\*\*任务：\*\* 根据提供的聊天记录（通过 memotrace 爬取导出的 txt 文件，\*\*文件名应包含群聊名称\*\*），生成 \*\*指定日期（自动识别为当前运行日期的上一个完整日）\*\* 的 \*\*[自动识别的群名称]\*\* 群日报。日报将以 \*\*交互式 HTML 格式\*\* 输出，呈现为可 \*\*左右滑动切换的堆叠卡片\*\* 样式，每个主要内容板块对应一张卡片，进行 \*\*响应式设计\*\*，并在 \*\*第一张卡片（封面）包含一个“今日必看”精选栏目\*\*。

\*\*核心变化说明：\*\*

\* \*\*自动群名识别：\*\* \*\*尝试从输入的 `.txt` 文件名中提取群聊名称\*\*。例如，若文件名为 `AI交流群\_2025-04-21.txt`，则识别群名为 `AI交流群`。若无法识别，则使用通用名称（如“群聊”）。

\* \*\*自动日期识别：\*\* 脚本运行时自动确定报告日期为 \*\*执行时间戳对应的上一个完整日\*\*。当前时间是 \*\*2025年4月22日 10:23:29 (UTC+8)\*\*，因此应处理 \*\*2025年4月21日\*\* 的数据。

\* \*\*通用云朵词云：\*\* 词云样式修改为 \*\*通用的、美观的云朵形状\*\*。

\* \*\*替换封面摘要为“今日必看”：\*\* 第一张卡片展示包含 \*\*抓人眼球标题\*\* 和 \*\*5 条精选内容\*\* 的“今日必看”栏目。

\* \*\*交互式卡片布局：\*\* 水平滑动的堆叠卡片。

\* \*\*JavaScript 交互：\*\* 实现滑动逻辑，优化移动端滑动体验。

\* \*\*固定动画效果：\*\* 卡片切换动画固定为平滑水平滑动加淡入淡出/缩放。

\* \*\*响应式设计：\*\* 页面布局和样式能自适应桌面和移动设备。

\* \*\*PNG 导出说明：\*\* 主要输出为功能完整的交互式 HTML 文件。

---

## 输入数据处理与验证

\* 聊天记录通过 memotrace 爬取成 txt 文档导出，格式为：`[时间] 昵称：消息内容`。

\* \*\*文件名处理（新增）：\*\*

\* \*\*尝试从输入的 `.txt` 文件名中提取群聊名称\*\*。优先使用文件名中日期或常见分隔符（如 `\_`, `-`）之前的部分作为群名。

\* \*\*清理名称：\*\* 移除文件名中常见的日期格式（如 `YYYY-MM-DD`, `YYYYMMDD`）、导出标识（如 `\_ChatHistory`, `\_Export`）等非群名部分。

\* \*\*无法识别处理：\*\* 如果无法从文件名中可靠识别群名（例如文件名过于简单如 `chat.txt`），则使用通用名称，如 `"群聊"` 或 `"聊天记录"` 作为 `[自动识别的群名称]`，并在页脚添加注释说明。

\* \*\*数据预检：\*\*

\* 验证文件格式一致性。

\* 检测时间戳格式 (`[YYYY-MM-DD HH:MM:SS]` 或类似格式)，确保能基于此 \*\*自动筛选指定日期\*\* 的记录。

\* 识别消息边界，正确处理跨行消息。

\* 对不符合标准格式的行，尝试启发式解析或标记待审核。

\* \*\*链接处理：\*\* (详细规则见下方【链接处理规则】部分)

\* 识别 URL 模式，验证完整性，标记状态并呈现。

\* \*\*时间范围：\*\* \*\*自动处理【上一个完整日】\*\* 的聊天记录（\*\*2025年4月21日 00:00:00 至 23:59:59\*\*，基于当前时间推算）。

\* \*\*无数据处理：\*\* 如未找到指定日期的记录，在 \*\*第一张卡片\*\* 顶部显示 `"未找到 [报告日期] 的聊天记录"`（例如 "未找到 2025年4月21日 的聊天记录"），且“今日必看”栏目显示“今日无精彩内容”。

\* \*\*格式异常处理：\*\* 如发现记录格式异常，尝试合理解析并在 \*\*最后一张卡片 (页脚)\*\* 提供注释。

## 新增要求：“今日必看”栏目生成

\* \*\*替换原摘要：\*\* 第一张卡片不再生成 `<p class="report-summary">`。

\* \*\*生成“今日必看”栏目：\*\* 在第一张卡片的标题和日期下方，生成 `<div class="must-read">...</div>`。

\* \*\*创作抓人眼球的标题：\*\* 在 `<div class="must-read">` 内生成 `<h3 class="must-read-title">...</h3>`，由 \*\*AI 根据当天内容亮点创作\*\*，风格引人注目。

\* \*\*精选 5 条内容：\*\* 在标题下方生成 `<ul class="must-read-list">...</ul>`，从“热点”、“重要消息”、“金句”中精选 5 条信息点，放入 `<li class="must-read-item">...</li>`。

\* \*\*固定格式与样式：\*\* 每条精选内容用 `<strong><em>...</em></strong>` 包裹（粗体+斜体），并应用醒目颜色（如 `#f25f4c` 或 `#e53170`）。

## HTML 输出格式与结构 (卡片式, 响应式)

请生成一个完整的静态 HTML 文档，包含：

1. 完整的 `<!DOCTYPE html>`, `<html>`, `<head>`, `<body>` 标签。

2. `<head>` 部分包含：

\* `<meta charset="UTF-8">`

\* `<title>[自动识别的群名称]日报 - [报告日期] (卡片版)</title>` （\*\*自动填充群名和日期\*\*）

\* `<meta name="viewport" content="width=device-width, initial-scale=1.0">`

\* \*\*内联 CSS 样式表 `<style>...</style>`\*\*：包含所有页面、卡片堆叠、固定滑动动画、响应式媒体查询及内部内容的样式。

3. `<body>` 部分包含：

\* \*\*主容器 `<div class="card-stack-container">`\*\*。

\* \*\*多个卡片元素 `<div class="report-card" id="card-N">`\*\*。

\* \*\*导航控件\*\*（左右箭头和底部圆点，如果卡片多于一张）。

4. \*\*JavaScript 代码 `<script>...</script>`\*\*：放在 `<body>` 结尾，原生 JS。

## 页面设计与样式要求 (卡片式, 响应式)

1. \*\*基础设计风格：\*\*

\* 配色方案：深色背景 (`#0f0e17`)，明亮主文本 (`#fffffe`)。强调色 (`#ff8906`)，副色调 (`#f25f4c`, `#e53170`, `#3da9fc`, `#7209b7`, `#00b4d8`, `#4cc9f0` 等)。

\* 字体：无衬线字体 (如 'Inter', 'Roboto', 'Helvetica Neue', sans-serif)，代码等宽字体。

\* \*\*容器宽度 (响应式)：\*\* 桌面端 `max-width: 800px; margin: 0 auto;`，移动端 `width: 95vw; margin: 0 auto;`。高度 `min-height: 500px; height: 80vh;`。

2. \*\*布局结构 (卡片堆叠)：\*\*

\* \*\*堆叠效果：\*\* 使用 CSS (`position: absolute`, `transform`, `z-index`, `opacity`)。

\* \*\*卡片状态定义 (固定效果):\*\* `.active`, `.next`, `.prev` 类控制 `transform: translateX()`、`opacity`、`scale()`、`z-index`、`pointer-events`。

\* \*\*卡片内容组织：\*\* 每个 `.report-card` 有内边距 (`padding: 25px` / `15px`)，内容过长时 `overflow-y: auto;`。

\* \*\*卡片顺序 (建议)：\*\* 1.封面 -> 2.热点 -> 3.教程/资源 -> 4.重要消息 -> 5.金句/对话 -> 6.问答 -> 7.数据可视化 -> 8.词云/页脚。

3. \*\*组件样式 (卡片内部)：\*\*

\* \*\*卡片本身：\*\* 背景色 (`#1a1924`)，圆角 (`border-radius: 10px;`)，阴影 (`box-shadow: 0 8px 25px rgba(0,0,0,0.2);`)。

\* \*\*“今日必看”样式：\*\* `.must-read`, `.must-read-title`, `.must-read-list`, `.must-read-item`, `.must-read-item strong em` 样式按要求定义。

\* \*\*内部元素 (响应式)：\*\* `h1`, `h2`, `h3`, `p`, `ul`, `li`, `a.link` (含 `valid`, `warning`, `invalid` 类), `table`, `blockquote`, `.keyword`, `.tag`, `.characteristic`, `.word`, `.priority-\*`, `.heat-bar`, `.heat-fill` 等样式按要求定义。

4. \*\*交互与动画 (固定效果 & 移动优化)：\*\*

\* \*\*切换方式：\*\* 左右触摸滑动，点击导航箭头/指示点。

\* \*\*固定动画效果：\*\* 平滑水平滑动 + 透明度/缩放变化 (`transition: transform 0.5s cubic-bezier(0.25, 0.8, 0.25, 1), opacity 0.5s ease-out;`)。

\* \*\*视觉反馈：\*\* 活动卡片及导航状态同步更新。

5. \*\*单卡片截图优化：\*\* 内容直接可见（卡片内可滚动），对比度高，字体大小适中 (>=14px)，布局得当。

## 内容区块与卡片映射 (结构示例)

\* \*\*`<div class="report-card" id="card-1">` (封面、今日必看与概览)\*\*

\* 包含 `页面头部` (H1 `[自动识别的群名称]日报`, P 日期 `[报告日期]` - \*\*自动填充\*\*)。

\* \*\*新增：`今日必看` 栏目\*\* (`<div class="must-read">` 含标题和 5 个 `<li><strong><em>...</em></strong></li>` 项)。

\* 包含 `元数据` (`<div class="meta-info">`)。

\* \*\*`<div class="report-card" id="card-2">` (今日讨论热点)\*\*

\* 包含 `<section class="hot-topics">...</section>`

\* \*\*`<div class="report-card" id="card-3">` (实用教程与AI热点)\*\*

\* 包含 `<section class="tutorials">...</section>`

\* \*\*`<div class="report-card" id="card-4">` (重要消息汇总)\*\*

\* 包含 `<section class="important-messages">...</section>`

\* \*\*`<div class="report-card" id="card-5">` (有趣对话或金句)\*\*

\* 包含 `<section class="interesting-dialogues">...</section>`

\* \*\*`<div class="report-card" id="card-6">` (问题与解答)\*\*

\* 包含 `<section class="questions-answers">...</section>`

\* \*\*`<div class="report-card" id="card-7">` (群内数据可视化)\*\*

\* 包含 `<section class="analytics">...</section>`

\* \*\*`<div class="report-card" id="card-8">` (AI话题词云与页脚)\*\*

\* 包含 `<section class="word-cloud">...</section>` (\*\*通用云朵样式\*\*)

\* 包含 `<footer>...</footer>`

## 各内容区块具体要求 (填充至对应卡片)

\*(请严格按照详细 HTML 结构、数据项和样式类名要求进行生成，填充至对应 `.report-card`。注意自动填充群名和日期，卡片 1 的新增要求，卡片 8 词云样式变更)\*

1. \*\*封面、今日必看与概览 (放入卡片 1)\*\*: 使用 `<h1>[自动识别的群名称]日报</h1>`, `<p class="date">[报告日期]</p>` (\*\*自动填充\*\*), \*\*`<div class="must-read">...</div>` (含 AI 生成标题和 5 条精选内容)\*\*, `<div class="meta-info">...</div>` 结构。

2. \*\*今日讨论热点 (放入卡片 2)\*\*: 使用 `<section class="hot-topics">...<div class="topic-card">...</div>...</section>` 结构。

3. \*\*实用教程与AI热点 (放入卡片 3)\*\*: 使用 `<section class="tutorials">...<div class="tutorial-card">...</div>...</section>` 结构。

4. \*\*重要消息汇总 (放入卡片 4)\*\*: 使用 `<section class="important-messages">...<div class="message-card">...</div>...</section>` 结构。

5. \*\*有趣对话或金句 (放入卡片 5)\*\*: 使用 `<section class="interesting-dialogues">...<div class="dialogue-card">...</div>...</section>` 结构。

6. \*\*问题与解答 (放入卡片 6)\*\*: 使用 `<section class="questions-answers">...<div class="qa-card">...</div>...</section>` 结构。

7. \*\*群内数据可视化 (放入卡片 7)\*\*: 使用 `<section class="analytics">...` 包含 \*\*话题热度\*\* (`.topic-heatmap`), \*\*话唠榜\*\* (`.top-participants`), \*\*熬夜冠军\*\* (`.night-owls`) 的精确结构。

8. \*\*词云 (通用云朵样式) (放入卡片 8)\*\*: 使用 `<section class="word-cloud"><h2>AI话题词云</h2><div class="word-cloud-container"><div class="word-cloud-area"...>...</div></div></section>` 结构，内部词汇 \*\*绝对定位\*\* (`<span class="cloud-word" style="...">词汇</span>`)。\*\*(详细规则见下方【词云数据生成要求】)\*\*

9. \*\*页面底部信息 (放入卡片 8)\*\*: 使用 `<footer><p>数据来源：[自动识别的群名称]聊天记录...</p><p>生成时间：[当前完整时间]</p><p>统计周期：[报告日期] 00:00 - 23:59</p><p class="disclaimer">免责声明...</p></footer>` 结构。

## CSS 样式要求 (内联 `<style>` 中，包含响应式)

```css

/\* 全局与卡片容器样式 \*/

body { background-color: #0f0e17; font-family: 'Inter', 'Roboto', 'Helvetica Neue', sans-serif; color: #fffffe; margin: 0; line-height: 1.6; }

.card-stack-container { position: relative; width: 95vw; max-width: 800px; height: 80vh; min-height: 500px; margin: 5vh auto; overflow: hidden; border-radius: 10px; }

/\* 卡片样式 (含固定动画) \*/

.report-card { position: absolute; top: 0; left: 0; width: 100%; height: 100%; background-color: #1a1924; border-radius: 10px; box-shadow: 0 8px 25px rgba(0,0,0,0.2); padding: 25px; overflow-y: auto; box-sizing: border-box; transition: transform 0.5s cubic-bezier(0.25, 0.8, 0.25, 1), opacity 0.5s ease-out; }

.report-card.active { transform: translateX(0); opacity: 1; z-index: 10; pointer-events: auto; }

.report-card.next { transform: translateX(100%) scale(0.95); opacity: 0.6; z-index: 9; pointer-events: none; }

.report-card.prev { transform: translateX(-100%) scale(0.95); opacity: 0.6; z-index: 9; pointer-events: none; }

.report-card:not(.active):not(.next):not(.prev) { transform: scale(0.9); opacity: 0; z-index: 8; pointer-events: none; } /\* 默认隐藏 \*/

/\* 卡片内部组件样式 \*/

h1, h2, h3 { color: #ff8906; margin-top: 0; }

h1 { font-size: 28px; text-align: center; margin-bottom: 5px;}

h2 { font-size: 22px; border-bottom: 1px solid #444; padding-bottom: 8px; margin-bottom: 20px; }

h3 { font-size: 18px; color: #3da9fc; margin-bottom: 10px; }

p { color: #e0e0e0; margin-bottom: 15px; }

.date { text-align: center; color: #a0a0a0; font-size: 14px; margin-bottom: 20px; }

.meta-info { text-align: center; font-size: 12px; color: #888; margin-top: 10px; border-top: 1px solid #444; padding-top: 10px; }

.meta-info span { margin: 0 10px; }

.must-read { margin: 20px 0; padding: 15px; background-color: rgba(255, 255, 255, 0.05); border-radius: 8px; }

.must-read-title { font-size: 20px; color: #ff8906; text-align: center; margin-bottom: 15px; font-weight: 600; }

.must-read-list { list-style: none; padding-left: 0; margin-bottom: 0; }

.must-read-item { margin-bottom: 10px; line-height: 1.5; }

.must-read-item strong em { font-weight: bold; font-style: italic; color: #f25f4c; }

ul { padding-left: 20px; margin-bottom: 15px; }

li:not(.must-read-item) { margin-bottom: 8px; }

a.link { text-decoration: none; }

a.link.valid { color: #3da9fc; }

a.link.warning { color: #ff8906; }

a.link.invalid { color: #888888; }

a.link:hover { text-decoration: underline; }

.keyword, .tag, .characteristic, .word { background-color: #3a0ca3; color: #ffffff; padding: 3px 8px; border-radius: 4px; font-size: 12px; display: inline-block; margin-right: 5px; margin-bottom: 5px; }

blockquote { background-color: #2a2934; border-left: 4px solid #ff8906; padding: 10px 15px; margin: 15px 0; font-style: italic; }

.priority-high { color: #f25f4c; font-weight: bold; } .priority-medium { color: #ff8906; } .priority-low { color: #888888; }

.heat-bar { background-color: #333; height: 10px; border-radius: 5px; overflow: hidden; margin-top: 5px; }

.heat-fill { height: 100%; border-radius: 5px; /\* AI应根据热度设置宽度和背景色 \*/ }

.cloud-word { display: inline-block; padding: 2px 5px; }

footer { margin-top: 30px; border-top: 1px solid #444; padding-top: 15px; font-size: 12px; color: #888; }

footer p { margin-bottom: 5px; }

.generation-time { font-style: italic; }

.disclaimer { margin-top: 10px; font-style: italic; }

.footer-note { font-style: italic; font-size: 11px; color: #666; } /\* 用于群名识别失败注释 \*/

/\* 导航控件样式 \*/

.nav-arrow { position: absolute; top: 50%; transform: translateY(-50%); background-color: rgba(255,255,255,0.1); color: #fff; border: none; border-radius: 50%; width: 40px; height: 40px; font-size: 20px; cursor: pointer; z-index: 15; transition: background-color 0.3s; user-select: none; }

.nav-arrow:hover { background-color: rgba(255,255,255,0.3); }

.nav-arrow.prev { left: 10px; }

.nav-arrow.next { right: 10px; }

.nav-dots { position: absolute; bottom: 15px; left: 50%; transform: translateX(-50%); display: flex; z-index: 15; }

.dot { width: 10px; height: 10px; background-color: rgba(255,255,255,0.3); border-radius: 50%; margin: 0 5px; cursor: pointer; transition: background-color 0.3s; }

.dot.active { background-color: #ffffff; }

/\* 响应式设计 \*/

@media (max-width: 768px) {

.card-stack-container { height: 85vh; }

.report-card { padding: 15px; }

h1 { font-size: 24px; }

h2 { font-size: 20px; }

h3 { font-size: 16px; }

.must-read-title { font-size: 18px; }

p, li:not(.must-read-item), .must-read-item { font-size: 14px; }

.nav-arrow { width: 35px; height: 35px; font-size: 18px; }

.nav-arrow.prev { left: 5px; }

.nav-arrow.next { right: 5px; }

.dot { width: 8px; height: 8px; margin: 0 4px; }

.word-cloud-area { width: 95%; height: 250px; }

}

@media (max-width: 480px) {

h1 { font-size: 22px; }

h2 { font-size: 18px; }

.must-read-title { font-size: 17px; }

.word-cloud-area { height: 200px; }

}

**JavaScript 交互要求 (**<script> **中，原生JS，含移动优化)**

JavaScript

<script>

document.addEventListener('DOMContentLoaded', () => {

const container = document.querySelector('.card-stack-container');

if (!container) { console.error("Card container not found."); return; }

const cards = Array.from(container.querySelectorAll('.report-card'));

const prevButton = container.querySelector('.nav-arrow.prev');

const nextButton = container.querySelector('.nav-arrow.next');

const dotsContainer = container.querySelector('.nav-dots');

let currentCardIndex = 0;

let isAnimating = false;

let touchStartX = 0;

let touchEndX = 0;

const swipeThreshold = 50; // Minimum pixels to swipe for action

function createDots() {

if (!dotsContainer || cards.length <= 1) {

if(dotsContainer) dotsContainer.style.display = 'none'; // Hide dots if 1 or 0 cards

return;

}

dotsContainer.innerHTML = ''; // Clear existing dots

dotsContainer.style.display = 'flex'; // Ensure dots are visible

cards.forEach((\_, index) => {

const dot = document.createElement('button');

dot.classList.add('dot');

dot.setAttribute('aria-label', `Go to card ${index + 1}`);

if (index === currentCardIndex) {

dot.classList.add('active');

}

dot.addEventListener('click', () => {

if (!isAnimating && index !== currentCardIndex) {

showCard(index);

}

});

dotsContainer.appendChild(dot);

});

}

function updateDots() {

if (!dotsContainer || cards.length <= 1) return;

const dots = dotsContainer.querySelectorAll('.dot');

dots.forEach((dot, index) => {

dot.classList.toggle('active', index === currentCardIndex);

});

}

function showCard(index) {

if (isAnimating || index < 0 || index >= cards.length) return;

isAnimating = true;

const previousActiveCard = cards[currentCardIndex]; // Store ref to previous card

currentCardIndex = index;

cards.forEach((card, i) => {

card.classList.remove('active', 'prev', 'next');

// Assign new classes based on the target index

if (i === currentCardIndex) {

card.classList.add('active');

// Focus management for accessibility: focus the new card

setTimeout(() => card.focus({ preventScroll: true }), 0); // preventScroll might be needed

} else if (i < currentCardIndex) {

card.classList.add('prev');

} else { // i > currentCardIndex

card.classList.add('next');

}

});

// Hide/show navigation buttons at boundaries

if(prevButton) prevButton.style.display = (currentCardIndex === 0 || cards.length <= 1) ? 'none' : 'block';

if(nextButton) nextButton.style.display = (currentCardIndex === cards.length - 1 || cards.length <= 1) ? 'none' : 'block';

updateDots();

// Use transitionend event for more reliable animation completion detection

const activeCard = cards[currentCardIndex];

if (activeCard) {

const handleTransitionEnd = (event) => {

// Ensure the event fired on the card itself and for the transform property

if (event.target === activeCard && event.propertyName === 'transform') {

isAnimating = false;

activeCard.removeEventListener('transitionend', handleTransitionEnd);

}

};

activeCard.addEventListener('transitionend', handleTransitionEnd);

// Fallback timeout in case transitionend doesn't fire reliably

setTimeout(() => {

if (isAnimating) { // If transitionend hasn't fired yet

// console.warn("Transitionend fallback triggered.");

isAnimating = false;

activeCard.removeEventListener('transitionend', handleTransitionEnd); // Clean up listener just in case

}

}, 550); // Slightly longer than transition duration

} else {

// Should not happen if cards exist, but as a safe fallback

setTimeout(() => {

isAnimating = false;

}, 550);

}

}

function nextCard() {

if (currentCardIndex < cards.length - 1) {

showCard(currentCardIndex + 1);

}

}

function prevCard() {

if (currentCardIndex > 0) {

showCard(currentCardIndex - 1);

}

}

// Initial setup

if (cards.length > 0) {

// Set initial classes without transition for first load

cards.forEach((card, i) => {

if (i === 0) card.classList.add('active');

else card.classList.add('next'); // Start all others as 'next'

});

createDots(); // Create dots after initial classes set

// Manually trigger showCard logic for button visibility and initial state

if(prevButton) prevButton.style.display = 'none'; // First card, hide prev

if(nextButton) nextButton.style.display = (cards.length === 1) ? 'none' : 'block'; // Hide next if only 1 card

cards[0].focus({ preventScroll: true }); // Focus first card initially

} else {

console.warn("No report cards found.");

if(dotsContainer) dotsContainer.style.display = 'none';

if(prevButton) prevButton.style.display = 'none';

if(nextButton) nextButton.style.display = 'none';

// Optionally display a message in the container if no cards

// container.innerHTML = '<p style="color: #fff; text-align: center; padding: 50px;">No report content available.</p>';

}

// Event Listeners

if (prevButton) {

prevButton.addEventListener('click', () => {

if (!isAnimating) prevCard();

});

}

if (nextButton) {

nextButton.addEventListener('click', () => {

if (!isAnimating) nextCard();

});

}

// Keyboard navigation for the container

container.addEventListener('keydown', (e) => {

if (isAnimating) return;

let handled = false;

if (e.key === 'ArrowLeft' || e.key === 'ArrowUp') { // Treat Up arrow like Left

prevCard();

handled = true;

} else if (e.key === 'ArrowRight' || e.key === 'ArrowDown') { // Treat Down arrow like Right

nextCard();

handled = true;

} else if (e.key === 'Home') { // Go to first card

if (currentCardIndex !== 0) showCard(0);

handled = true;

} else if (e.key === 'End') { // Go to last card

if (currentCardIndex !== cards.length - 1) showCard(cards.length - 1);

handled = true;

}

if(handled) {

e.preventDefault(); // Prevent page scroll if arrow keys were used for navigation

}

});

// Touch listeners

let touchStartY = 0; // Store Y start for scroll detection

container.addEventListener('touchstart', (e) => {

// Allow touch events on interactive elements within the card

if (e.target.closest('a, button, input, textarea, select')) {

return;

}

if (isAnimating) return;

touchStartX = e.touches[0].clientX;

touchStartY = e.touches[0].clientY; // Record Y position

touchEndX = touchStartX; // Reset endX

}, { passive: true });

container.addEventListener('touchmove', (e) => {

if (isAnimating || touchStartX === 0) return; // Don't track if not started or animating

touchEndX = e.touches[0].clientX;

const touchEndY = e.touches[0].clientY;

// Basic check to prioritize horizontal swipe over vertical scroll

const deltaX = Math.abs(touchEndX - touchStartX);

const deltaY = Math.abs(touchEndY - touchStartY);

if (deltaX > deltaY + 10) { // Horizontal movement is significantly larger

// Potential place to add e.preventDefault() if listener wasn't passive

// console.log("Horizontal swipe detected during move");

}

}, { passive: true }); // Keep passive if not preventing default

container.addEventListener('touchend', () => {

if (isAnimating || touchStartX === 0) return; // Exit if animating or no touch start

const swipeDistance = touchEndX - touchStartX;

// Also consider vertical distance to avoid triggering on vertical scrolls

// We don't have touchEndY here directly without storing it in touchmove

// Use swipeDistance threshold as primary detector for now

if (Math.abs(swipeDistance) > swipeThreshold) {

// Optional: Add a check here if deltaY was large, maybe don't swipe. Needs state from touchmove.

if (swipeDistance < 0) {

nextCard();

} else {

prevCard();

}

}

// Reset coordinates

touchStartX = 0;

touchEndX = 0;

touchStartY = 0;

});

// Make container focusable for keyboard nav

container.setAttribute('tabindex', '0');

container.style.outline = 'none'; // Hide focus outline if desired

});

</script>

**词云数据生成要求 (静态生成 - 通用云朵样式)**

1. **词汇选择原则**：核心 AI 词汇 (>=2次提及)，特色/新兴术语 (>=1次)，常见行业术语，当日热点。
2. **词汇布局 (通用云朵样式)**：
   * **形状**：自然、蓬松的云朵轮廓。
   * **分布**：高频/重要词汇更大、更中心；低频词汇较小、分布外围。避免严重重叠，允许自然交错。
   * **容器**：使用 .word-cloud-container 和 .word-cloud-area。
3. **权重与样式对应**：
   * 权重高 -> 字体大 (18px-42px)，颜色用亮色/强调色 (#ff8906, #f25f4c, #4cc9f0)，可加粗 (font-weight: bold;)。
   * 权重低 -> 字体小 (12px-17px)，颜色用次要/稍暗色 (#3da9fc, #7209b7, #a0a0a0)。
   * 部分词汇随机小角度旋转 (transform: rotate(Zdeg); Z 在 -15 到 15)。
4. **位置坐标计算 (静态)**：为每个 <span class="cloud-word"> 生成具体 style 属性 (position: absolute; left: Xpx; top: Ypx; font-size: SZpx; color: #HEX; transform: rotate(Zdeg); font-weight: FW;)，构成云朵。
5. **背景与图例：** 无需特定 SVG 背景或图例。

**链接处理规则**

1. **链接文本展示**：显示链接文本和清晰 URL，长 URL 可截断。使用 target="\_blank" rel="noopener noreferrer"。
2. **链接状态标记**：
   * **有效:** <a ... class="link valid">...</a> (#3da9fc)。
   * **需登录/付费:** 加注 (需登录)/(付费)，用 <a ... class="link warning">...</a> (#ff8906)。
   * **可能无效:** 加注 (可能无效)，用 <a ... class="link invalid">...</a> (#888888)。

**熬夜冠军评选规则**

* **定义熬夜时段：** 当日 00:00 至 06:00 (基于统一时区判断)。
* **评选标准：** 最晚发言者优先；同时间或接近时，参考发言数；可考虑连续性。
* **时区处理：** **必须统一时区** (建议 UTC+8) 后判断。若无法统一，需备注偏差。
* **结果呈现：** 选 1 位，在 .night-owls 中展示：👑 + 称号 + 最晚时间 (HH:MM) + 消息数 + 最后消息 + 特殊背景 + 时区说明。

**敏感内容处理**

* **过滤词汇：** 自动过滤/替换政治、网络规避工具等敏感词为 "[\*\*\*]"。
* **免责声明：** 页脚 **必须包含** <p class="disclaimer">免责声明：...</p>。

**使用前须知 (Prerequisites before use):**

* **文件名约定：** 请确保您提供的 .txt 聊天记录文件名包含清晰可识别的群聊名称（例如 AI交流群\_2025-04-21.txt 或 MyProjectGroup-ChatLog.txt）。脚本将尝试从此文件名自动提取群名。如果文件名不规范或无法识别，报告中将使用通用名称（如“群聊”）。
* **日期处理：** 报告日期将自动设为运行脚本时的上一个完整日（当前脚本运行时将处理 **2025年4月21日** 的数据）。
* **无需手动修改群名：** 此版本提示词已配置为自动识别群名，无需再手动替换占位符。