aws

COMMUNITY DAY

Zaragoza 2025

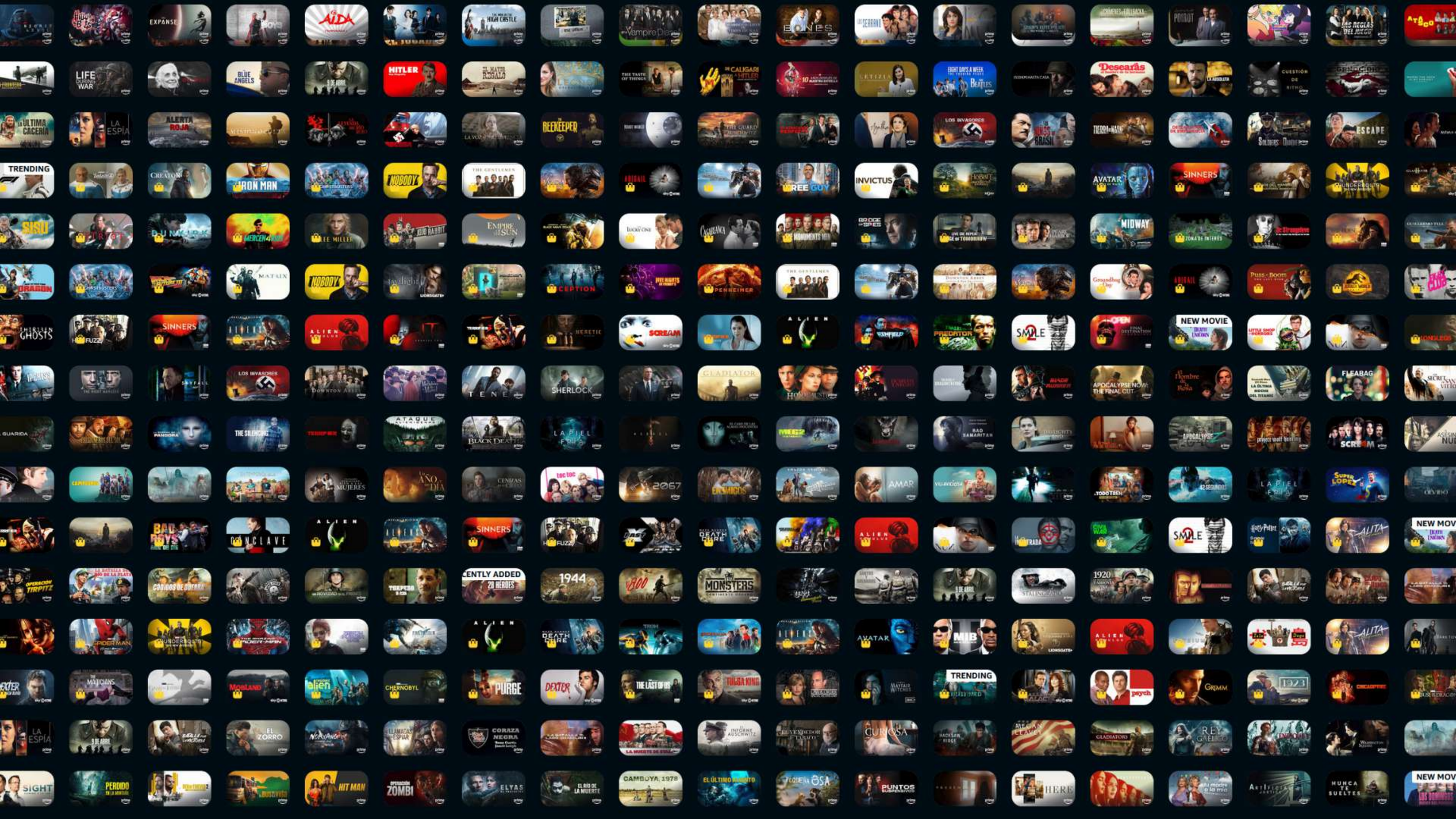
Spain

javi moreno

NTT DATA

UPC

free time!


EXPANSE
AIDA
HIGH CASTLE
BONES
POIROT
LIFE DURING WAR
BLUE ANGELS
HITLER
Desearás
LA ÚLTIMA CACERÍA
LA ESPÍA
ALERTA ROJA
BEEKEEPER
LOS INVASORES
ESCAPE
TRENDING
CREATOR
IRON MAN
NOBODY
INVICTUS
AVATAR
SINNERS
SISU
DUNKIRK
MIDWAY
DRAGON
MATRIX
NOBODY
CEPTION
OPENHEIMER
THE GHOSTS
SINNERS
SCREAM
ALIEN
PREDATOR
NEW MOVIE
FLEABAG
LOS INVASORES
SHERLOCK
GLADIATOR
APOCALYPSE NOW THE FINAL CUT
THE SILENCING
BLACK DEATH
LA PIEL FRÍA
SCREAM
2067
AMAR
SUPER LOPEZ
BAD BOYS
ENCLAVE
ALIEN
SINNERS
DEATH CURE
ALITA
NEW MOV
OPERACIÓN TIRPITZ
RECENTLY ADDED
28 HÉROES
1944
MONSTERS
1920
SPIDER-MAN
ALIEN
DEATH CURE
AVATAR
MIB
ALITA
THE MAGICIANS
MORLAND
CHERNOBYL
PURGE
DEXTER
THE LAST OF US
TULSA KING
TRENDING
psych
GRIMM
1923
LA ESPÍA
EL ZORRO
CORAZA NEGRA
GURKOSA
GLADIATORS
REY GÁLICO
SIGHT
PERDIDO
HIT MAN
OPERACIÓN ZOMBI
ELYAS
CAMBOYA, 1978
PEQUEÑA OSA
PUNTOS SUSPENSIVOS
HERE
ARTIFICIAL JUSTICE
NUNCA TE SUELTES
NEW MOV

modelos, agentes, mcps y javascript ¿qué puede salir mal?

demo!

dance course

Whatever my brown balls say

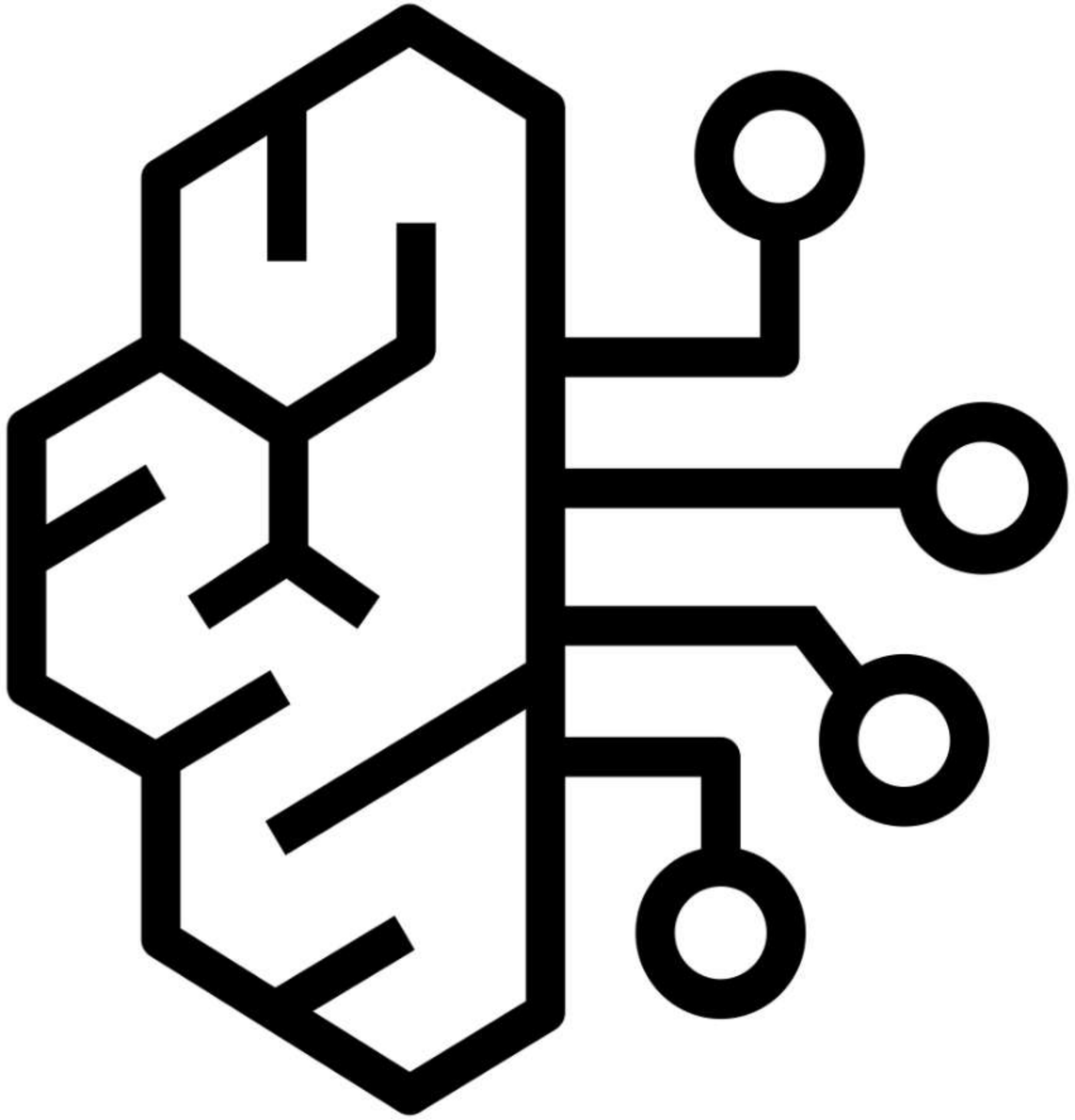
Jim Smith, Marketing director

Autonomous application that uses generative AI to complete tasks

Takes actions on its own - not just answering questions, but using tools, making decisions, running workflows

Has goals - works toward objectives with minimal human supervision

Uses reasoning - breaks down complex tasks into steps and adapts based on results



amazon bedrock

Amazon Bedrock is a fully managed service offering **serverless** access to foundation models from **Anthropic**, Meta, and Amazon, etc. with no infrastructure management, pay-per-use pricing, and **enterprise-grade** security.

show me the code

langchain

A toolkit that helps you build apps with LLMs without losing your sanity. It handles the boring stuff like connecting to different AI models, managing conversation memory, and hooking up external tools.

the workflow

```
// Workflow for movie recommendation
async function movieWorkflow() {
    const userPreferences = await askUserWhatToWatch();
    const refinedCriteria = await refineCriteriaNode(userPreferences);

    const movies = await movieDiscoveryNode();
    let candidates = [];
    let currentBatch = 0;

    do {
        const evaluated =
            await evaluateMoviesNode(movies, currentBatch, refinedCriteria)
        candidates.push(...evaluated);
        currentBatch++;
    } while (shouldIContinueNode(candidates));

    return await presentRecommendations(candidates);
}
```

the workflow

```
// Workflow for movie recommendation
async function movieWorkflow() {
    const userPreferences = await askUserWhatToWatch();
    const refinedCriteria = await refineCriteriaNode(userPreferences);

    const movies = await movieDiscoveryNode();
    let candidates = [];
    let currentBatch = 0;

    do {
        const evaluated =
            await evaluateMoviesNode(movies, currentBatch, refinedCriteria)
        candidates.push(...evaluated);
        currentBatch++;
    } while (shouldIContinueNode(candidates));

    return await presentRecommendations(candidates);
}
```

the state

It is the **shared memory** or data structure that persists throughout a workflow execution, containing all the information that agents need to access, modify, and pass between each other.

```
// Workflow for movie recommendation
async function movieWorkflow() {
    const userPreferences = await askUserWhatToWatch();
    const refinedCriteria = await refineCriteriaNode(userPreferences);
    const movies = await movieDiscoveryNode();
    let candidates = [];
    let currentBatch = 0;

    do {
        const evaluated =
            await evaluateMoviesNode(movies, currentBatch, refinedCriteria)
        candidates.push(...evaluated);
        currentBatch++;
    } while (shouldIContinueNode(candidates));

    return await presentRecommendations(candidates);
}
```

talking to bedrock

```
const llm = new ChatBedrockConverse({
    model: 'us.anthropic.claude-3-7-sonnet-20250219-v1:0',
    region: region,
    temperature: 0.1, // Low temperature for consistent, structured outputs
});

const prompt = ...
const response = await client.invoke(
    [
        { role: 'user', content: prompt }
    ]
);

return response;
```

the prompt

```
You are an expert movie recommendation analyst.
Analyze the user request and extract structured preferences for finding mov

User Request: "${userInput}"

Please analyze this request and provide structured movie preferences. Consi

1. **Genre Analysis**: What primary genres does the user want?
What related/similar genres might they enjoy?
   - Example: "sci-fi" → include "Science Fiction", "Futuristic", etc.
   - Example: "action" → include "Action", "Adventure", etc.
   ...
2. **Exclusions**: What genres or styles should be avoided
based on their preferences?
   - If they hate "cheesy stories" → exclude "Romance Comedy", "Melodrama"
   - If they want "serious" movies → exclude "Slapstick Comedy", "Parody"
   ...

Be intelligent about context and inference. For example, a
49-year-old wanting family movies suggests PG/PG-13 content
```

```
// Workflow for movie recommendation
async function movieWorkflow() {
    const userPreferences = await askUserWhatToWatch();
    const refinedCriteria = await refineCriteriaNode(userPreferences);
    const movies = await movieDiscoveryNode();
    let candidates = [];
    let currentBatch = 0;

    do {
        const evaluated =
            await evaluateMoviesNode(movies, currentBatch, refinedCriteria)
        candidates.push(...evaluated);
        currentBatch++;
    } while (shouldIContinueNode(candidates));

    return await presentRecommendations(candidates);
}
```

"This is a Prime Video page, extract the movie data."

750kb of html

retrieval
augmented
generation

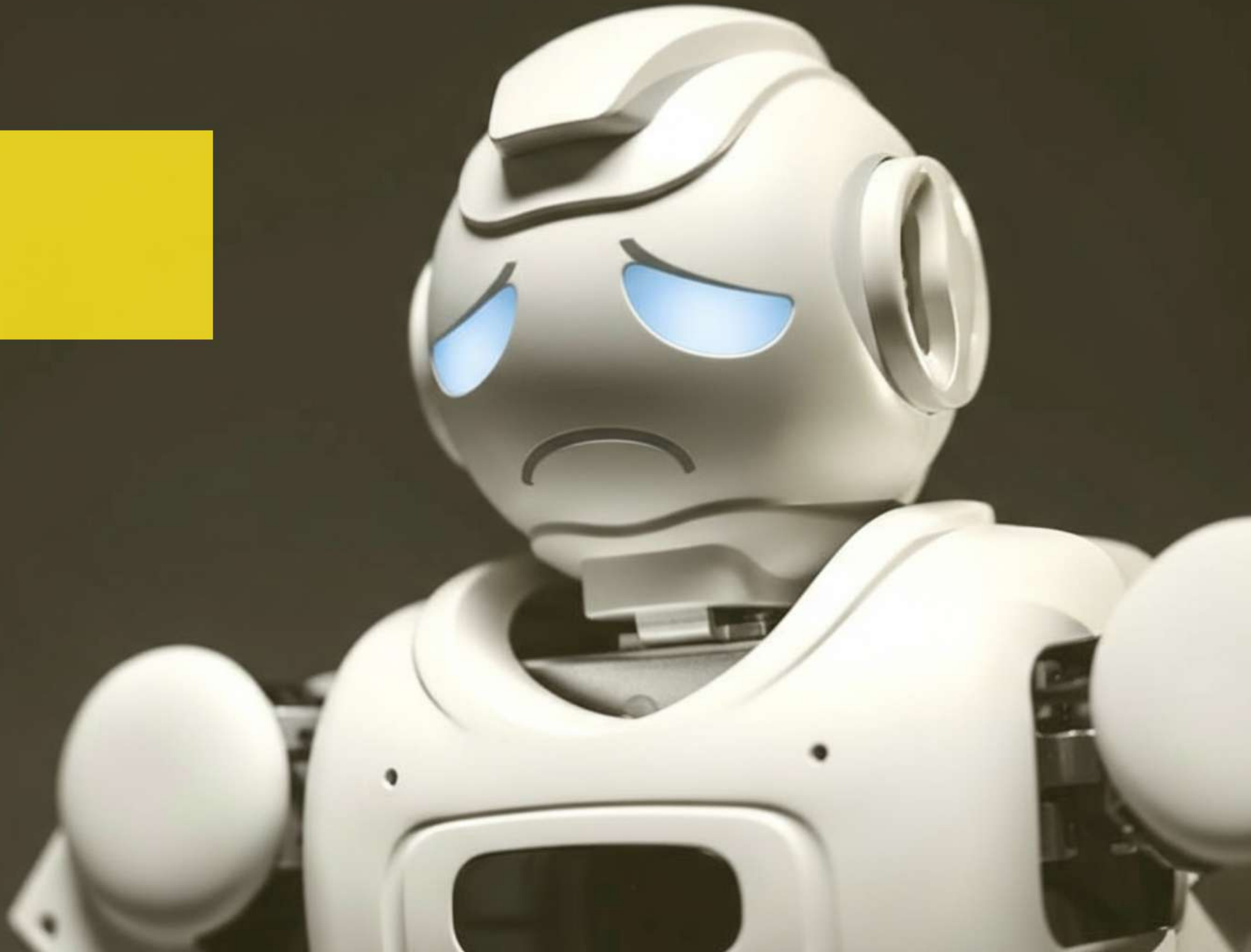
```
import { convert } from 'html-to-text';

function extractRelevantMovieContent(html: string): string {
  const cleanedText = convert(html, {
    selectors: [
      { selector: 'script', format: 'skip' },
      { selector: 'style', format: 'skip' },
      { selector: 'noscript', format: 'skip' }
    ],
  });
  // Limit size to prevent token overuse
  return cleanedText.substring(0, 25000);
}
```

```javascript
// Workflow for movie recommendation
async function movieWorkflow() {
    const userPreferences = await askUserWhatToWatch();
    const refinedCriteria = await refineCriteriaNode(userPreferences);
    const movies = await movieDiscoveryNode();
    let candidates = [];
    let currentBatch = 0;

    do {
        const evaluated =
            await evaluateMoviesNode(movies, currentBatch, refinedCriteria)
        candidates.push(...evaluated);
        currentBatch++;
    } while (shouldIContinueNode(candidates));

    return await presentRecommendations(candidates);
}
```

Hi there,
Let's talk about TMDB

llm can invoke tools, isn't it?

no



it's not the spoon
that bends, it is
only your agent

making some hands for our agent

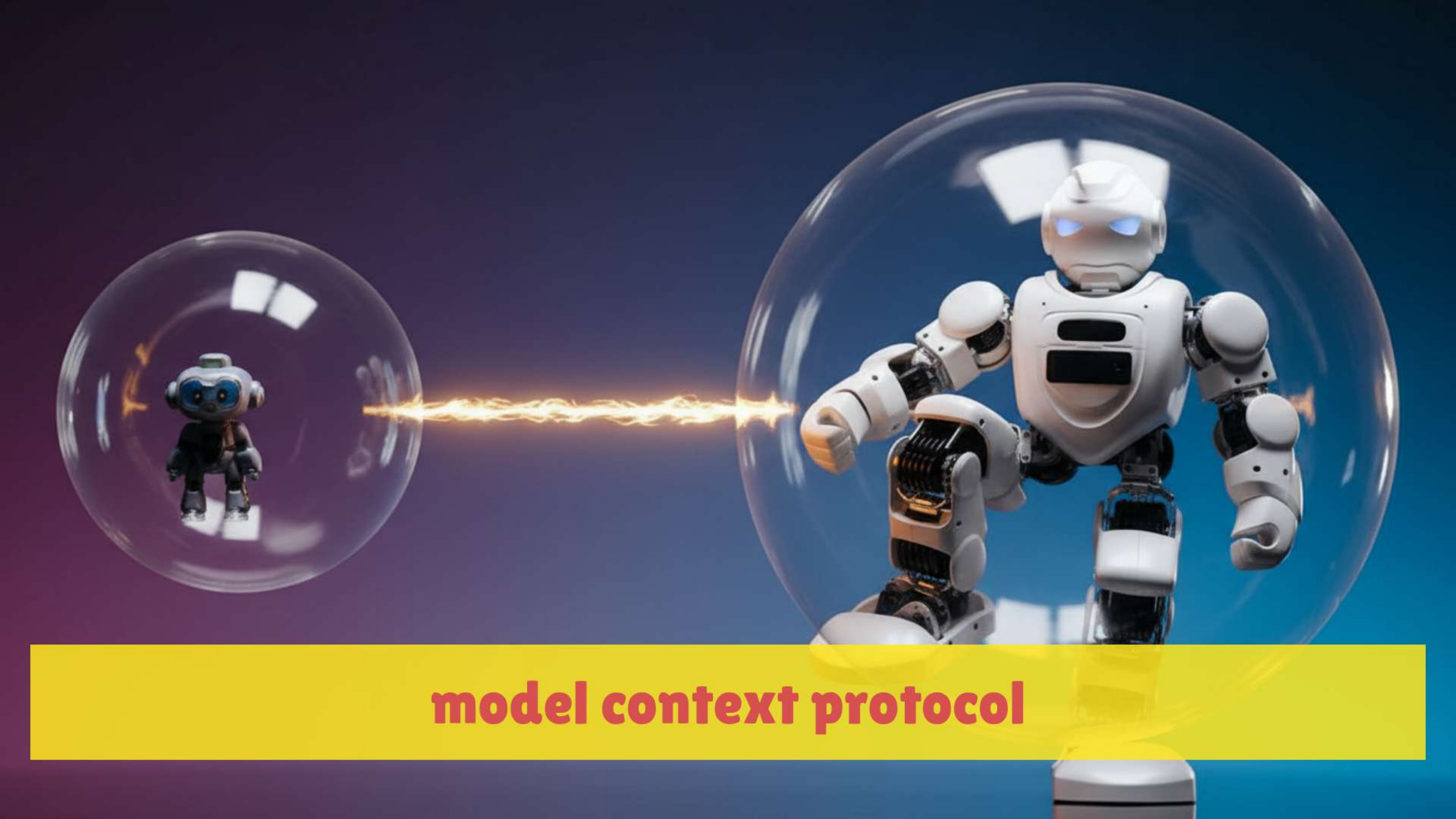
```
function tmdbEnrichmentToolFn() {
    this.tmdb = new TMDB(accessToken);
    const searchResults = await this.tmdb.search.movies({
        query: movie.title,
        year: movie.year,
    });
    ...
}
```

explaining what can it do with its hands

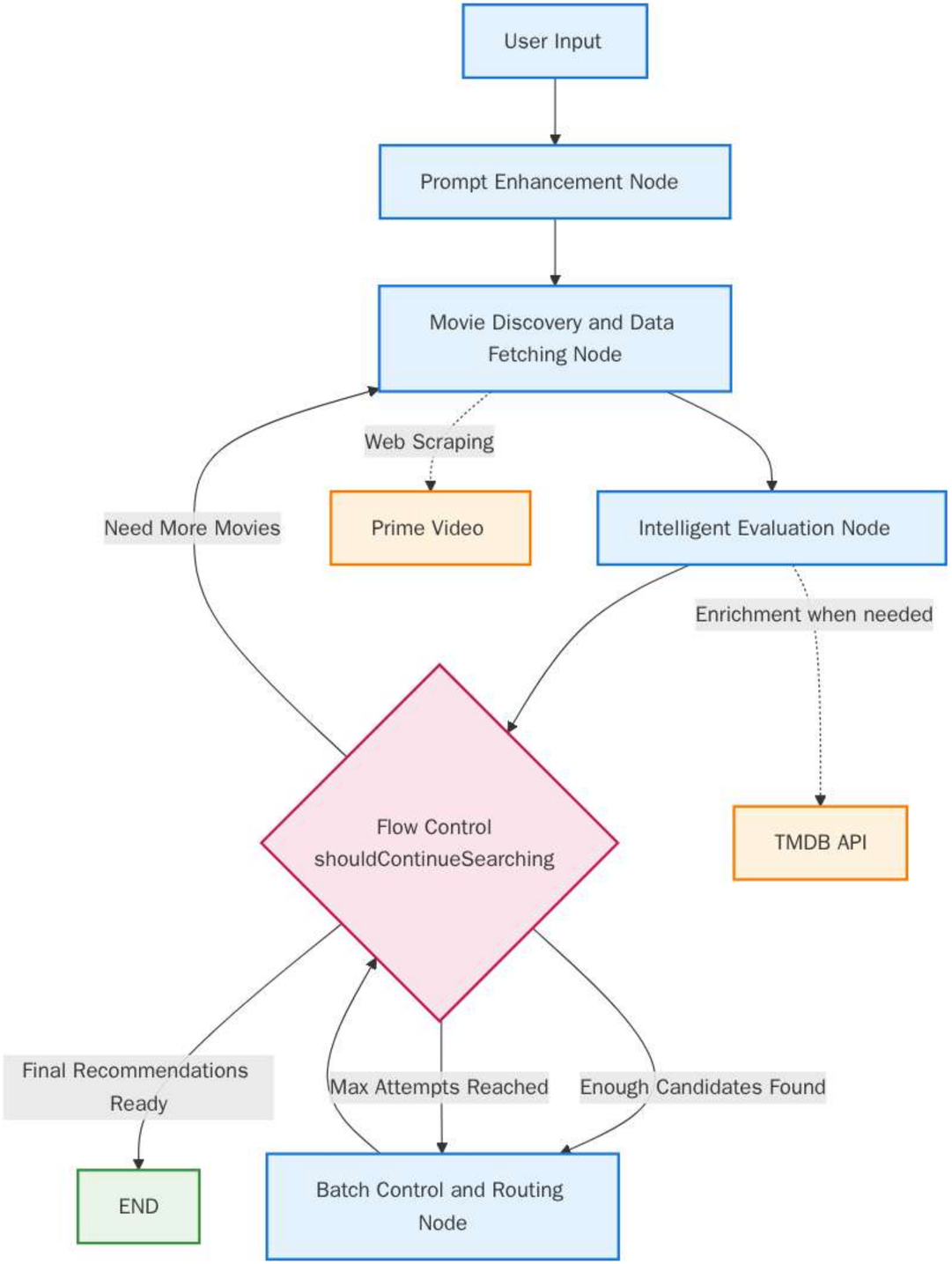
```
const tmdbEnrichmentToolOptions = {
  name: 'tmdb_movie_enrichment',
  description: `Get additional movie information from The Movie
    Database (TMDB) when existing data is insufficient for evaluation.`,
  schema: z.object({
    title: z.string().describe('The movie title to search for'),
    year: z
      .number()
      .optional()
      .describe('The release year of the movie (optional, helps with accura
  }),
};
```

attaching the hands to the agent

```
const tmdbEnrichmentTool = tool(tmdbEnrichmentToolFn, tmdbEnrichmentToolOpt
const agent = createAgent({
  model: new ChatBedrockConverse({
    model: modelId,
    region: process.env.AWS_REGION || 'us-east-1',
    temperature: 0.2,
  }),
  tools: [tmdbEnrichmentTool],
});
const input = `... ...
    Use the TMDB enrichment tool if you need additional information,
    then provide your score evaluation as JSON. ... ...
`;
const response = await agent.invoke({
  messages: [{ role: 'user', content: input }],
});
```


model context protocol

```
// Workflow for movie recommendation
async function movieWorkflow() {
    const userPreferences = await askUserWhatToWatch();
    const refinedCriteria = await refineCriteriaNode(userPreferences);
    const movies = await movieDiscoveryNode();
    let candidates = [];
    let currentBatch = 0;

    do {
        const evaluated =
            await evaluateMoviesNode(movies, currentBatch, refinedCriteria)
        candidates.push(...evaluated);
        currentBatch++;
    } while (shouldIContinueNode(candidates));

    return await presentRecommendations(candidates);
}
```



graphlang

It provides **composable** workflow definition that makes complex AI pipelines maintainable. Also, it gives you built-in **state persistence**, automatic **retry** mechanisms, **visual representation**, error handling, and the ability to pause/resume long-running processes.

the workflow

```
// Workflow for movie recommendation
async function movieWorkflow() {
    const userPreferences = await askUserWhatToWatch();
    const refinedCriteria = await refineCriteriaNode(userPreferences);

    const movies = await movieDiscoveryNode();
    let candidates = [];
    let currentBatch = 0;

    do {
        const evaluated =
            await evaluateMoviesNode(movies, currentBatch, refinedCriteria)
        candidates.push(...evaluated);
        currentBatch++;
    } while (shouldIContinueNode(candidates));

    return await presentRecommendations(candidates);
}
```

the real workflow

```
const videoRecommendationWorkflow = new StateGraph(VideoRecommendationAgent
  .addNode('prompt_enhancement_node', promptEnhancementNode)
  .addNode('movie_discovery_node', movieDiscoveryNode)
  .addNode('intelligent_evaluation_node', intelligentEvaluationNode)
  .addNode('batch_control_and_routing_node', batchRoutingNode)

  .addEdge(START, 'prompt_enhancement_node')
  .addEdge('prompt_enhancement_node', 'movie_discovery_node')
  .addEdge('movie_discovery_node', 'intelligent_evaluation_node')

  .addConditionalEdges('intelligent_evaluation_node', shouldContinueSearchi
  .addConditionalEdges('batch_control_and_routing_node', shouldContinueSear
```

demo?

eMule

rules after midnight

rules after midnight

Set a a maximum execution **time**

rules after midnight

Set a a maximum execution **time**

Provide excellent **explainability**

rules after midnight

Set a a maximum execution **time**

Provide excellent **explainability**

Be generous with your logs

rules after midnight

Set a a maximum execution **time**

Provide excellent **explainability**

Be generous with your logs

Simpler is better

rules after midnight

Set a a maximum execution **time**

Provide excellent **explainability**

Be generous with your logs

Simpler is better

No AI is better

A PROGRAM OF Clarion West

ENSHITTIFICATION AND THE ROT ECONOMY: A DEEP DIVES CONVERSATION WITH CORY DOCTOROW AND ED ZITRON

MODERATED BY WHITNEY BELTRÁN

OFFERED WITH THE SUPPORT OF

4 CULTURE

The Seattle Public Library

THIRD PLACE BOOKS

LINDA BRENEMAN

OCTOBER 22, 2025 | 7:00PM PT | LIVE FROM THE SEATTLE CENTRAL LIBRARY

the future

To explore strange new worlds, to seek out new life and new civilizations, to boldly go where no one has gone before

tinyurl.com/ghavid

the future

To explore strange new worlds, to seek out new life and new civilizations, to boldly go where no one has gone before

tinyurl.com/ghavid