aware workflows, and developer integrations.

The product is designed for application integration rather than end-user chat. Use it to give applications one API base URL, issue separate virtual keys, connect supported model providers, and manage access through the LiteLLM Admin UI.

| Component | Included configuration |
|---|---|
| Operating system | Ubuntu 24.04.4 LTS |
| Default Linux user | ubuntu |
| AI API control plane | LiteLLM v1.83.10 behind Nginx |
| Web / API access | Nginx on TCP 80 |
| Internal LiteLLM listener | 127.0.0.1:4000 |
| Persistent services | PostgreSQL and Redis Docker containers |
| Customer first boot | Fresh credentials, public-IP instructions, health-aware completion, and baked container images |

**Provider keys are optional at launch.** The Admin UI, API authentication, virtual-key management, status checks, and local examples are available without an external provider key. A provider key is required only before a model request can reach that provider.

# 2. AWS security group requirements

Open only the inbound ports required for your administrators and applications.

| Port | Use | Recommended source |
|---|---|---|
| TCP 22 | SSH administration | Your trusted administrator public IP or corporate CIDR only. |
| TCP 80 | Landing page, Admin UI, API documentation, and OpenAI-compatible API | Your application networks, administrator IPs, or another intentionally controlled CIDR. |

**Do not expose internal services.** Do not open TCP 4000, TCP 5432, or TCP 6379 in the AWS security group. LiteLLM is intentionally bound to localhost and PostgreSQL/Redis are internal Docker services.

## 3. Launch the instance and wait for first boot

1. Launch the AMI with a public IPv4 address enabled and an EC2 key pair selected. Use the EC2 key pair selected at launch to sign in as **ubuntu**
2. Allow several minutes for first boot. The server creates fresh credentials, starts PostgreSQL, Redis, and LiteLLM, waits for LiteLLM readiness, and then generates FIRST_LOGIN.txt.
3. Connect with your selected key pair using the ubuntu user.

```
ssh -i /path/to/your-key.pem ubuntu@YOUR_PUBLIC_IP
```

**First-boot completion check.** Do not treat a temporary 502 response during the first few minutes as a product failure. Check that first boot has finished before troubleshooting the web interface.

```
sudo systemctl status codecreator-ai-gateway-firstboot.service --no-pager

# Ready when this file exists:
test -f /var/lib/codecreator-ai-gateway-firstboot.done && echo "First boot complete"
```

## 4. Read your instance-specific first-login guide

After first boot completes, open the generated first-login guide. It contains the current public-IP URLs, generated administrator credentials, gateway API key, management commands, and security reminders.

```
cat /home/ubuntu/FIRST_LOGIN.txt
```

| Helper command | Purpose |
|---|---|
| codecreator-ai-gateway-url | Show current public-IP URLs. |
| codecreator-ai-gateway-credentials | Show the generated Admin UI credentials and gateway API key. |
| codecreator-ai-gateway-status | Show service, container, and local health information. |
| codecreator-ai-gateway-logs | Show product logs. |
| codecreator-ai-gateway-restart | Restart the gateway stack and show container state. |
| codecreator-ai-gateway-examples | Show the location of included examples. |
| codecreator-ai-gateway-set-openai-key | Store your OpenAI provider API key and restart LiteLLM. |

## 5. Open the web interface and API documentation

| Destination | Address |
|---|---|
| Landing page | http://YOUR_PUBLIC_IP/ |
| LiteLLM Admin UI | http://YOUR_PUBLIC_IP/ui/ |
| API documentation | http://YOUR_PUBLIC_IP/docs |
| OpenAI-compatible API base | http://YOUR_PUBLIC_IP/v1 |

Replace **YOUR_PUBLIC_IP** with the public IPv4 address shown by codecreator-ai-gateway-url or in FIRST_LOGIN.txt.

- Open the LiteLLM Admin UI at /ui/.
- Use the administrator username and password shown by codecreator-ai-gateway-credentials.
- After signing in, review models, virtual keys, usage information, and gateway settings.

**Production recommendation.** This AMI is designed to work with a public IP so a domain is not required. For an internet-facing production deployment, add a domain and HTTPS certificate through your preferred Nginx, load balancer, or reverse-proxy approach.

## 6. Add your provider key when you are ready to make model calls

The AMI does not include provider credentials. Add your own OpenAI key only when you are ready to send model requests through OpenAI. Keep provider keys out of tickets, screenshots, shared terminal history, and application source code.

```
codecreator-ai-gateway-set-openai-key
```

The helper prompts for the key, stores it in the local gateway environment, and recreates LiteLLM so the key is loaded. The same principle applies when you later configure other supported providers through your own LiteLLM configuration process.

## 7. Test the OpenAI-compatible API

The included examples use the local API endpoint from inside the instance. Model list requests authenticate with your generated gateway API key. A provider-backed chat request also requires a valid provider key.

```
/opt/codecreator-ai-gateway/examples/curl-models.sh
/opt/codecreator-ai-gateway/examples/curl-chat-openai-compatible.sh
python3 /opt/codecreator-ai-gateway/examples/python-openai-compatible-client.py
```

Applications outside the instance should use the public API base URL and a gateway API key or a virtual key:

```
Base URL:       http://YOUR_PUBLIC_IP/v1
Authorization:  Bearer YOUR_GATEWAY_OR_VIRTUAL_KEY
```

**API behavior.** A request to /v1/models without a gateway key correctly returns HTTP 401. This confirms that the API is not anonymously available.

## 8. Create virtual keys for applications and teams

Virtual keys let you give each application, developer, or team a separate credential instead of sharing the master gateway key. You can use virtual keys to scope model access and apply budgets or rate limits where supported by the configured LiteLLM features.

1. Open the Admin UI at /ui/ and sign in with your generated administrator credentials.
2. Open Virtual Keys or Keys in the left navigation.
3. Choose Create New Key, give it a meaningful name or alias, and configure optional model, budget, expiration, or rate-limit settings.

4. Copy the key at creation time and store it securely in your application secret store.
5. Revoke or delete keys that are no longer needed.

## 9. Basic operations and troubleshooting

| What you need | Command or action |
|---|---|
| Check product health | codecreator-ai-gateway-status |
| Check first-boot progress | sudo systemctl status codecreator-ai-gateway-firstboot.service --no-pager |
| Check local LiteLLM readiness | curl -s http://127.0.0.1:4000/health/readiness \| jq |
| View container state | cd /opt/codecreator-ai-gateway && sudo docker compose ps |
| View logs | codecreator-ai-gateway-logs 200 |
| Restart stack | codecreator-ai-gateway-restart |

### Common situations

- Admin UI, docs, or API returns 502 shortly after launch: first boot may still be waiting for LiteLLM readiness. Check the first-boot service and wait until FIRST_LOGIN.txt exists.
- You cannot connect by SSH: verify TCP 22 is allowed from your current public IP and that you selected the correct EC2 key pair.
- You cannot open the web page: verify the instance has a public IPv4 address and TCP 80 is allowed from your testing or application network.
- A model request fails: verify you added a valid provider key, the requested model is configured, and the provider account has the required billing and access.

## 10. Security practices

- Restrict TCP 22 to trusted administrator IP addresses or your corporate network. Use the EC2 key pair selected at launch to sign in as **ubuntu.**
- Restrict TCP 80 to the intended administrator and application networks wherever possible. Do not open internal ports 4000, 5432, or 6379.
- Treat the generated administrator password, gateway master key, and each virtual key as secrets. Store them in an approved secret manager and rotate or revoke keys when access changes.
- Add HTTPS before using this product for internet-facing production application traffic. A domain is optional for the initial public-IP setup, but strongly recommended for production use.
- Keep the instance operating system and AMI release current. Version 4.4 removes the unused Telnet client components and does not listen on TCP 23.

## 11. Recommended instance sizes and storage

| Use case | Suggested EC2 size | Storage |
|---|---|---|
| Small testing or proof of concept | t3.large | 40 GB gp3 minimum |
| Recommended default | m6i.large | 60 GB gp3 recommended |
| Heavier team use or higher concurrency | m6i.xlarge or larger | Increase based on log, database, and workload growth |

Provider model use can create additional charges from your selected provider and AWS infrastructure charges still apply. Size the instance based on concurrent API traffic, application demand, and desired operating headroom.

## 12. What is included in version 4.4

- Ubuntu 24.04.4 LTS with current standard package updates applied at release time.
- LiteLLM v1.83.10 with PostgreSQL, Redis, Docker Compose, and Nginx.
- Public-IP onboarding with fresh credentials and instance-specific FIRST_LOGIN.txt instructions.
- First boot that uses baked container images and marks setup complete only after LiteLLM readiness.
- OpenAI-compatible API base, LiteLLM Admin UI, API docs, helper commands, and developer examples.
- Virtual-key lifecycle validated in the Admin UI: create and delete temporary keys.
- CVE-2026-32746 remediation: Telnet components removed and no TCP 23 listener.

## 13. Learning resources and help URLs

Use these official references for deeper product, provider, and AWS operations guidance.

- **LiteLLM documentation:** https://docs.litellm.ai/docs/
- **LiteLLM Proxy / AI Gateway:** https://docs.litellm.ai/docs/simple_proxy
- **LiteLLM Admin UI:** https://docs.litellm.ai/docs/proxy/ui
- **LiteLLM virtual keys:** https://docs.litellm.ai/docs/proxy/virtual_keys
- **LiteLLM budgets and rate limits:** https://docs.litellm.ai/docs/proxy/users
- **OpenAI API developer quickstart:** https://developers.openai.com/api/docs/quickstart
- **Amazon EC2 SSH connection guide:** https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/connect-linux-inst-ssh.html
- **Amazon EC2 security groups:** https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-security-groups.html

## 14. Support information

When requesting support, include the following outputs after removing passwords, API keys, tokens, and provider credentials:

```
codecreator-ai-gateway-status
codecreator-ai-gateway-logs 200
cd /opt/codecreator-ai-gateway && sudo docker compose ps
```