# LLM提取

<xsl:stylesheet version="1.0">

<xsl:template match="/">

<!-- 记忆上下文初始化 -->

<xsl:variable name="memory" select="

if (/conversation/memory) then /conversation/memory

else document('')//default\_memory

"/>

<!-- 默认记忆存储 -->

<default\_memory>

<last\_disease>数据缺失</last\_disease>

<last\_growth>数据缺失</last\_growth>

<last\_location>数据缺失</last\_location>

</default\_memory>

<!-- 增强版输入解析 -->

<xsl:variable name="raw\_input" select="normalize-space(#context#)"/>

<!-- 地点识别增强（支持中国所有县级行政区） -->

<xsl:variable name="detected\_location">

<xsl:choose>

<!-- 显式参数修改指令 -->

<xsl:when test="contains($raw\_input, '将地点更改为')">

<xsl:value-of select="normalize-space(substring-after($raw\_input, '将地点更改为'))"/>

</xsl:when>

<!-- 中国县级行政区正则匹配（简化版） -->

<xsl:when test="matches($raw\_input, '([\u4e00-\u9fa5]{2,5}县|[\u4e00-\u9fa5]{2,5}区|[\u4e00-\u9fa5]{2,5}市)(\s|$)')">

<xsl:analyze-string select="$raw\_input" regex="([\u4e00-\u9fa5]{{2,5}}(?:县|区|市))">

<xsl:matching-substring>

<xsl:value-of select="regex-group(1)"/>

</xsl:matching-substring>

</xsl:analyze-string>

</xsl:when>

<!-- 未识别到地点时使用记忆值 -->

<xsl:otherwise>

<xsl:value-of select="$memory/last\_location"/>

</xsl:otherwise>

</xsl:choose>

</xsl:variable>

<!-- 参数提取与记忆更新（严格模式） -->

<xsl:variable name="current\_disease">

<xsl:choose>

<xsl:when test="疾病 and 疾病!='' and 疾病!='数据缺失'">

<xsl:value-of select="疾病"/>

</xsl:when>

<!-- 仅当输入明确包含疾病特征时才使用记忆值 -->

<xsl:when test="contains($raw\_input, '病') or contains($raw\_input, '虫') or contains($raw\_input, '斑')">

<xsl:value-of select="$memory/last\_disease"/>

</xsl:when>

<xsl:otherwise>数据缺失</xsl:otherwise>

</xsl:choose>

</xsl:variable>

<xsl:variable name="current\_growth">

<xsl:choose>

<xsl:when test="生長期 and 生長期!=''">

<xsl:value-of select="生長期"/>

</xsl:when>

<xsl:otherwise>数据缺失</xsl:otherwise>

</xsl:choose>

</xsl:variable>

<!-- 最终地点确认（非空检查） -->

<xsl:variable name="current\_location">

<xsl:choose>

<xsl:when test="$detected\_location != ''">

<xsl:value-of select="$detected\_location"/>

</xsl:when>

<xsl:otherwise>数据缺失</xsl:otherwise>

</xsl:choose>

</xsl:variable>

<!-- 记忆存储更新 -->

<update\_memory>

<last\_disease><xsl:value-of select="$current\_disease"/></last\_disease>

<last\_growth><xsl:value-of select="$current\_growth"/></last\_growth>

<last\_location><xsl:value-of select="$current\_location"/></last\_location>

<timestamp><xsl:value-of select="substring(#context#,1,16)"/></timestamp>

</update\_memory>

<!-- 严格输出保证 -->

<output>

<xsl:value-of select="concat(

$current\_disease, ',',

$current\_growth, ',',

$current\_location

)"/>

</output>

<!-- 地点识别测试案例 -->

<test\_cases>

<case input="陇南徽县">数据缺失,数据缺失,徽县</case>

<case input="河南杞县">数据缺失,数据缺失,杞县</case>

<case input="将地点更改为 甘肃礼县">数据缺失,数据缺失,甘肃礼县</case>

<case input="叶片有锈斑 石家庄">玉米锈病,数据缺失,石家庄</case>

<case input="库里">数据缺失,数据缺失,数据缺失</case>

</test\_cases>

</xsl:template>

</xsl:stylesheet>

# 参数提取器

<xsl:stylesheet version="1.0">

<xsl:template match="/">

<!-- 时间处理 -->

<xsl:variable name="current\_time" select="substring(#context#,1,16)"/>

<strict\_rules>

1. 仅输出3参数：疾病,生长期,地点

2. 禁止输出任何其他文本

3. 缺失参数显示"数据缺失"

</strict\_rules>

<parameter\_options>

<疾病 strict="true">

● 玉米斑病（匹配词：大斑病/小斑病/叶片褐斑）

● 玉米锈病（匹配词：锈斑/铁锈色粉末）

● 玉米黑粉病（匹配词：黑粉包/瘤状物）

● 玉米根腐病（匹配词：烂根/根发黑）

● 玉米苗枯病（匹配词：死苗/苗基部腐烂）

● 玉米螟（匹配词：蛀孔/钻心虫）

● 玉米黏虫（匹配词：吃叶虫/叶片缺刻）

● 玉米蓟马（匹配词：银白色条纹）

</疾病>

<生长期 strict="true">

★ 苗期（匹配词：小苗/3-5片叶）

★ 拔节期（匹配词：茎秆伸长）

★ 抽雄期（匹配词：抽穗/吐丝）

★ 灌浆期（匹配词：籽粒乳熟）

★ 成熟期（匹配词：叶片发黄）

</生长期>

</parameter\_options>

<output>

<xsl:variable name="disease" select="if (疾病 and 疾病!='') then 疾病 else '数据缺失'"/>

<xsl:variable name="growth" select="if (生長期 and 生長期!='') then 生長期 else '数据缺失'"/>

<xsl:variable name="location" select="if (地点 and 地点!='') then 地点 else '数据缺失'"/>

<xsl:value-of select="concat(

$disease, ',',

$growth, ',',

$location

)"/>

</output>

</xsl:template>

</xsl:stylesheet>

# 天气代码执行

import urllib.parse

import urllib.request

import json

import ssl

def main(location: str) -> dict:

# 如果location是"数据缺失"，直接返回

if location == "数据缺失":

return {

"weather": "数据缺失",

"temperature": "数据缺失",

"humidity": "数据缺失",

"windLevel": "数据缺失"

}

# 创建不验证SSL的上下文（针对某些环境问题）

ssl\_context = ssl.\_create\_unverified\_context()

# 配置API参数

host = 'https://ali-weather.showapi.com'

path = '/area-to-weather'

appcode = 'cecca0586c5343d1ac5180ed6329f32b'

# 构造查询参数（确保城市名称正确编码）

query\_params = urllib.parse.urlencode({

'area': location,

'needMoreDay': '0',

'needIndex': '0',

'need3HourForcast': '0',

'needAlarm': '0',

'needHourData': '0'

})

# 构建完整URL

url = f"{host}{path}?{query\_params}"

try:

# 创建请求对象

request = urllib.request.Request(url)

request.add\_header('Authorization', f'APPCODE {appcode}')

# 发送请求

with urllib.request.urlopen(request, context=ssl\_context) as response:

raw\_data = response.read().decode('utf-8')

data = json.loads(raw\_data)

# 检查API响应状态

if data.get('showapi\_res\_code') != 0:

error\_msg = data.get('showapi\_res\_error', 'API未返回具体错误信息')

return {

"weather": "数据缺失",

"temperature": "数据缺失",

"humidity": "数据缺失",

"windLevel": "数据缺失",

"error": f"API错误: {error\_msg}"

}

# 直接提取实时天气数据

now\_data = data['showapi\_res\_body']['now']

return {

"weather": now\_data.get('weather', '数据缺失'),

"temperature": f"{now\_data.get('temperature', 'N/A')}℃" if now\_data.get('temperature') else "数据缺失",

"humidity": now\_data.get('sd', '数据缺失'),

"windLevel": now\_data.get('wind\_power', '数据缺失')

}

except Exception as e:

return {

"weather": "数据缺失",

"temperature": "数据缺失",

"humidity": "数据缺失",

"windLevel": "数据缺失",

"error": f"请求异常: {str(e)}"

}

# 合并代码执行

def main(disease: str, growth: str, temperature: str,

humidity: str, weather: str, windLevel: str,

location: str) -> dict:

"""

将七个参数按固定顺序拼接为字符串，用中文逗号分隔

参数:

disease: 疾病

growth: 生长期

temperature: 温度

humidity: 湿度

weather: 天气

windLevel: 风级

location: 地点

返回:

包含拼接结果的字典，如: {"result": "玉米斑病,拔节期,25°,75%,晴,2级,昌平"}

"""

# 拼接七个参数，用中文逗号分隔

result = f"{disease},{growth},{temperature},{humidity},{weather},{windLevel},{location}"

# 返回结果字典（Dify工作流要求返回字典）

return {

"result": result

}

# LLM追问

<instruction>

1. 任务目标：根据用户提供的参数缺失情况，生成对应的数据补充提示，严格禁止输出任何防治方案。

2. 处理逻辑：

- 第一优先级：检查"疾病"参数是否缺失或值为"数据缺失"

- 第二优先级：检查"生长期"参数是否缺失或值为"数据缺失"

- 第三优先级：检查"地点"参数是否缺失或值为"数据缺失"

- 第四优先级：当地点存在但温度/湿度任一缺失时触发

3. 输出要求：

- 必须使用中文双引号包裹完整回复内容

- 每个选项前使用特定符号标记（●/★/数字）

- 必须包含"示例输入"引导格式

- 禁止输出XML/HTML标签

4. 异常处理：

- 若多个参数同时缺失，按上述优先级顺序响应

- 若参数完整但值为空，视为数据缺失

</instruction>

<response\_template>

{{#if (not 疾病) or (eq 疾病 "数据缺失")}}

"需要补充病虫害信息：

请选择或描述以下症状：

● 玉米斑病（匹配词：大斑病/小斑病/叶片褐斑）

● 玉米锈病（匹配词：锈斑/铁锈色粉末）

● 玉米黑粉病（匹配词：黑粉包/瘤状物）

● 玉米根腐病（匹配词：烂根/根发黑）

● 玉米苗枯病（匹配词：死苗/苗基部腐烂）

● 玉米螟（匹配词：蛀孔/钻心虫）

● 玉米黏虫（匹配词：吃叶虫/叶片缺刻）

● 玉米蓟马（匹配词：银白色条纹）

示例输入：玉米叶片有锈斑"

{{#elseif (not 生长期) or (eq 生长期 "数据缺失")}}

"需要确认生长阶段：

可选阶段：

★ 苗期（匹配词：小苗/3-5片叶）

★ 拔节期（匹配词：茎秆伸长）

★ 抽雄期（匹配词：抽穗/吐丝）

★ 灌浆期（匹配词：籽粒乳熟）

★ 成熟期（匹配词：叶片发黄）

示例输入：拔节期"

{{#elseif (not 地点) or (eq 地点 "数据缺失")}}

"需要提供地理位置：

格式要求：

- 市级：北京市

- 区县级：昌平区

或说'使用当前位置'

示例输入：河南省周口市"

{{#elseif (and 地点 (neq 地点 "数据缺失") (or (not 温度) (not 湿度))}}

"气象数据获取失败

可选择操作：

1. 输入'重新获取天气'尝试刷新

2. 手动输入当前天气（格式：温度/湿度）

示例输入：28℃/65%"

{{/if}}

</response\_template>

<examples>

<example>

输入：{"疾病":"数据缺失"}

输出："需要补充病虫害信息：

请选择或描述以下症状：

● 玉米斑病（匹配词：大斑病/小斑病/叶片褐斑）

● 玉米锈病（匹配词：锈斑/铁锈色粉末）

● 玉米黑粉病（匹配词：黑粉包/瘤状物）

● 玉米根腐病（匹配词：烂根/根发黑）

● 玉米苗枯病（匹配词：死苗/苗基部腐烂）

● 玉米螟（匹配词：蛀孔/钻心虫）

● 玉米黏虫（匹配词：吃叶虫/叶片缺刻）

● 玉米蓟马（匹配词：银白色条纹）

示例输入：玉米叶片有锈斑"

</example>

<example>

输入：{"疾病":"玉米螟","生长期":""}

输出："需要确认生长阶段：

可选阶段：

★ 苗期（匹配词：小苗/3-5片叶）

★ 拔节期（匹配词：茎秆伸长）

★ 抽雄期（匹配词：抽穗/吐丝）

★ 灌浆期（匹配词：籽粒乳熟）

★ 成熟期（匹配词：叶片发黄）

示例输入：拔节期"

</example>

<example>

输入：{"地点":"数据缺失","温度":"30"}

输出："需要提供地理位置：

格式要求：

- 市级：北京市

- 区县级：昌平区

或说'使用当前位置'

示例输入：河南省周口市"

</example>

<example>

输入：{"地点":"昌平区","湿度":""}

输出："气象数据获取失败

可选择操作：

1. 输入'重新获取天气'尝试刷新

2. 手动输入当前天气（格式：温度/湿度）

示例输入：28℃/65%"

</example>

</examples>

<format\_specification>

1. 疾病提示格式：

- 标题："需要补充病虫害信息："

- 症状列表：每行以●开头，包含正式病名和括号内匹配词

- 示例后缀："示例输入："+典型症状描述

2. 生长期提示格式：

- 标题："需要确认生长阶段："

- 阶段列表：每行以★开头，包含阶段名和括号内匹配词

- 示例后缀："示例输入："+典型阶段名

3. 地点提示格式：

- 标题："需要提供地理位置："

- 格式说明：以"-"开头的条目化要求

- 特殊选项："或说'使用当前位置'"

- 示例后缀："示例输入："+标准地名

4. 气象提示格式：

- 标题："气象数据获取失败"

- 操作列表：数字编号的选项

- 格式说明："（格式：温度/湿度）"

- 示例后缀："示例输入："+标准格式数据

</format\_specification>

# LLM搜索

{{#1752113221567.disease#}}{{#1752113221567.growth#}}{{#1752113123348.temperature#}}{{#1752113123348.humidity#}}{{#1752113123348.weather#}}{{#1752113123348.windLevel#}}{{#1752113221567.location#}}

<xsl:stylesheet version="1.0">

<xsl:template match="/">

<strict\_rules>

1. 保持原始输出结构和顺序不变

2. 所有已知参数必须显示实际值

3. 缺失参数明确标注"数据缺失"

4. 保留完整的参数修改提示

</strict\_rules>

<!-- 知识库查询模块 -->

<knowledge\_query>

<xsl:variable name="candidates" select="知识库/农药[

病虫害名称=当前疾病 and

生长阶段=当前生长期

]"/>

<!-- 综合评估函数保持不变 -->

</knowledge\_query>

<!-- 完整输出模板 -->

<!-- 输出模板（去标签版） -->

<output\_template>

【最优方案】<xsl:value-of select="$best\_pesticide/农药名称"/>（<xsl:value-of select="$best\_pesticide/剂型"/>）

├─ 综合评分：<xsl:value-of select="format-number(综合评估($best\_pesticide), '0.0')"/>/100

├─ 经济生态评估：

│- 💰 价格：<xsl:value-of select="$best\_pesticide/价格"/>元/亩

│- ♻️ 毒性：<xsl:value-of select="$best\_pesticide/毒性等级"/>

└- ⭐ 效果：<xsl:value-of select="string-join(for $i in 1 to $best\_pesticide/效果评级 return '★','')"/>

【标准施用量】

├─ 农药用量：<xsl:value-of select="recommend\_dose"/>/亩（固定基准值）

├─ 天气适配：<xsl:value-of select="concat(天气,if(风级)then concat('/',风级,'级风') else '')"/>

└─ 技术参数：

• 稀释比例：1:<xsl:value-of select="$best\_pesticide/稀释倍数"/>

• 安全间隔期：<xsl:value-of select="$best\_pesticide/安全间隔期"/>天

💡 实用配比方案（按15升喷雾器计算）：

<xsl:choose>

<xsl:when test="contains($best\_pesticide/剂型, '悬浮剂')">

• 每桶加药 <xsl:value-of select="round(recommend\_dose div $best\_pesticide/稀释倍数 \* 15000)"/>毫升 ≈ <xsl:value-of select="round(recommend\_dose div $best\_pesticide/稀释倍数 \* 15000 div 5)"/>瓶盖（标准5ml瓶盖）

</xsl:when>

<xsl:when test="contains($best\_pesticide/剂型, '可湿性粉剂')">

• 每桶加药 <xsl:value-of select="round(recommend\_dose div $best\_pesticide/稀释倍数 \* 15000)"/>克 ≈ <xsl:value-of select="round(recommend\_dose div $best\_pesticide/稀释倍数 \* 15000 div 10)"/>汤匙（标准10g量匙）

</xsl:when>

</xsl:choose>

【候选药剂对比】

1. <xsl:value-of select="$candidates[1]/农药名称"/>（评分：<xsl:value-of select="format-number(综合评估($candidates[1]), '0.0')"/>）

2. <xsl:value-of select="$candidates[2]/农药名称"/>（评分：<xsl:value-of select="format-number(综合评估($candidates[2]), '0.0')"/>）

3. <xsl:value-of select="$candidates[3]/农药名称"/>（评分：<xsl:value-of select="format-number(综合评估($candidates[3]), '0.0')"/>）

🌾 请提供以下信息完成定制方案：

• 种植面积（示例：3亩/0.2公顷）

• 施药器械类型（背负式/拖拉机喷雾）

📝 参数修正指令：

当前设置：<xsl:value-of select="concat(疾病,' ',生长期,' ',温度,'℃ ',湿度,'% ',天气,' ',风级,'级 ',地点)"/>

修改示例："将湿度调整为85%" 或 "更改生长期为抽雄期"

支持药剂方案更换

修改示例：

”药品改为嘧菌酯“

</output\_template>

</xsl:template>

</xsl:stylesheet>

# LLM计算

```xml

<instruction>

根据已知的药物用量和输入的亩地数，计算总用药量。请按照以下步骤操作：

1. 确认已知的药物用量（单位：克/亩或毫升/亩）。

2. 获取用户输入的亩地数（单位：亩）。

3. 将药物用量与亩地数相乘，得到总用药量。

4. 确保输出结果的单位与药物用量的单位一致（例如，克或毫升）。

5. 输出结果时，仅提供数值和单位，不要包含任何XML标签或其他无关信息。

注意事项：

- 如果药物用量或亩地数为零或负数，需提示用户输入无效。

- 如果输入值包含非数字字符，需提示用户输入格式错误。

- 输出结果应简洁明了，避免冗余信息。

</instruction>

<examples>

<example>

输入：

- 药物用量：50克/亩

- 亩地数：10亩

输出：

500克

</example>

<example>

输入：

- 药物用量：100毫升/亩

- 亩地数：5亩

输出：

500毫升

</example>

<example>

输入：

- 药物用量：30克/亩

- 亩地数：0亩

输出：

输入无效，亩地数必须大于零。

</example>

</examples>

<additional\_notes>

1. 如果药物用量或亩地数为小数，计算结果应保留两位小数。

2. 如果用户未提供单位，默认使用“克”作为单位。

3. 确保计算过程准确无误，避免因四舍五入导致的误差。

</additional\_notes>

```

# LLM药品

<instruction>

请严格按以下规则处理八个参数：

1. 参数顺序固定为：

- 疾病: {{disease||"数据缺失"}}

- 生长期: {{growth||"数据缺失"}}

- 温度: {{temperature||"数据缺失"}}

- 湿度: {{humidity||"数据缺失"}}

- 天气: {{weather||"数据缺失"}}

- 风级: {{windLevel||"数据缺失"}}

- 地点: {{location||"数据缺失"}}

- 药品: {{pesticide||"数据缺失"}}

药品为输入中新的参数，请进行替换

2. 格式要求：

- 参数间用中文逗号分隔

- 保留原始单位符号（°、%、级等）

- 药品参数需保留完整名称（含剂型）

- 最终输出仅包含8个参数，不加任何说明

3. 特殊处理：

- 地点参数必须验证存在性

- 药品参数需验证有效性（非空字符串）

- 空值/null/undefined均显示为"数据缺失"

4. 示例：

<正常案例>

输入：

disease="玉米斑病"

growth="拔节期"

temperature="25°"

humidity="80%"

weather="多云"

windLevel="3级"

location="昌平"

pesticide="戊唑醇悬浮剂"

输出：

玉米斑病，拔节期，25°，80%，多云，3级，昌平，戊唑醇悬浮剂

<缺失案例>

输入：

disease=null

growth="抽穗期"

temperature="数据缺失"

pesticide=undefined

输出：

数据缺失，抽穗期，数据缺失，数据缺失，数据缺失，数据缺失，数据缺失，数据缺失

</instruction>

# LLM其他方案

{{#1752160952251.text#}}

<xsl:stylesheet version="1.0">

<xsl:template match="/">

<strict\_rules>

1. 保持原始输出结构和顺序不变

2. 所有已知参数必须显示实际值

3. 缺失参数明确标注"数据缺失"

4. 保留完整的参数修改提示

5. 明确标注为非最优方案

</strict\_rules>

<!-- 知识库查询模块 -->

<knowledge\_query>

<xsl:variable name="candidates" select="知识库/农药[

病虫害名称=当前疾病 and

生长阶段=当前生长期

]"/>

<!-- 综合评估函数保持不变 -->

</knowledge\_query>

<!-- 完整输出模板 -->

<!-- 输出模板（去标签版） -->

<output\_template>

【调整后方案（非最优）】<xsl:value-of select="$adjusted\_pesticide/农药名称"/>（<xsl:value-of select="$adjusted\_pesticide/剂型"/>）

⚠️ 注意：当前为人工调整方案，非系统推荐最优解

├─ 综合评分：<xsl:value-of select="format-number(综合评估($adjusted\_pesticide), '0.0')"/>/100

├─ 经济生态评估：

│- 💰 价格：<xsl:value-of select="$adjusted\_pesticide/价格"/>元/亩

│- ♻️ 毒性：<xsl:value-of select="$adjusted\_pesticide/毒性等级"/>

└- ⭐ 效果：<xsl:value-of select="string-join(for $i in 1 to $adjusted\_pesticide/效果评级 return '★','')"/>

【标准施用量】

├─ 农药用量：<xsl:value-of select="adjusted\_dose"/>/亩（调整后值）

├─ 天气适配：<xsl:value-of select="concat(天气,if(风级)then concat('/',风级,'级风') else '')"/>

└─ 技术参数：

• 稀释比例：1:<xsl:value-of select="$adjusted\_pesticide/稀释倍数"/>

• 安全间隔期：<xsl:value-of select="$adjusted\_pesticide/安全间隔期"/>天

💡 实用配比方案（按15升喷雾器计算）：

<xsl:choose>

<xsl:when test="contains($adjusted\_pesticide/剂型, '悬浮剂')">

• 每桶加药 <xsl:value-of select="round(adjusted\_dose div $adjusted\_pesticide/稀释倍数 \* 15000)"/>毫升 ≈ <xsl:value-of select="round(adjusted\_dose div $adjusted\_pesticide/稀释倍数 \* 15000 div 5)"/>瓶盖（标准5ml瓶盖）

</xsl:when>

<xsl:when test="contains($adjusted\_pesticide/剂型, '可湿性粉剂')">

• 每桶加药 <xsl:value-of select="round(adjusted\_dose div $adjusted\_pesticide/稀释倍数 \* 15000)"/>克 ≈ <xsl:value-of select="round(adjusted\_dose div $adjusted\_pesticide/稀释倍数 \* 15000 div 10)"/>汤匙（标准10g量匙）

</xsl:when>

</xsl:choose>

【系统推荐最优方案对比】

1. <xsl:value-of select="$best\_pesticide/农药名称"/>（评分：<xsl:value-of select="format-number(综合评估($best\_pesticide), '0.0')"/>）

- 优势：<xsl:value-of select="$best\_pesticide/优势说明"/>

【其他药剂】

1. <xsl:value-of select="$candidates[1]/农药名称"/>（评分：<xsl:value-of select="format-number(综合评估($candidates[1]), '0.0')"/>）

2. <xsl:value-of select="$candidates[2]/农药名称"/>（评分：<xsl:value-of select="format-number(综合评估($candidates[2]), '0.0')"/>）

3. <xsl:value-of select="$candidates[3]/农药名称"/>（评分：<xsl:value-of select="format-number(综合评估($candidates[3]), '0.0')"/>）

【当前环境参数】

• 病害：<xsl:value-of select="疾病"/>

• 生长期：<xsl:value-of select="生长期"/>

• 环境：<xsl:value-of select="concat(温度,'℃ ',湿度,'% ',天气,' ',风级,'级风')"/>

• 地点：<xsl:value-of select="地点"/>

📝 参数修正指令：

当前设置：<xsl:value-of select="concat(疾病,' ',生长期,' ',温度,'℃ ',湿度,'% ',天气,' ',风级,'级 ',地点)"/>

修改示例：

”地点改为海淀“

支持药剂方案更换

修改示例：

”药品改为嘧菌酯“

</output\_template>

</xsl:template>

# 前端部分：

创建一个响应式网页应用，界面风格简约而富有科技感，专注于玉米病虫害决策支持。主要页面应包含：

1. 用户输入区：设计直观的表单供用户填写病虫害类型、当前天气条件、温度范围等关键信息。

2. 提交按钮：点击后触发向服务器发送请求的动作，同时保持用户输入的数据以便于后续参考。

3. 结果展示区：动态更新以反映根据提供的参数计算得出的最佳防治策略。

4. 历史记录面板：允许用户浏览过去的查询记录，并提供选项删除不再需要的历史条目。

5. 上下文感知机制：当用户再次访问时，自动加载最近一次的会话数据，增强用户体验。

6. 使用深色调配色方案，辅以蓝色或绿色高亮，突出重要信息。

7. 集成本地存储功能，保存用户的查询历史，即使刷新页面也能保留这些数据。

8. 保证所有交互元素都具备良好的可访问性，考虑不同设备上的显示效果。