**SelfPacedStudy – WP3: Proactivity Plan & Summary of Key Questions**

**Responsible: Zhang Yang**

**1. Task Overview**

The core goal of this task is to enhance SAM (the learning assistant)'s interactivity by providing timely, personalized learning support based on learners' behavior patterns and course difficulty ratings. This will help improve student engagement and optimize their learning experience.

**2. Plan**

**Clarify Requirements and Detail the Task**

* Discuss task details with the team, clarify specific requirements, ensure consistency with development direction and architecture, and confirm interface requirements.

**Data Processing and Difficulty Rating Model Design**

* Analyze slides and video transcripts, design and implement an initial difficulty rating model to assess the difficulty of course content.

**Develop User Behavior Tracking Module**

* Develop a video interaction data collection module (e.g., pause, rewind, fast-forward) to log and store these interactions for later analysis.

**Interaction Trigger Logic Design**

* Determine when to trigger SAM's proactive interaction based on learner behavior (e.g., when a student rewinds the video, SAM could ask related questions).

**Interaction Frequency Adjustment & Pre-test Feedback Integration**

* Implement an interaction frequency adjustment feature (low, medium, high) and integrate pre-test questionnaire data into SAM’s interaction logic for personalized feedback.

**Design Socratic Questioning**

* Implement Socratic questioning techniques in SAM’s feedback to guide students towards deeper thinking rather than just providing answers.

**Advanced Logging and Data Storage**

* Build an advanced logging system that supports exporting logs in PDF and JSON formats, recording interaction history, user behavior, timestamps, etc.

**Testing and Optimization**

* Conduct comprehensive testing, fix bugs, and optimize code in preparation for the final demo.

**3. Key Questions for Clarification**

To ensure smooth progress, the following key questions need to be addressed by the team and supervisor:

1. **Difficulty Rating Model**
   * Are there existing scoring standards or sample courses for reference?
   * What factors will the difficulty rating be based on? Video content, language difficulty, or student feedback?
2. **User Behavior Tracking**
   * Besides pause, rewind, fast-forward, are there other behaviors to track that could trigger SAM’s interaction?
   * Are there additional data sources available, such as historical learning data or course participation details?
3. **Interaction Mechanism Design**
   * When triggering interactions, should SAM pop up as a prompt or appear as a dialog box? Can the message content be customized?
   * Are there preset interaction templates, or should we create interactions based on the content?
4. **Questionnaire Data Format & Storage**
   * Is there an existing database design for storing questionnaire data? How will the data integrate with the interaction mechanism?
   * How should we handle the data collected from questionnaires? Should it be integrated with user behavior data (e.g., pauses, rewinds)?
5. **Logging Storage**
   * What specific information needs to be recorded (e.g., interaction time, student behavior, SAM responses)? Are there any special storage format requirements?
   * Should the logs be exported in formats other than PDF and JSON, such as CSV or XML?

**4. Current Progress**

* Reviewed WP3 task description and organized a detailed plan with clear task objectives.
* Analyzing the source code to explore how to optimize the model for better interaction efficiency and ensure compatibility and scalability.
* Completed the initial design of the difficulty rating model and preparing to test the first version.

**5. Next Steps**

* **Discuss and clarify questions with the team**: Continue discussions with team members to clarify the key issues and ensure alignment on task objectives and interface requirements.
* **Deepen model optimization**: Continue refining the difficulty rating model and speed up development of the user behavior tracking module.
* **Modular development**: Start with developing user behavior tracking and interaction trigger logic, followed by interaction frequency adjustment and pre-test feedback integration.
* **Testing and feedback**: Conduct small-scale testing during development, gather feedback, and make necessary adjustments.

**6. Risks & Challenges**

* **Data Accuracy Issues**: There may be inaccuracies in collecting and analyzing student behavior data (such as pauses, rewinds). Continuous optimization of data processing is necessary.
* **Complexity of Interaction Frequency Adjustment**: Adjusting SAM’s interaction frequency based on learner needs to ensure it does not disrupt the learning process is a technical challenge.
* **Model Adaptability**: SAM must adjust feedback based on different students' progress and interaction frequency, requiring substantial personalization and optimization.

**第1步**：先实现**后端数据处理**和**交互触发机制**，确保能够根据学生行为（如视频暂停、回放）和课程难度评分来主动干预学习过程。

* 在此阶段，你需要：
  + 设计和实现**难度评分模型**，为每部分内容分配难度。
  + 开发**学生行为追踪模块**，确保行为数据能够准确记录到日志中。
  + 设计**交互触发机制**，如根据学生行为自动触发问题或提示。

**第2步**：**交互频率选择界面**，提供给学生选择频率的功能。这涉及到前端的改动，需要和前端团队协作实现。

**第3步**：集成**问卷数据**，确保SAM能够根据学生的前测反馈调整互动策略。这可能需要和WP2协作，确保数据能顺利传递。

**第4步**：实现**Socratic Questioning**，并根据反馈优化SAM的响应机制。

**第5步**：设计并实现**日志保存**功能，包括导出PDF和JSON。这部分工作完成后，前端只需要提供导出按钮，调用后端API即可。

**1. 基于 OpenAI API 完成难度评分模型**

在 **WP2** 团队实现离线 LLM 后，你将可以切换到本地化模型，但现在可以依赖 OpenAI API 来进行交互。

**a. 设计难度评分模型**

首先，设计一个简单的 **难度评分模型**，它可以利用 OpenAI API 分析每个部分的难度。你可以使用 OpenAI 模型（例如 GPT-3 或 GPT-4）来生成内容的难度评分。

**如何实现：**

* **输入：** 每个课程部分的幻灯片和转录文本。
* **输出：** 模型为每个部分生成一个难度评分（例如：1-5 级评分，1 为易，5 为难）。

**代码示例：**

javascript

复制编辑

// 使用 OpenAI API 获取文本难度评分

const openai = require("openai");  // 确保已安装 openai SDK

const apiKey = 'YOUR\_OPENAI\_API\_KEY'; // 替换为你的 OpenAI API 密钥

const model = 'text-davinci-003'; // 使用合适的 GPT-3 模型

openai.apiKey = apiKey;

async function getDifficultyScore(text) {

  const prompt = `Based on the following text, rate the difficulty of the material on a scale from 1 to 5 (1 = easy, 5 = very hard):\n\n${text}`;

  const response = await openai.completions.create({

    model: model,

    prompt: prompt,

    max\_tokens: 60,

  });

  const difficultyScore = response.choices[0].text.trim();

  return difficultyScore;

}

**b. 将难度评分结果存储和使用**

* **存储评分：** 将每个部分的评分存储在数据库中，供后续行为分析和交互策略使用。
* **使用评分：** 根据评分决定何时进行主动干预。难度较高的部分可以触发更频繁的互动。

**2. 行为追踪与数据记录**

在 WP3 的任务中，行为追踪是非常重要的。你需要记录学生在视频中的行为（例如：暂停、回放等）并基于这些行为来确定何时进行交互。

**a. 行为追踪代码实现**

你可以编写代码来追踪学生在观看视频时的行为，并将其与 **OpenAI** 模型的反馈结合起来。

**代码示例：**

javascript

复制编辑

// 示例：视频交互行为追踪（暂停、回放）

let videoElement = document.getElementById("video");

let pauseTime = 0;

let rewindedTime = 0;

videoElement.addEventListener("pause", () => {

  pauseTime = videoElement.currentTime;

  console.log(`Video paused at: ${pauseTime}`);

  // 记录行为数据到日志系统

  trackBehavior('pause', pauseTime);

});

videoElement.addEventListener("seeked", () => {

  rewindedTime = videoElement.currentTime;

  console.log(`Video rewound to: ${rewindedTime}`);

  // 记录行为数据到日志系统

  trackBehavior('rewind', rewindedTime);

});

function trackBehavior(action, time) {

  // 将行为和时间戳存储到数据库或日志中

  const behaviorData = {

    action: action,

    time: time,

    timestamp: new Date().toISOString()

  };

  console.log(behaviorData);

  // 可以在此处将数据发送到服务器进行处理

}

**b. 使用行为数据触发互动**

* 当学生暂停或回放视频时，你可以使用这些行为数据触发与学生的互动，尤其是基于模型生成的反馈。
* 例如，当学生回放某个部分时，SAM 可以提出与该部分相关的问题，帮助学生深入思考。

**3. 选择交互频率和控制**

在 WP3 中，你需要提供一个选项，让学生选择交互的频率（例如：高、低、中）。你可以根据 **OpenAI API** 的反馈以及学生的选择来调节互动频率。

**交互频率选择界面**

提供一个简单的界面让学生选择他们希望的交互频率。

**代码示例：**

javascript

复制编辑

// 在前端创建交互频率选择按钮

const interactionLevels = ["High", "Medium", "Low"];

let selectedLevel = "Medium";  // 默认选项

const interactionSelect = document.createElement("select");

interactionLevels.forEach(level => {

  const option = document.createElement("option");

  option.value = level;

  option.textContent = level;

  interactionSelect.appendChild(option);

});

document.body.appendChild(interactionSelect);

interactionSelect.addEventListener("change", (e) => {

  selectedLevel = e.target.value;

  console.log(`Selected interaction level: ${selectedLevel}`);

  // 可以根据选择的频率调整 SAM 的互动策略

});

**4. 集成问卷反馈**

在 WP3 中，你需要将 **前测问卷** 的数据集成到系统中，以影响 SAM 的互动策略。你可以设计一个简单的系统，在学生观看视频前进行问卷调查，之后根据反馈调整交互。

**问卷集成**

你可以将问卷直接嵌入到页面中，或使用 **Qualtrics** 集成（根据团队的决定）。

**代码示例：**

javascript

复制编辑

// 假设前测问卷数据通过 API 获取，并影响 SAM 互动策略

async function fetchPreTestData() {

  const response = await fetch('/api/preTest');

  const data = await response.json();

  return data;  // 返回问卷数据，影响后续交互

}

function integratePreTestData(data) {

  // 使用前测数据影响交互频率、内容等

  console.log('Pre-test data:', data);

}

**5. 测试与优化**

* 在前端实现基本交互界面后，你需要进行 **集成测试**，确保 **OpenAI API**、行为追踪、问卷反馈等各个模块能够顺利配合工作。
* 如果你希望系统更智能化，可以考虑 **调整 OpenAI API 的模型**，例如使用不同的 **提示工程** 来优化生成的反馈质量。

**总结：**

虽然 WP2 还未完成，你可以依赖 **OpenAI API** 来实现 **WP3** 的核心功能，例如难度评分、行为追踪和互动。通过编写前端代码和集成 OpenAI API，你可以实现基于学生行为和课程内容的互动机制，并为后续切换到离线模型做好准备。在整个过程中，与 WP2 团队保持密切沟通，确保在模型切换时能够顺利过渡。