จากไอเดียสู่ระบบจริง

ง่ายกว่าที่คิด

AI-Powered Internal Developer Framework

สนับสนุนทุกไอเดีย ให้ deploy ได้เร็ว ปลอดภัย และยั่งยืน

สิงหาคม 2026

🎯 วาระการประชุม (90 นาที)

ขั้นตอน	หัวข้อ	เวลา
1	🌟 จุดเริ่มต้น: ไอเดียที่ดีสมควรได้รับการสนับสนุน	15 นาที
2	🚀 Journey: จากไอเดียสู่ระบบจริง	20 นาที
3	🤝 ทีมของเรา: ใครช่วยอะไรได้บ้าง	15 นาที
4	📚 AI ช่วยอะไรได้บ้าง	15 นาที
5	💡 เริ่มต้นอย่างไร	15 นาที

องค์ประชุม (1/2)

Maintenance (User)

- แชร์ไอเดียและความต้องการ
- ให้ feedback
- เลือก pilot projects

SA

- ออกแบบ architecture
- Wiki schema
- Approval workflow

AI-Eng

- LLM integration
- Auto-doc generation
- MCP, auto PR

Dev

- Implementation
- Workflow, templates

องค์ประชุม (2/2)

Infra&Security

- k3s, GitLab, ArgoCD
- Vault, RBAC, monitoring
- Network policies

🌟 ขั้นตอนแรก

จุดเริ่มต้น: ไอเดียที่ดีสมควรได้รับการสนับสนุน

เวลา: 15 นาที | ผู้นำเสนอ: Maintenance (User)

อะไรที่ทำให้คนอยาก deploy เร็ว?

จากประสบการณ์ที่ผ่านมา:

- คนที่มีไอเดียดี มักอยากเห็นผลงานตัวเองทำงานเร็ว
- กระบวนการปัจจุบันอาจยังไม่ทันกับความรวดเร็วของไอเดีย
- เครื่องมือที่มีอยู่อาจยังไม่ตอบโจทย์ self-service

คำถามเปิด: มีประสบการณ์อะไรที่อยากแชร์บ้างคะ?

💡 สิ่งที่เรามาเรียนรู้ร่วมกัน

ความท้าทายด้านกระบวนการ

- รอ approval นาน
- Infra อาจยังไม่พร้อมทันที

ความท้าทายด้านเครื่องมือ

- ไม่มี self-service
- ต้องพึ่ง infra team

ความท้าทายด้านข้อมูล

- ขาด documentation ที่ทันสมัย
- ความรู้กระจายอยู่ตามตัวบุคคล

โอกาสในการพัฒนา

- ทำให้ทุกขั้นตอนง่ายและเร็วขึ้น
- สนับสนุนคนที่มีไอเดียดี

🎯 สิ่งที่เราอยากสนับสนุน

สิ่งที่ควรมี:

- ✅ ใช้ง่ายเหมือน Vercel/Render
- ✅ Self-service deployment
- ✅ LLM integration + Auto-documentation
- ✅ AI-assisted development

คำถามเปิด: ต้องการอะไรเพิ่มเติม? มีข้อจำกัดอะไร?

📊 Output ของขั้นตอนแรก

รายการสิ่งที่เราอยากสนับสนุน (จัดลำดับความสำคัญ):

1.
2.
3.
4.
5.

🚀 ขั้นตอนที่สอง

Journey: จากไอเดียสู่ระบบจริง

เวลา: 20 นาที | ผู้นำเสนอ: SA

🏗️ Architecture Overview

```
💡 ไอเดีย → 💻 Code → 🤖 Auto Test → 📚 AI Docs → ✅ Review → 🚀 Deploy
                ↓            ↓              ↓          ↓          ↓
             GitLab    GitLab CI/CD     LLM Wiki   SA/SME    ArgoCD
             Push        (Pipeline)    (Qwen AI)   Review    → k3s
```

🎁 สิ่งที่ Framework จะมอบให้ (1/2)

1. Self-Service Deployment

- ใช้ง่ายเหมือน Vercel
- Deploy บน k3s ขององค์กร

2. AI-Powered Documentation

- LLM สร้าง wiki อัตโนมัติ
- อัพเดทเมื่อ code เปลี่ยน

3. Knowledge Management

- ระบบ approve ก่อน publish
- Version control

4. Auto Code Review

- AI ตรวจ code quality, security

🎁 สิ่งที่ Framework จะมอบให้ (2/2)

5. MCP Integration

- ทุก app มี MCP server
- AI agent ใช้งานได้ทันที

6. Governance & Support

- สนับสนุนเต็มที่
- ตรวจสอบได้
- Audit trail

🛠️ Tech Stack (1/2)

Layer	Tool	หน้าที่
Git + CI	**GitLab**	Git server + CI/CD pipeline
CD	**ArgoCD**	GitOps deploy to k3s
Orchestration	**k3s**	Lightweight Kubernetes

🛠️ Tech Stack (2/2)

Layer	Tool	หน้าที่
LLM Gateway	**LiteLLM**	Cache, log, route LLM calls
LLM	**Alibaba Qwen**	Token Plan (credit-based)
Secrets	**Vault**	API keys, credentials
Registry	**Harbor**	Container images
Monitoring	**Prometheus + Grafana**	Metrics, dashboards

🤝 ขั้นตอนสาม

ทีมของเรา: ใครช่วยอะไรได้บ้าง

เวลา: 15 นาที | ผู้นำเสนอ: ทุกทีม

👥 ทีมพร้อมสนับสนุน

ทีม	หน้าที่	ช่วยอะไรได้บ้าง
SA	Architecture	ออกแบบระบบ, wiki schema
AI-Eng	LLM Integration	Auto-docs, MCP, AI review
Dev	Implementation	Workflow, templates
Infra	Infrastructure	k3s, monitoring, security

คุณแค่มีไอเดีย — ที่เหลือเราช่วย!

📚 LLM Wiki คืออะไร?

หลักการ: ทุก app มี documentation อัตโนมัติ

- AI อ่าน code + design docs → สร้าง structured wiki อัตโนมัติ
- เก็บเป็น markdown files (ไม่ต้องใช้ vector DB)
- อัพเดทอัตโนมัติเมื่อ code เปลี่ยน

🔄 Workflow

```
1. Dev push code → GitLab CI/CD pipeline trigger
                   ↓
2. LLM (Qwen) อ่าน code → สร้าง wiki pages
                   ↓
3. Wiki ถูก commit กลับเข้า repo
                   ↓
4. SA/SME review → approve/reject
                   ↓
5. approve → อนุญาต deploy | reject → แจ้ง Dev แก้ไข
```

📂 โครงสร้าง LLM Wiki

```
project-wiki/
├── raw/              ← source material (code, design docs)
├── wiki/             ← AI-generated structured knowledge
│   ├── index.md
│   ├── architecture.md
│   ├── api-reference.md
│   ├── deployment-guide.md
│   └── troubleshooting.md
└── outputs/          ← reports, answers
```

✅ KM Approval Process

SME Review

- Subject Matter Expert review ทุก wiki page

Version Control

- Track การเปลี่ยนแปลง, Rollback ได้

Periodic Review

- ตั้งเวลา review ทุก 6 เดือน

Audit Log

- รู้ว่าใคร approve, เมื่อไหร่, อะไรเปลี่ยน

🤔 ทำไมต้อง Approve?

เพื่อสร้างคุณภาพ:

- ✅ ข้อมูลถูกต้อง เป็นปัจจุบัน
- ✅ Documentation ที่เชื่อถือได้
- ✅ Quality assurance + Accountability

เพื่อสร้างความเป็นเจ้าของ:

- ✅ ทีมช่วยกันตรวจสอบ
- ✅ ทุกคนมีส่วนร่วมในการสร้างความรู้

📚 ขั้นตอนสี่

AI ช่วยอะไรได้บ้าง

เวลา: 15 นาที | ผู้นำเสนอ: AI-Eng

🎯 LiteLLM คืออะไร?

LLM Gateway

- รวม LLM calls ทั้งหมดที่เดียว

Caching

- Cache input/output → ลด cost, เพิ่ม speed

Logging

- Log ทุก call → ทำเป็น knowledge base

Rate Limiting

- ควบคุม usage per project

💰 Alibaba Token Plan

คิดแบบ Credit ไม่ใช่ per-call

Seat Type	Price	Quota
Standard	$30/seat/mo	25,000 credits
Pro	$100/seat/mo	100,000 credits
Max	$200/seat/mo	250,000 credits

Models: Qwen, DeepSeek, Kimi, GLM, MiniMax

💰 Alibaba Token Plan — Models

Brand	Models
Qwen	qwen3.7-max, qwen3.7-plus, qwen3.6-plus, qwen3.6-flash
DeepSeek	deepseek-v4-pro, deepseek-v4-flash, deepseek-v3.2
Kimi	kimi-k2.7-code, kimi-k2.6, kimi-k2.5
GLM	glm-5.2, glm-5.1, glm-5
MiniMax	MiniMax-M2.5

🏗️ Architecture

```
User Request → 9router (round-robin)
                      ↓
               LiteLLM Proxy
                      ↓
            ┌─────────┴─────────┐
    API Key 1             API Key 2
    (Seat 1)              (Seat 2)
            └─────────┬─────────┘
               Alibaba Qwen API
```

ประโยชน์: กระจาย load, หลีกเลี่ยง rate limits, serve 10+ users

🧠 Knowledge Base จาก LLM Logs

Input/Output ที่ log ไว้ใน LiteLLM:

- นำมาสร้าง wiki ได้
- Cache hits → ลด cost, เพิ่ม speed

Workflow:

```
LLM Call → LiteLLM Log → Langfuse → Analyze → Wiki Update
```

💡 ขั้นตอนห้า

เริ่มต้นอย่างไร

เวลา: 15 นาที | ผู้นำเสนอ: ทุกทีม

🌟 3 ขั้นตอนง่ายๆ

1. แชร์ไอเดีย

- บอกเราว่าอยากทำอะไร
- ไม่ต้องมี code ก็ได้

2. ทีมช่วย setup

- เราเตรียม infra ให้
- AI ช่วยสร้างเอกสาร

3. Deploy ได้เลย

- ใช้ง่ายเหมือน Vercel
- แต่ปลอดภัย + มี backup + มี monitoring

🔌 Network & Access: ไม่ต้องกังวลเรื่อง Port และ DNS

ปัญหาเดิม:

- ❌ User ต้องรู้ว่า app รันบน port ไหน
- ❌ ต้องเปิด ticket ขอ IT สร้าง DNS record (รอ 1-3 วัน)
- ❌ IT เป็น bottleneck สร้าง DNS ให้ทุก app

วิธีแก้:

- ✅ **Wildcard DNS** — IT ทำครั้งเดียว `*.apps.company.com`
- ✅ **Ingress Controller** — Traefik จัดการ routing อัตโนมัติ
- ✅ **ไม่ต้องเปิด ticket อีกเลย!**

🌐 Wildcard DNS: IT ทำครั้งเดียว จบ!

สิ่งที่ IT ทำ (ครั้งเดียว):

1. สร้าง DNS: `*.apps.company.com`
2. ชี้ไป IP ของ k3s cluster
3. ตั้งค่า TLS certificate (Let's Encrypt)

เสร็จแล้ว! ไม่ต้องทำอะไรอีก

สิ่งที่ User ทำ:

```
# แค่กำหนดชื่อใน Ingress
spec:
  rules:
  - host: my-app.apps.company.com
```

ผลลัพธ์:

- ✅ App เข้าถึงได้ทันที
- ✅ ไม่ต้องรอ IT
- ✅ ไม่ต้องรู้ port

🚀 ตัวอย่าง: Deploy แล้วได้ URL ทันที

```
# User push code → GitLab CI/CD deploy
$ git push origin main

# GitLab CI/CD สร้าง Ingress อัตโนมัติ
$ kubectl apply -f ingress.yaml

# ✅ App พร้อมใช้งาน!
🎉 Your app is live at: http://my-app.apps.company.com
```

ไม่ต้อง:

- ❌ เปิด ticket ขอ DNS
- ❌ รอ IT สร้าง record
- ❌ รู้ว่า app รันบน port ไหน
- ❌ ตั้งค่า firewall

แค่ push code → ได้ URL ทันที!

📅 Phase 1: Foundation (Month 1-2)

Infrastructure Setup:

- [ ] Setup k3s cluster (3 nodes)
- [ ] Setup GitLab + GitLab Runner
- [ ] Deploy ArgoCD, Harbor, Traefik
- [ ] Deploy LiteLLM + Redis + Langfuse

Deliverables: k3s cluster + GitLab repo + LLM gateway

📅 Phase 2: Automation (Month 3-4)

AI Integration:

- [ ] Integrate Alibaba Qwen API
- [ ] Build auto-doc generation pipeline
- [ ] Implement KM approval workflow
- [ ] Create MCP server template

Deliverables: Auto-doc + KM workflow + MCP template

📅 Phase 3: Polish (Month 5-6)

User Experience:

- [ ] Build self-service portal
- [ ] Implement governance policies
- [ ] Onboard 3 pilot projects
- [ ] Training & documentation

Deliverables: Portal + 3 pilots + Training materials

🔌 MCP Server Auto-Registration

ทุก app มี MCP server

```
from fastmcp import FastMCP
mcp = FastMCP("my-app")

@mcp.tool()
def query_data(query: str) -> dict:
    """Query data from app"""
    pass
```

Auto-register to MCP catalog on deploy

🤖 Auto MR & Code Review

```yaml
stages:
  - review

ai-review:
  stage: review
  image: python:3.11
  rules:
    - if: '$CI_PIPELINE_SOURCE == "merge_request_event"'
  script:
    - pip install openai
    - python scripts/ai-review.py
      --llm_endpoint "http://litellm:4000/v1"
      --model "qwen-plus"
      --mr_iid $CI_MERGE_REQUEST_IID
```

🚀 Immediate Actions (สัปดาห์หน้า)

Infra&Security:

- Setup k3s cluster (test)
- Deploy LiteLLM + Redis

AI-Eng:

- ทดสอบ Alibaba Qwen API
- สร้าง prototype auto-doc

SA:

- ออกแบบ wiki schema
- กำหนด KM approval criteria

Dev:

- Setup GitLab CI/CD pipelines
- สร้าง project template

🎯 Pilot Projects

เลือก 3 projects จาก Maintenance (User)

Criteria:

- Complexity ปานกลาง
- Team พร้อมร่วมมือ
- มี business impact ชัดเจน

Timeline: Onboard 2 สัปดาห์ → Deploy แรกภายใน 1 เดือน

📊 Success Metrics

Developer Satisfaction

- Target: >80% satisfaction

Deployment Time

- Target: <15 นาที

Documentation Coverage

- Target: 100% มี wiki

Self-driven Solutions on Internal Platform

- Target: เพิ่ม 50% ใน 3 เดือน

📝 Summary

จุดเริ่มต้น:

- ไอเดียที่ดีสมควรได้รับการสนับสนุน
- ทำให้ทุกขั้นตอนง่ายและเร็วขึ้น

แนวทาง:

- AI-Powered Framework
- LLM Wiki + KM Approval

Tech Stack:

- k3s + ArgoCD + GitLab
- LiteLLM + Alibaba Qwen

Timeline: 6 months, 3 phases

🙏 ขอบคุณค่ะ

ทุกไอเดียสมควรได้รับการสนับสนุน

Project: `~/projects/ai-powered-internal-developer-framework`

📎 Appendix: Backup Data

ข้อมูลสำรอง (ไม่พูดในการประชุม):

1. **Alibaba Token Plan Details**
   - Full model list, credit calculation, pricing
   - ดูที่: `docs/backup/alibaba-token-plan.md`
2. **CI/CD Comparison**
   - GitLab CI vs GitHub Actions vs Argo Workflows
   - ดูที่: `docs/backup/cicd-comparison.md`