# Building an AI French Teacher

Introduction

# The Mission: What Are We Building?

**Project Goal:** Create an AI French teacher that speaks naturally with Chinese learners.

# Three AI Systems Working Together

# Three AI Systems Working Together

To make this work, we need **THREE different AI systems** to collaborate:

🎤 AI #1: Speech-to-Text (STT)

🧠 AI #2: Large Language Model (LLM)

📣 AI #3: Text-to-Speech (TTS)

# Three AI Systems Working Together

## 🎤 AI #1: Speech-to-Text (STT)

**What it does:** Converts spoken words into written text
**Why we need it:** Students will speak to the AI teacher, so we need to "hear" and understand their voice
**Example model:** Whisper (by OpenAI)
**Real-world use:** Voice assistants like Siri, Google Assistant, dictation software

# Three AI Systems Working Together

## 🧠 AI #2: Large Language Model (LLM)

**What it does:** Understands and generates human language, like having a conversation
**Why we need it:** This is the "brain" - it understands questions, creates lessons, explains grammar, gives feedback
**Example model:** Qwen2.5-Instruct (by Alibaba Cloud)

# Three AI Systems Working Together

## 🔊 AI #3: Text-to-Speech (TTS)

**What it does:** Converts written text into spoken words with natural voice
**Why we need it:** The AI teacher needs to speak back to students with correct French pronunciation
**Example model:** Coqui TTS
**Real-world use:** Audiobooks, navigation systems, accessibility tools for visually impaired

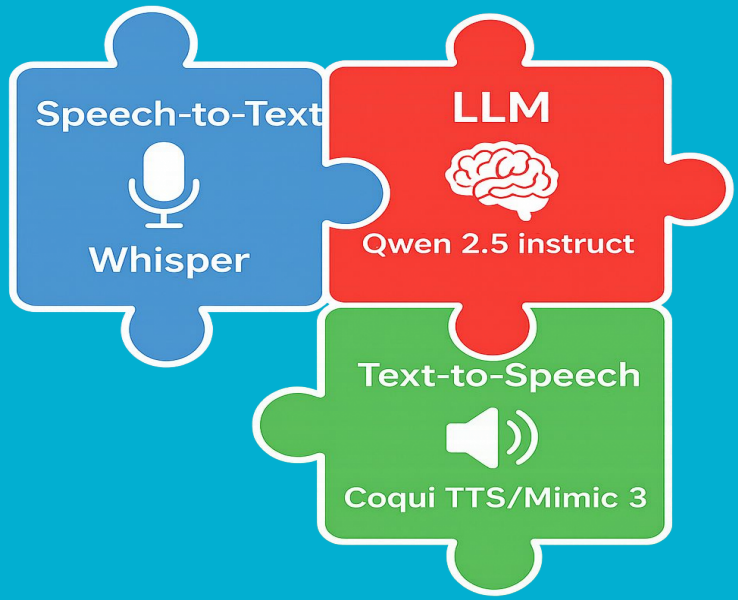
# Modular Design: Like Building with LEGO Blocks

We will build this project in a **modular way**

Think of each AI as a separate **puzzle piece** or **LEGO block**:

**Why is this important?**

1. **Flexibility:** Test different AI models to find the best one
2. **Future-proof:** When better AI comes out, easily upgrade
3. **Learning:** Understand how professional systems are built
4. **Reusability:** Use this architecture for other projects (English teacher, Math tutor, etc.)



# Why This Project Will Change Your Career

# Why This Project Will Change Your Career

**By the end of this project, you will have:**

1. **Built a complete AI system** from scratch
2. **Learned by doing** (not just theory)
3. **Worked in teams** like professional developers
4. **Solved real technical problems** independently
5. **Created something actually useful** that helps people learn

**Most importantly:** You'll have the complete skill set companies look for when hiring developers.

# Why This Project Will Change Your Career

## 🛠️ Technical Skills Overview

| Skill | Why We Need It |
| --- | --- |
| Git & Gitee | Share code, collaborate, and track changes |
| Python Basics | Programming language for the entire project |
| Object-Oriented Programming (OOP) | Organize code into modular, reusable components |
| APIs (RESTFUL) | Make different systems communicate with each other |
| Docker | Package the entire system to run anywhere |

# Why This Project Will Change Your Career

## 🛠️ Technical Skills Overview

| Skill | Why We Need It |
| --- | --- |
| MCP (Model Control Platform) | Manage lessons, exercises, and control AI workflow |
| Working with LLMs | Understand and integrate language models like Qwen |
| Database & ORM | Save student progress, lessons, and conversation history |
| Alibaba Cloud | Deploy and scale larger AI models in the cloud |

# 🎯 The Most Important Skill: Learning to Learn

The Real Challenge:

❌ In companies, there's no professor to teach you every new framework
❌ Technologies evolve faster than university curricula can update
❌ Every project uses different tools you've never seen before
✅ **You must learn to figure things out independently**

# 🎯 The Most Important Skill: Learning to Learn

The Real Challenge:

University Knowledge
↓ [Graduates]
↓ Job Market (40 years of career)
↓ New frameworks every 2-3 years
New AI models every 6 months
New best practices constantly
↓ Your university knowledge
becomes outdated quickly!

# 🎯 The Most Important Skill: Learning to Learn

The Real Challenge:

**What DOESN'T become outdated:**

- ✅ Your ability to **learn new technologies independently**
- ✅ Your skill at **reading documentation and understanding it**
- ✅ Your confidence to **try, fail, and try again**
- ✅ Your resourcefulness in **finding solutions**

# 🎯 The Most Important Skill: Learning to Learn

This project teaches you independence:

❌ Traditional learning (passive):

```
Professor teaches → You memorize → You pass exam → You forget
```

✅ Project-based learning (active):

```
You need X → You research X → You test X → You struggle with X
→ You ask for help → You finally understand X → You NEVER forget X
```

# How to Start the Project: Understand Before You Code

# **How to Start the Project**: Understand Before You Code

## **Step 1:** Theoretical Understanding

**Important principle:** Don't start coding immediately! Professional developers always follow this process:

**Before writing any code, understand HOW the project will work.**

Learn about each technology you'll need:

- **RESTful APIs:** How do different programs communicate?
- **LLMs:** How do language models process and generate text?
- **OOP:** How to structure code with classes and objects?
- **ORM:** How to interact with databases using Python objects?
- **Docker:** How to package applications for deployment?
- **MCP:** How to orchestrate multiple AI systems?

# **How to Start the Project**: Understand Before You Code

**Step 1:** Theoretical Understanding

## Learning resources:

- 🎥 **Bilibili videos** - Search for tutorials in Chinese
- 🤖 **Use AI assistants**
- 👥 **Talk with experienced students**

**Goal:** Understand the concepts theoretically, even if you can't code them yet.

# **How to Start the Project**: Understand Before You Code

## **Step 2:** Visualize the Architecture

**Once you understand the technologies, create a diagram of the project.**

This verifies your understanding before writing code.

**Create a diagram (digital or paper) showing:**

- The 3 AI systems (STT, LLM, TTS)
- How they connect to each other
- Where the MCP fits
- How the database stores information
- How APIs communicate between components
- Where Docker containers are used

# **How to Start the Project**: Understand Before You Code

**Step 2:** Visualize the Architecture

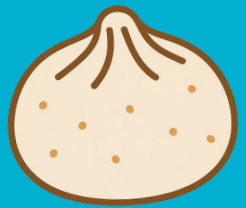
**Example questions to answer in your diagram:**

1. When a user speaks, what happens first?
2. How does the voice become text?
3. Where does the text go next?
4. How does Qwen generate a response?
5. How does the response become voice?
6. Where is student progress saved?

# **How to Start the Project**: Understand Before You Code

**Step 2:** Visualize the Architecture

**Validation:** Show your diagram to teammates and professor for feedback. Does it make sense? Did you miss anything?



# **How to Start the Project**: Understand Before You Code

**Step 3:** Break Down the Project into Categories and Tasks

**The secret to success**: Make big problems small!

**Why this matters for first-year students:** Even if you're just learning Python, you CAN contribute to this project! How?

# **How to Start the Project**: Understand Before You Code

## **Step 3:** Break Down the Project into Categories and Tasks

❌ **Large task**: "Build an AI teacher" → Too difficult, too vague
**Small task**: "Install Qwen 2.5 on your computer + test it following a Bilibili tutorial" → Possible! ✅
**Small task**: "Install Whisper on your computer and test it" → Possible! ✅
**Small task:** "Create a Student class with name and level" → Possible! ✅
**Small task**: "Test if Qwen responds to a simple question in Chinese" → Possible! ✅
**Small task:** "Write a function to save progress data to a text file" → Possible! ✅

# **How to Start the Project**: Understand Before You Code

## **Step 3:** Break Down the Project into Categories and Tasks

**How we'll organize the work:**

**Professor Baptiste will provide the initial breakdown:**

- 📂 **Categories** (major components of the project)
- 📝 **Tasks** (specific work items within each category)
- 🎯 **Difficulty levels** (so you know which tasks match your skill level)

# **How to Start the Project**: Understand Before You Code

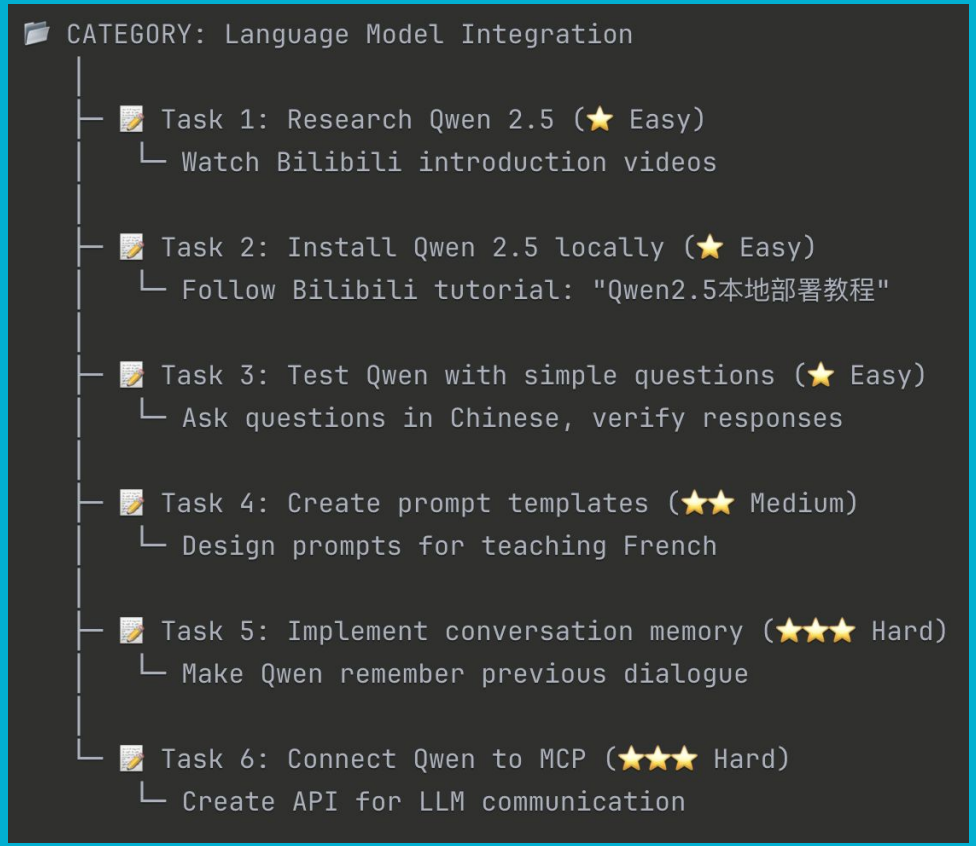
## Example of how a project is broken down:

### Clear progress tracking

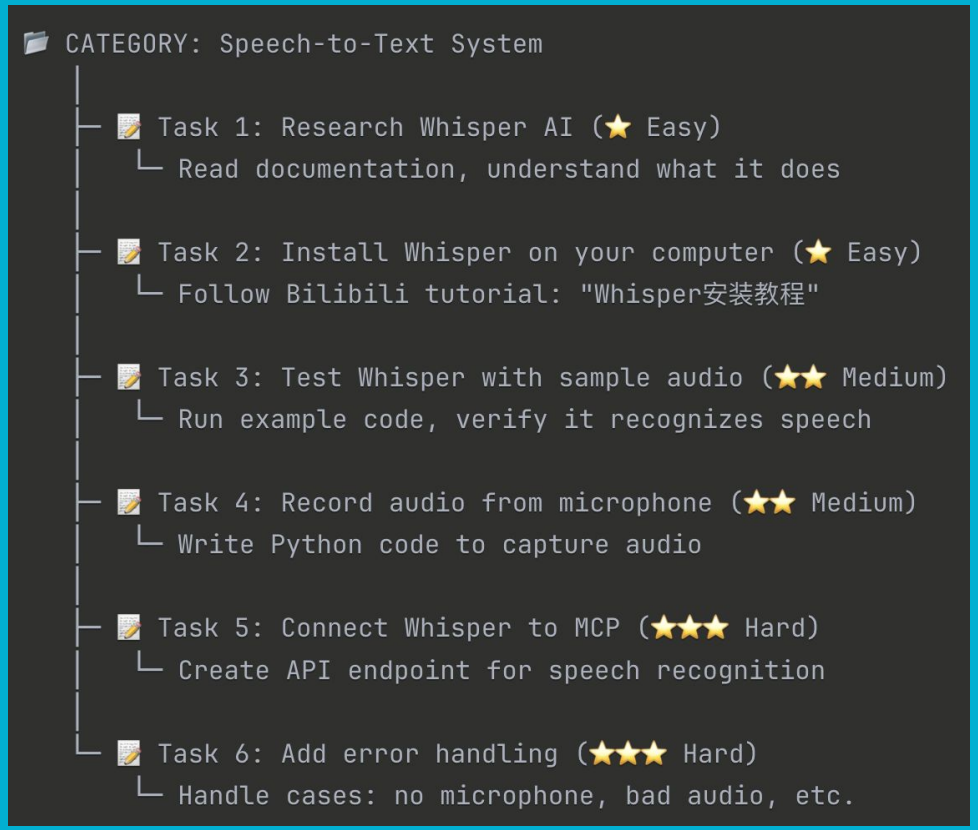
- "We completed 15 out of 50 tasks" ✅
- Motivating to see progress!

### Parallel work

- Multiple team members work on different tasks simultaneously
- Project moves faster



# **How to Start the Project**: Understand Before You Code

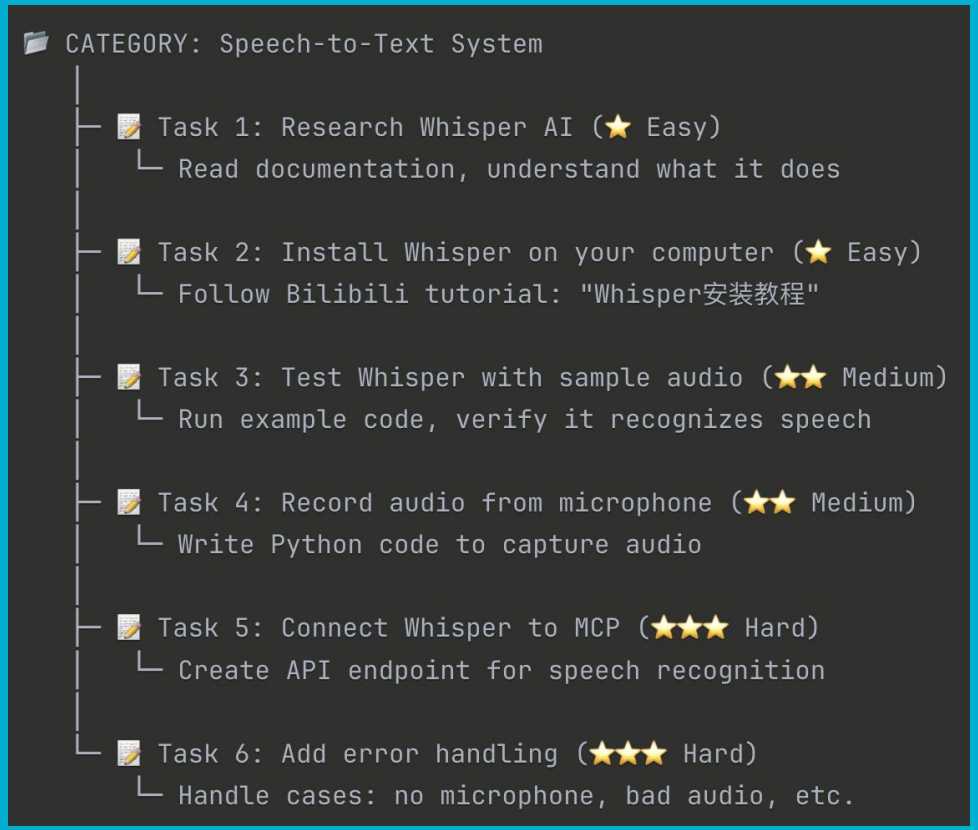



**Example of how a project is broken down:**

**Skill-appropriate assignments**

- First-years can work on ⭐ and ⭐⭐ tasks
- Second-years can focus on ⭐⭐ and ⭐⭐⭐ tasks
- Everyone contributes!

# **How to Start the Project**: Understand Before You Code




**Example of how a project is broken down:**

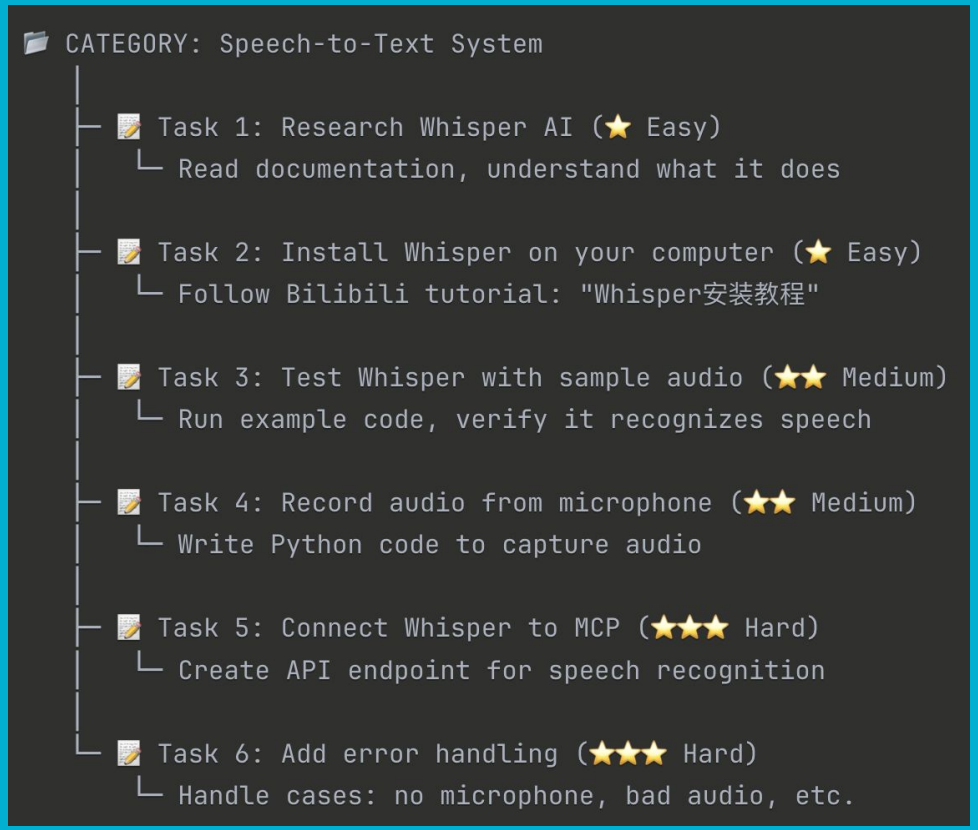
**Reduced overwhelm**

- Focus on ONE task at a time
- Celebrate small wins
- Build confidence gradually

**Learning by doing**

- Each task teaches specific skills
- Cumulative learning throughout the project

# **How to Start the Project**: Understand Before You Code




**Example of how a project is broken down:**

**Reduced overwhelm**

- Focus on ONE task at a time
- Celebrate small wins
- Build confidence gradually

**Learning by doing**

- Each task teaches specific skills
- Cumulative learning throughout the project

# Project Organization: Teams, Support, and Collaboration

# **Project Organization:** Teams, Support, and Collaboration

## Team Formation

You will form mixed t**eams of 3-4 students:**

**Why mixed teams?**

- First-year students learn from second-year students
- Second-year students strengthen their knowledge by teaching
- Everyone contributes at their skill level
- This mirrors real company teams with junior and senior developers

# **Project Organization:** Teams, Support, and Collaboration

## Team Formation

## Project Leaders (Third-Year Students)

**Third-year students will serve as Project Leaders** to support multiple teams:

- They are **NOT** part of one specific team
- They **help several teams** as mentors and guides
- They act as **technical advisors** and **project coordinators**

# **Project Organization:** Teams, Support, and Collaboration

## Team Formation

## How Project Leaders work:

Project Leaders don't just answer questions—they actively **learn, test, and share solutions**:

1. **Learn ahead of the teams**
   - Study new technologies before teams need them
   - Watch tutorials on Bilibili (搜索并学习新技术)
   - Read official documentation (Qwen, Whisper, Docker, etc.)
   - Test concepts on their own computers first

# **Project Organization:** Teams, Support, and Collaboration

**Team Formation**

**How Project Leaders work:**

**2 - Test solutions independently**

- Install and experiment with each technology
- Try different approaches to solve problems
- Document what works and what doesn't
- Create simple examples and demos

# **Project Organization:** Teams, Support, and Collaboration

**Team Formation**

**How Project Leaders work:**

**3 - Share resources and knowledge**

- Curate best Bilibili tutorials (分享最好的教程)
- Share relevant documentation links
- Create quick reference guides
- Recommend specific articles or videos for each topic

# **Project Organization:** Teams, Support, and Collaboration

## Team Formation

## How Project Leaders work:

### 4 - Guide teams through tasks

- Break down complex tasks into smaller steps
- Suggest learning paths for each technology
- Help teams when they're stuck
- Review approaches before teams spend too much time

# **Project Organization:** Teams, Support, and Collaboration

**Example of workflow**

```
Week 1: Teams need to learn Docker
    ↓
Project Leader:
  • Watches 3-4 Docker tutorials on Bilibili
  • Tests Docker installation on own machine
  • Creates a simple example container
  • Shares best tutorial link in WeChat
  • Writes quick guide: "5 Steps to Install Docker"
    ↓
Teams:
  • Follow the tutorial
  • Use the guide
  • Ask Project Leader specific questions
  • Successfully install Docker!
```

# **Project Organization:** Teams, Support, and Collaboration

**Project Leader responsibilities:**

- ✅ **Learn and test solutions first** before guiding teams
- ✅ **Find and share quality resources** (Bilibili, docs, articles)
- ✅ Help teams understand technical concepts (OOP, APIs, Docker, etc.)
- ✅ Guide teams when they're stuck on difficult problems
- ✅ Monitor project progress across teams
- ✅ Share best practices and working examples
- ✅ Facilitate knowledge transfer between teams
- ✅ Organize code review sessions

# **Project Organization:** Teams, Support, and Collaboration

**Why this role benefits third-year students:**

This Project Leader role isn't just about helping others—it's a powerful learning opportunity for third-year students themselves!

## 1. The Feynman Technique: Learn by Teaching

"If you can't explain it simply, you don't understand it well enough." - Richard Feynman

# **Project Organization:** Teams, Support, and Collaboration

**When you teach others, you:**

🧠 **Deepen your own understanding** - Explaining forces you to truly master concepts

🔍 **Discover gaps in your knowledge** - Students' questions reveal what you need to study more

💡 **See problems from new angles** - Beginners ask questions experts never think of

🎯 **Solidify learning** - Teaching is the best way to remember

# **Project Organization:** Teams, Support, and Collaboration

**When you teach others, you:**

**Example:**

**Before teaching**: "I think I understand Docker..."
**After explaining to 3 teams**: "Now I REALLY understand Docker containers, images, volumes, and networking!"

# **Project Organization:** Teams, Support, and Collaboration

## 2. Project Management Skills

**As a Project Leader, you'll develop crucial professional skills:**

**Technical Project Management:**

- 📊 **Planning and coordination** - Organize work across multiple teams
- 🎯 **Priority setting** - Decide what's important vs what can wait
- ⏰ **Time estimation** - Learn how long tasks actually take
- 🚧 **Risk management** - Identify potential problems before they happen
- 📈 **Progress tracking** - Monitor multiple teams simultaneously

# **Project Organization:** Teams, Support, and Collaboration

## 2. Project Management Skills

**As a Project Leader, you'll develop crucial professional skills:**

**People Management:**

- 👥 **Mentoring and coaching** - Guide others without doing their work
- 💬 **Communication skills** - Explain complex topics clearly
- 🤝 **Conflict resolution** - Help teams work through disagreements
- 🎉 **Motivation** - Keep teams engaged and encouraged
- 📣 **Presentation skills** - Share knowledge effectively

# **Project Organization:** Teams, Support, and Collaboration

## 2. Project Management Skills

**As a Project Leader, you'll develop crucial professional skills:**

**Strategic Thinking:**

- 🗺️ **Big picture perspective** - Understand how all pieces fit together
- 🔄 **Resource allocation** - Match tasks to team capabilities
- 📚 **Knowledge management** - Organize and share information
- 🎓 **Continuous improvement** - Identify what works and what doesn't

# **Project Organization:** Teams, Support, and Collaboration

## 2. Project Management Skills

**As a Project Leader, you'll develop crucial professional skills:**

**Real-world preparation:**

These are the **exact skills** companies look for when hiring:

- Senior developers who can mentor juniors ✅
- Technical leads who coordinate teams ✅
- Project managers who deliver results ✅
- Engineers who communicate effectively ✅

# **Project Organization:** Teams, Support, and Collaboration

**Your growth as a Project Leader:**

```
Learn technologies, prepare resources
↓
Guide first teams, answer basic questions
↓
Facilitate between teams, share patterns
↓
Coordinate complex integrations
↓
Lead code reviews, teach best practices
↓
: Manage final integration, prepare presentation
```

# Next Step

# Next Step

📋 **Get Started**

1. Form Your Team

- Mix first-year and second-year students
- Create team WeChat group
- Choose a team name
- Announce in general WeChat group

# Next Step

📋 **Get Started**

## 2. Understand the Project

- Watch tutorials on Bilibili about LLMs, APIs, Speech-to-Text
- Discuss together: How does the system work?
- Create a simple diagram (paper or digital) showing:
  - User → UI → MCP → 3 AIs → Database
  - Data flow with arrows

# Next Step

📋 Get Started

## 3. Organize Your Work

- Set up a Trello board: [TO DO] [IN PROGRESS] [DONE]
- Professor Baptiste will provide the initial task breakdown
- Add tasks to Trello with difficulty levels (⭐ to ⭐⭐⭐⭐)

# Next Step

📋 Get Started

## 4. Start with Easy Tasks

**For first-year students (⭐):**

- Install Qwen 2.5 following Bilibili tutorial
- Install Whisper and test it
- Create a simple Student class in Python