사전 제작 영상 기반 인터랙티브 홈페이지 개발 방안 (간단 버전)

사용자가 미리 제작한 아바타 립싱크 영상을 질문에 맞춰 재생하는 방식의 홈페이지 메인 페이지 개발 방안입니다.

1. 준비물

- 질문 목록 및 답변 스크립트**:** 예상되는 사용자 질문과 그에 대한 답변 스크립트를 미리 준비합니다.
- 아바타 립싱크 영상**:** 각 답변 스크립트별로 아바타가 말하는 립싱크 영상을 미리 제작합니다. (예: 회사소개.mp4, 주요서비스안내.mp4, 문의방법.mp4 등)
  - 영상의 파일명이나 ID를 질문과 매칭할 수 있도록 체계적으로 관리합니다.
- 기본 아바타 이미지**/**영상 **(**선택 사항**):** 질문 입력 대기 상태나, 매칭되는 영상이 없을 때 보여줄 아바타의 기본 상태 이미지 또는 짧은 반복 영상을 준비할 수 있습니다.

2. 개발 핵심 요소

- 사용자 입력 인터페이스**:**
  - 텍스트 입력창**:** 사용자가 직접 질문을 입력할 수 있는 공간.
  - 질문 버튼 **(**선택 사항**):** 자주 묻는 질문(FAQ)들을 버튼 형태로 미리 제공하여 사용자가 클릭만으로 영상을 볼 수 있게 합니다.
  - **(**옵션**)** 간단한 음성 입력**:** 웹 브라우저의 Web Speech API를 활용하여 음성을 텍스트로 변환하고, 이 텍스트를 기반으로 영상을 매칭할 수 있습니다. 이는 "아바타가 직접 말을 듣는" 느낌을 줄 수 있습니다.
- 질문**-**영상 매칭 로직**:**
  - 사용자의 입력(텍스트 또는 버튼 선택)을 분석하여 어떤 영상을 재생할지 결정하는 로직입니다.
  - 가장 간단하게는 특정 키워드가 포함되어 있으면 해당 영상을 재생하는 방식을 사용할 수 있습니다.
  - 예시 (JavaScript 객체 형태):

```
const videoMap = {
  "회사 소개": "videos/introduction.mp4",
  "어떤 서비스": "videos/services.mp4",
  "우리 회사": "videos/introduction.mp4", // 동의어 처리
  "연락처": "videos/contact.mp4",
  "문의": "videos/contact.mp4"
};
```

- 영상 재생기**:**
  - 선택된 영상을 화면에 보여주고 재생하는 컴포넌트입니다. HTML5의 <video> 태그를 주로 사용합니다.

- 기본 **UI/UX** 디자인**:**
  - 아바타 영상이 자연스럽게 보일 수 있는 레이아웃.
  - 사용자가 쉽게 질문하고 답변을 확인할 수 있는 직관적인 디자인.

3. 개발 단계 **(**프론트엔드 중심**)**

1. **HTML** 구조 설계**:**
   - 아바타 영상이 표시될 영역 (<video> 태그).
   - 사용자 질문 입력창 (<input type="text">).
   - 질문 전송 버튼 (<button>).
   - (선택 사항) FAQ 버튼들.
   - (선택 사항) 음성 입력 버튼.

```
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>AI 아바타 회사 소개</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <div class="avatar-container">
    <video id="avatarVideo" width="640" height="480"
src="videos/default_idle.mp4" autoplay loop muted></video>
    </div>
  <div class="interaction-container">
    <input type="text" id="questionInput" placeholder="아바타에게
질문해주세요...">
    <button id="askButton">질문하기</button>
    <div class="faq-buttons">
      <button class="faq-btn" data-question="회사 소개">회사 소개</button>
      <button class="faq-btn" data-question="주요 서비스">주요
서비스</button>
      </div>
    </div>

  <script src="script.js"></script>
</body>
</html>
```

2. **CSS** 스타일링**:**
   - 아바타 영상, 입력창, 버튼 등의 디자인을 보기 좋게 꾸밉니다.
   - 반응형 디자인을 적용하여 다양한 화면 크기에서 잘 보이도록 합니다.
3. **JavaScript** 로직 구현 **(script.js):**

- 질문-영상 매핑 데이터 정의:

```
const videoMap = {
  "회사소개": "videos/company_intro.mp4", // 사용자가 준비한 영상 파일 경로
  "서비스": "videos/services_overview.mp4",
  "사업영역": "videos/services_overview.mp4",
  "연락처": "videos/contact_info.mp4",
  "기본": "videos/default_greeting.mp4", // 매칭되는 질문이 없을 때
  // 추가적인 질문과 영상 경로들을 매핑합니다.
};
const defaultVideo = "videos/default_idle.mp4"; // 정말 아무것도 없을 때, 또는 대기 영상
const iDontUnderstandVideo = "videos/dont_understand.mp4"; // 이해 못했을 때 영상
```

- **DOM** 요소 가져오기:

```
const avatarVideo = document.getElementById('avatarVideo');
const questionInput = document.getElementById('questionInput');
const askButton = document.getElementById('askButton');
const faqButtons = document.querySelectorAll('.faq-btn');
// const voiceButton = document.getElementById('voiceButton'); // 음성 입력 시
```

- 이벤트 리스너 설정:
  - 질문하기 버튼 클릭 시: 입력된 텍스트를 가져와 findAndPlayVideo 함수 호출.
  - **FAQ** 버튼 클릭 시: 버튼의 data-question 값을 가져와 findAndPlayVideo 함수 호출.
  - **(**옵션**)** 음성 입력 버튼 클릭 시: Web Speech API를 사용하여 음성 인식 시작, 인식된 텍스트로 findAndPlayVideo 함수 호출.
  - 엔터키 입력 시: 텍스트 입력창에서 엔터키를 누르면 질문하기 버튼 클릭과 동일하게 동작.

```
askButton.addEventListener('click', () => {
  const question = questionInput.value.trim();
  if (question) {
    findAndPlayVideo(question);
    questionInput.value = ''; // 입력창 비우기
  }
});

questionInput.addEventListener('keypress', (event) => {
```

```
    if (event.key === 'Enter') {
        askButton.click();
    }
});

faqButtons.forEach(button => {
    button.addEventListener('click', () => {
        const question = button.dataset.question;
        findAndPlayVideo(question);
    });
});
```

- 질문 분석 및 영상 재생 함수 **(findAndPlayVideo):**

```
function findAndPlayVideo(questionText) {
    let videoSrc = null;
    const lowerQuestion = questionText.toLowerCase(); // 소문자로 변환하여 비교

    // 1. 정확히 일치하는 키워드 찾기 (FAQ 버튼 등)
    if (videoMap[questionText]) {
        videoSrc = videoMap[questionText];
    } else {
    // 2. 키워드 포함 여부로 찾기
        for (const keyword in videoMap) {
            if (lowerQuestion.includes(keyword.toLowerCase())) {
                videoSrc = videoMap[keyword];
                break;
            }
        }
    }

    if (videoSrc) {
        playVideo(videoSrc);
    } else {
        playVideo(iDontUnderstandVideo); // 매칭되는 영상이 없을 때 "이해 못했다" 영상 재생
    }
}
```

- 영상 재생 함수 **(playVideo):**

```
function playVideo(src) {
    avatarVideo.src = src;
    avatarVideo.muted = false; // 실제 답변 영상 재생 시에는 소리 켜기
    avatarVideo.loop = false;  // 답변 영상은 반복 안 함
    avatarVideo.play();

    // 영상 재생이 끝나면 다시 대기 영상으로 돌아가거나, 사용자의 다음 입력을 기다리는 상태로 전환
    avatarVideo.onended = () => {
        avatarVideo.src = defaultVideo; // 또는 videoMap["기본"]
        avatarVideo.muted = true;
        avatarVideo.loop = true;
        avatarVideo.play();
    };
}
```

- **(옵션) Web Speech API**를 이용한 음성 인식**:**

```
// const recognition = new (window.SpeechRecognition || window.webkitSpeechRecognition)();
// recognition.lang = 'ko-KR';
// recognition.interimResults = false;
// recognition.maxAlternatives = 1;

// if (voiceButton) {
//     voiceButton.addEventListener('click', () => {
//         recognition.start();
//     });
// }

// recognition.onresult = (event) => {
//     const speechResult = event.results[0][0].transcript;
//     questionInput.value = speechResult; // 인식된 내용을 입력창에 표시
//     findAndPlayVideo(speechResult);
// };

// recognition.onspeechend = () => {
//     recognition.stop();
// };
```

```
// recognition.onerror = (event) => {
//     console.error('Speech recognition error:', event.error);
//     playVideo(iDontUnderstandVideo); // 음성 인식 실패 시
// };
```

(Web Speech API는 브라우저 호환성 문제가 있을 수 있으며, HTTPS 환경에서 주로 작동합니다.)

4. 테스트 및 디버깅**:**
    - 다양한 질문을 입력하여 의도한 대로 영상이 재생되는지 확인합니다.
    - 오타나 예상치 못한 입력에 대한 예외 처리를 확인합니다. (예: iDontUnderstandVideo 재생)
    - 여러 브라우저에서 테스트합니다.

4. 회사 소개 자료 활용

올려주실 회사 소개 자료는 어떤 질문들을 예상하고**,** 그에 대한 답변으로 어떤 내용의 영상을 만들어야 할지 결정하는 데 핵심적인 역할을 합니다. 즉, videoMap의 키(질문 키워드)와 값(영상 파일 경로)을 구성하는 기초 자료가 됩니다.

예를 들어, 회사 소개 자료에 "우리의 주요 사업 영역은 A, B, C 입니다."라는 내용이 있다면,

- 예상 질문: "주요 사업이 뭐예요?", "어떤 서비스를 하나요?"
- 영상 제목/키워드: 주요사업, 서비스
- 영상 내용: 아바타가 "저희 회사의 주요 사업 영역은 A, B, C입니다."라고 말하는 립싱크 영상 (services_overview.mp4)

이처럼 회사 소개 자료의 내용을 바탕으로 Q&A 세트를 만들고, 각 A에 해당하는 영상을 제작하시면 됩니다.

장점

- 구현 용이성**:** 실시간 AI, 복잡한 립싱크 동기화 기술 없이 간단한 HTML, CSS, JavaScript만으로 구현 가능합니다.
- 빠른 개발 속도**:** 영상만 준비된다면 프론트엔드 개발은 상대적으로 빠르게 진행될 수 있습니다.
- 예측 가능한 동작**:** 모든 답변이 미리 정의되어 있어, AI의 예측 불가능한 답변에 대한 걱정이 없습니다.

고려할 점

- 제한된 유연성**:** 미리 정의된 질문 외에는 답변하기 어렵습니다. 다양한 질문에 모두 대응하려면 많은 수의 영상을 제작해야 합니다.
- 영상 제작 부담**:** 질문 하나하나에 맞춰 립싱크 영상을 제작하는 것이 주된 작업이

됩니다.
- 자연스러움**:** 사용자의 미묘한 질문 변화나 복잡한 질문에는 부자연스럽게 느껴질 수 있습니다.

이 방법은 말씀하신 것처럼 "가장 쉬운 방법론"에 가까우며, 회사 소개 내용이 많지 않고 사업 영역이 간단하다면 효과적으로 활용할 수 있습니다. 핵심은 사용자가 어떤 질문을 할지 잘 예측하여 관련 영상을 충분히 준비하는 것입니다.