

Setup Guide - HyperAccel LLM Chatbot

Prerequisite

Before launching, ensure you meet the following requirements:

1. Hugging Face Account

- You must have an active Hugging Face account. You can create one at <https://huggingface.co/join>.

2. Hugging Face Access Token

- Obtain a personal access token from Hugging Face. This token is required to download and use models from the Hugging Face Hub.
- Generate a token at <https://huggingface.co/settings/tokens>

3. Model Access Permissions (for gated models)

- If you plan to use gated models such as LLaMA, HyperCLOVA X, ensure that your Hugging Face account has been granted permission to access them.
- Visit the model's page on Hugging Face and request access if needed.

Quick Start

1. Launch an F2 Instance

Use the following configuration to launch your EC2 instance:

- **Region:** us-east-1[US East (N. Virginia)], us-west-2[US West (Oregon)], eu-west-2[Europe (London)], ap-southeast-2[Asia Pacific (Sydney)]
- **AMI ID:**
select AMI ID according to your region

Region	AWS Name	AMI ID
us-east-1	US East (N. Virginia)	ami-0c4af4010f9239288

Region	AWS Name	AMI ID
us-west-2	US West (Oregon)	ami-05dcc91b7421149d4
ap-southeast-2	Asia Pacific (Sydney)	ami-0c7a04a703f31254d
eu-west-2	Europe (London)	ami-0e61771b9f0775d55

us-east-1 example

AMI from catalog
Recents
Quick Start

Name
ami-hyperaccel-f2-release-251014-2-prod-t2o2twvacnudy Verified provider

Description
-

Image ID
ami-0c4af4010f9239288

Username ⓘ
root (Check with the AMI provider.)

Catalog	Published	Architecture	Virtualization	Root device type	ENA Enabled
AWS Marketplace AMIs	2025-10-17T12:18:39.000Z	x86_64	hvm	ebs	Yes

If you have an existing license entitlement to use this software, then you can launch this software without creating a new subscription. If you do not have an existing entitlement, then by launching this software, you will be subscribed to this software and agree that your use of this software is subject to the pricing terms and the seller's [End User License Agreement](#)

Search

[Browse more AMIs](#)

Including AMIs from AWS, Marketplace and the Community

- **Instance Type: f2.6xlarge**

▼ Instance type [Info](#) | [Get advice](#)

Instance type

f2.6xlarge

Family: f2 24 vCPU 256 GiB Memory Current generation: true

On-Demand RHEL base pricing: 2.2392 USD per Hour

On-Demand Ubuntu Pro base pricing: 2.022 USD per Hour

On-Demand Linux base pricing: 1.98 USD per Hour On-Demand SUSE base pricing: 2.105 USD per Hour

▼

All generations

[Compare instance types](#)

Additional costs apply for AMIs with pre-installed software

After launching the instance, check the following inbound rules in the security tab:

- **Port 22** (TCP protocol): SSH access
- ***Port 5173** (**TCP protocol): Chat UI frontend server access
- **Source - 0.0.0.0/0** (applies to all ports, allowing access from any IP address)

Go to instances and select Security

Check if Port 5173 is open

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source	Description
-	sgr-0c8b9c1094191d3f2	IPv4	SSH	TCP	22	0.0.0.0/0	-
-	sgr-0d0a2b1b7eeb8abca	IPv4	Custom TCP	TCP	5173	0.0.0.0/0	-

if Port 5173 is missing, go to Troubleshooting B. Port 5173 is not found

2. Run Backend Server (vLLM)

Notice : Please proceed following steps in a *new terminal*

run the following commands step by step:

Launch the Docker container

```
>>> set_container
```

Set Huggingface token

```
>>> export HF_TOKEN="<huggingface_token>"
```

Activate the virtual environment

```
>>> cd /workspace/dev/Projects/vllm-orion  
>>> source .venv/bin/activate
```

Set environment variables for F2

```
>>> set-f2-env
```

Load the FPGA bitstream

```
>>> load-f2-bitstream
```

Start vllm server

(For example, use: `vllm serve LGAI-EXAONE/EXAONE-3.5-2.4B-Instruct --config config/config.yaml`)

Notice : model setup takes a few minutes

```
>>> vllm serve <huggingface_model_name> --config config/config.yaml
```

3. Run Frontend Server (Chat UI)

*Notice : You need to proceed this step in a **new terminal***

```
# This step doesn't need to use docker container.
```

```
>>> cd ~/Projects/HyperDex-Container/hyperaccel-chat-ui  
>>> npm run dev -- --host          # Start the frontend server
```

Functional Test

1. Backend Server Only

Notice : You need to proceed this step in a *new terminal*

To verify the backend is working properly, run the following test command in your terminal:

- Test command:

```
>>> curl <http://localhost:8000/v1/completions> \\  
-H "Content-Type: application/json" \\  
-d '{"model": "LGAI-EXAONE/EXAONE-3.5-2.4B-Instruct", "prompt": "Who are you?", "max_tokens": 30, "temperature": 1 }'
```

- Expected output:

```
{"id":"cmpl-AI-EXAONE/EXAONE-3.5-2.4B-Instruct","choices":[{"index":0,"text":"\n\n  
Options:\n- a scientist\n- a farmer\n- you\n- athlete\n- a schoolbo  
y/","logprobs":null,"finish_reason":"length","stop_reason":null,"prompt_logp  
robs":null}], "usage":{"prompt_tokens":4,"total_tokens":34,"completion_toke  
ns":30,"prompt_tokens_details":null},"kv_transfer_params":null}
```

2. Frontend Server Connection Test

To verify frontend webpage is connected to server, check if we can reach to port 5173

- Test Command

```
nc -zv <F2_instance_public_IPv4_address> 5173
```

- Expected output

```
Connection to <F2_instance_public_IPv4_address> 5173 port [tcp/*] su  
cceded!
```

3. Backend + Frontend Server Test

To test both backend and frontend servers together:

1. Open the Chat UI in a web browser:

```
http://<F2_instance_public_IPv4_address>:5173/
```

2. In the Chat UI, enter a prompt into the "Ask anything" input box at the bottom of the screen, and press Enter.

- Example Input:

```
Who are you?
```

- Expected Output:

```
I am an artificial intelligence designed to assist and communicate with users like yourself. My goal is to provide helpful information, engage in conversations, and support tasks across a wide range of topics. How may I assist you today? If you have any specific questions or need information on particular subjects, feel free to ask!
```

How to Change Large Language Models?

To switch to a different Hugging Face model, follow these steps:

1. When starting the backend server, replace <huggingface_model_name> with the new model you want to use.

```
>>> vllm serve <huggingface_model_name> --config config/config.yaml
```

2. Edit the .env.local file in the Chat UI project directory and change the "name" field to match the new model.

```
# File: ~/Projects/HyperDex-Container/hyperaccel-chat-ui/.env.local

MODELS=[
  {
    "name": "<huggingface_model_name>",
    "parameters": {
      "temperature": 1.0,
```

```
"truncate_prompt_tokens": 256
},
"endpoints": [
  {
    "baseURL": "<http://127.0.0.1:8000/v1>",
    "type": "openai"
  }
]
}
```

Replace `<huggingface_model_name>` with the name of your new model.

3. After updating the configuration, restart the Chat UI server:

```
>>> cd ~/Projects/HyperDex-Container/hyperaccel-chat-ui
>>> npm run dev -- --host
```

Now your Chat UI is connected to the new model.

Supported Large Language Models

(Replace `<huggingface_model_name>` with one of the following models)

You can use any of the models listed below by replacing `<huggingface_model_name>` in the command or configuration files:

- meta-llama/Llama-3.1-8B-Instruct
- meta-llama/Llama-3.2-1B-Instruct
- meta-llama/Llama-3.2-3B-Instruct
- naver-hyperclova/HyperCLOVAX-SEED-Text-Instruct-0.5B
- naver-hyperclova/HyperCLOVAX-SEED-Text-Instruct-1.5B
- LGAI-EXAONE/EXAONE-3.5-2.4B-Instruct
- LGAI-EXAONE/EXAONE-3.5-7.8B-Instruct

Be sure to use the full model name exactly as shown above when configuring your backend and frontend settings.

Troubleshooting

A. vLLM server failure

- If the vLLM server fails to initialize, please re-upload the F2 bitstream and restart the vLLM server.

```
>>> load-f2-bitstream # Load the FPGA bitstream
>>> vllm serve <huggingface_model_name> --config config/config.yaml
```

- If you don't have access to the model on Hugging Face, please check whether your account (with the issued token) has been granted permission to access the model. Especially, LLaMA models and HyperCLOVA X models are gated models and require access permissions.

B. Port 5173 is not found

The screenshot shows the AWS IAM console for a security group named 'sg-0b477faeac739dafc'. The 'Inbound rules' tab is selected, showing a table with one rule. The rule is for SSH (Type) over TCP (Protocol) on port 22 (Port range), with source 0.0.0.0/0 (Source). The 'Edit inbound rules' button is visible in the top right of the table area.

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source	Description
-	sgr-0c8b9c1094191d3f2	IPv4	SSH	TCP	22	0.0.0.0/0	-

Select Edit inbound rules

This screenshot is identical to the previous one, but the 'Edit inbound rules' button in the top right of the table area is highlighted with a red rectangle.

Click Add rule

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID: sgr-0c8b9c1094191d3f2

Type	Protocol	Port range	Source	Description - optional
SSH	TCP	22	Custom	Q

[Add rule](#)

Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

[Cancel](#) [Preview changes](#) [Save rules](#)

Set Port range as 5173 and source as 0.0.0.0/0

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID: sgr-0c8b9c1094191d3f2

Type	Protocol	Port range	Source	Description - optional
SSH	TCP	22	Custom	Q
Custom TCP	TCP	5173	Anywhere...	Q 0.0.0.0/0

[Add rule](#)

Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

[Cancel](#) [Preview changes](#) [Save rules](#)

Save rules to save added rule

Edit inbound rules Info

Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID: sgr-0c8b9c1094191d3f2

Type	Protocol	Port range	Source	Description - optional
SSH	TCP	22	Custom	Q
Custom TCP	TCP	5173	Anywhere...	Q 0.0.0.0/0

[Add rule](#)

Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

[Cancel](#) [Preview changes](#) [Save rules](#)

Now we have opened port 5173

sg-0b477faec739dafc - LLM Chatbot on AWS F2 Instance-v1.4.4-AutogenByAWSMP--4 Actions

Details

Security group name LLM Chatbot on AWS F2 Instance-v1.4.4-AutogenByAWSMP--4	Security group ID sg-0b477faec739dafc	Description LLM Chatbot on AWS F2 Instance-v1.4.4-AutogenByAWSMP--4 created 2025-11-06T08:38:53.076Z	VPC ID vpc-06905740295fe6452
Owner 292195077655	Inbound rules count 2 Permission entries	Outbound rules count 1 Permission entry	

Inbound rules | Outbound rules | Sharing - new | VPC associations - new | Tags

Inbound rules (2) Manage tags Edit inbound rules

Search

<input type="checkbox"/>	Name	Security group rule ID	IP version	Type	Protocol	Port range
<input type="checkbox"/>	-	sgr-0c8b9c1094191d3f2	IPv4	SSH	TCP	22
<input type="checkbox"/>	-	sgr-02af6b0068136ab6a	IPv4	Custom TCP	TCP	5173

C. MongoDB failure

Our chatbot relies on MongoDB to store chat logs and user data. Although MongoDB should automatically start and connect to the chat-ui, this behavior can vary depending on the environment. If the connection fails, follow the steps below to resolve the issue.

example of MongoDB docker fail log at chat-ui

```
VITE v6.3.5 ready in 1167 ms
+ Local: http://localhost:5173/
+ Network: http://172.31.40.75:5173/
+ press h + enter to show help
[08:19:19.218] ERROR (8406): Connection error
err: {
  "type": "MongoServerSelectionError",
  "message": "connect ECONNREFUSED 127.0.0.1:27017",
  "stack":
    MongoServerSelectionError: connect ECONNREFUSED 127.0.0.1:27017
      at Timeout._onTimeout (/home/ubuntu/Projects/HyperDex-Container/hyperaccel-chat-ui/node_modules/mongodb/lib/sdam/topology.js:278:38)
      at listOnTimeout (node:internal/timers:581:17)
      at process.processTimers (node:internal/timers:519:7)
  "reason": {
    "type": "Unknown",
    "servers": {},
    "stale": false,
    "compatible": true,
    "heartbeatFrequencyMS": 10000,
    "localThresholdMS": 15,
    "setName": null,
    "maxElectionId": null,
    "maxSetVersion": null,
    "commonWireVersion": 0,
    "logicalSessionTimeoutMinutes": null
  }
}
```

Check if MongoDB docker exists

```
$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS
```

```

PORTS  NAMES
$ docker ps
CONTAINER ID  IMAGE          COMMAND          CREATED    STATU
S          PORTS          NAMES
bdac7cd4a431  mongo:latest  "docker-entrypoin...  3 months ago  E
xited (128) 3 days ago          mongo-chatui

```

Restart the MongoDB Docker container and verify that it is running

```

$ docker restart mongo-chatui
$ docker ps
CONTAINER ID  IMAGE          COMMAND          CREATED    STATUS
PORTS  NAMES
1b62dbfc1d1e  mongo:latest  "docker-entrypoin...  28 minutes ago  Up
28 minutes  0.0.0.0:27017→27017/tcp, [::]:27017→27017/tcp  mongo-chatu
i

```

Once the container is running, please try the Quick Start steps again.