OVA: LEXIMIND

LLM (LARGE LANGUAGE MODELS)

JUAN DAVID CONTRERAS LEYVA

ANGEL DAVID CASTILLO MARÍN

FRANCISCO DE LOS REYES BURGOS PACHECO

ALEXANDER ENRIQUE TOSCANO RICARDO

UNIVERDIDAD DE CORDOBA

FACULTAD DE EDUCACION Y CIENCIAS HUMANAS

LIC. INFORMATICA

MONTERÍA

Introducción

El avance acelerado de la inteligencia artificial ha traído consigo herramientas cada vez más sofisticadas, entre las cuales destacan los **Modelos de Lenguaje de Gran Escala (LLMs)**. Estas tecnologías, capaces de procesar y generar lenguaje natural de forma coherente, están transformando múltiples sectores, incluida la educación. Los LLMs como ChatGPT, Gemini o Claude ofrecen nuevas oportunidades para la creación de contenido, la personalización del aprendizaje y la asistencia académica, generando un impacto significativo en la forma en que docentes y estudiantes interactúan con la información.

Frente a este panorama, surge la necesidad de capacitar a la comunidad educativa en el uso crítico y pedagógico de estas herramientas. En respuesta a ello, desarrollamos *Leximind*, un **Objeto Virtual de Aprendizaje (OVA)** diseñado para introducir, contextualizar y explorar el uso de los LLMs en la educación. Este recurso busca no solo informar sobre el funcionamiento de los modelos, sino también fomentar su implementación responsable en el aula, reflexionando sobre sus beneficios, retos y consideraciones éticas.

El OVA fue desarrollado utilizando tecnologías modernas como **Nuxt 3**, **Vuetify** y **Pinia**, permitiendo una estructura modular, interactiva y escalable. Está compuesto por varios módulos temáticos que incluyen contenidos teóricos, casos de uso, actividades prácticas y evaluaciones interactivas. De esta manera, *Leximind* ofrece una experiencia de aprendizaje integral y actualizada, pensada para fortalecer las competencias digitales de los futuros docentes en el contexto de la inteligencia artificial aplicada a la educación.

Módulos del OVA

Basándonos en el diseño instruccional del proyecto, el OVA se divide en 6 módulos que se abordan progresivamente en la interfaz:

1. **Introducción a los LLMs**
2. **LLMs en el contexto educativo**
3. **Implementación práctica**
4. **Desafíos y consideraciones éticas**
5. **Casos de estudio y buenas prácticas**
6. **El futuro de los LLMs en educación**

Módulo 1: Introducción a los LLMs

Objetivo: Comprender qué son los LLMs y su relevancia en el contexto actual.
Contenidos:

1. **¿Qué son los LLMs?**
    - Definición y ejemplos (GPT, Gemini, Claude, etc.).
    - Funcionamiento básico: entrenamiento, datos y arquitectura neuronal.
2. **Breve historia de los LLMs**
    - Evolución desde modelos simples hasta la inteligencia artificial generativa.
3. **Características clave de los LLMs**
    - Generación de texto, traducción, resumen, respuesta a preguntas.

Módulo 2: Los LLMs en el Contexto Educativo

Objetivo: Analizar el impacto de los LLMs en la enseñanza y el aprendizaje.
Contenidos:

1. **Aplicaciones en la educación**
    - Creación de contenido educativo (guías, ejercicios, explicaciones).
    - Tutorías personalizadas y soporte 24/7.
    - Evaluación automatizada de tareas y retroalimentación.
2. **Beneficios para docentes y estudiantes**
    - Reducción de carga administrativa para docentes.
    - Acceso a recursos personalizados para estudiantes.
3. **Ejemplos de herramientas basadas en LLMs**
    - ChatGPT para dudas académicas, Copilot para codificación, herramientas de resumen de textos.

Módulo 3: Implementación Práctica de LLMs en el Aula

Objetivo: Aprender a integrar LLMs en actividades educativas.
Contenidos:

1. **Diseño de actividades con LLMs**
    - Creación de ejercicios interactivos.

    - Uso de LLMs para debates simulados o role-playing histórico.
2. **Personalización del aprendizaje**
    - Adaptación de contenidos a niveles de habilidad (ejercicios para principiantes vs. avanzados).
3. **Herramientas y plataformas recomendadas**
    - Uso de ChatGPT, Bing Chat, Gemini, y plataformas educativas con IA integrada (como Khan Academy o Duolingo).

Módulo 4: Desafíos y Consideraciones Éticas

Objetivo: Reflexionar sobre los límites y responsabilidades del uso de LLMs.
Contenidos:

1. **Riesgos y limitaciones**
    - Sesgos en los datos de entrenamiento.
    - Posible plagio académico y dependencia excesiva.
2. **Ética en el uso de LLMs**
    - Transparencia en la autoría de contenidos generados por IA.
    - Protección de datos y privacidad estudiantil.
3. **Cómo mitigar los riesgos**
    - Estrategias para verificar información generada por LLMs.
    - Fomento del pensamiento crítico en estudiantes.

Módulo 5: Casos de Estudio y Buenas Prácticas

Objetivo: Analizar experiencias reales de uso de LLMs en educación.
Contenidos:

1. **Casos exitosos**
    - Escuelas que integran IA para tutorías.
    - Universidades usando LLMs en investigación.
2. **Lecciones aprendidas**
    - Equilibrio entre tecnología y enseñanza humana.
    - Importancia de la capacitación docente.

Módulo 6: Futuro de los LLMs en la Educación

Objetivo: Explorar tendencias y desarrollos futuros.
Contenidos:

1. **Tendencias tecnológicas**
    - LLMs multimodalidad (texto, imagen, voz).
    - Integración con realidad virtual/aumentada.
2. **Escenarios futuros**
    - Educación hiperpersonalizada.
    - Desafíos regulatorios y normativos.

Recursos Adicionales

- Enlaces a artículos, guías de uso ético de IA, y plataformas educativas con LLMs.
- Vídeos explicativos (ej: TED Talks sobre IA en educación).
- Plantillas para diseñar actividades con LLMs.

Actividades de Evaluación

1. **Cuestionarios interactivos**
    - Preguntas de opción múltiple sobre conceptos clave.
2. **Ejercicios prácticos**
    - Crear una lección usando ChatGPT y reflexionar sobre sus ventajas/desventajas.

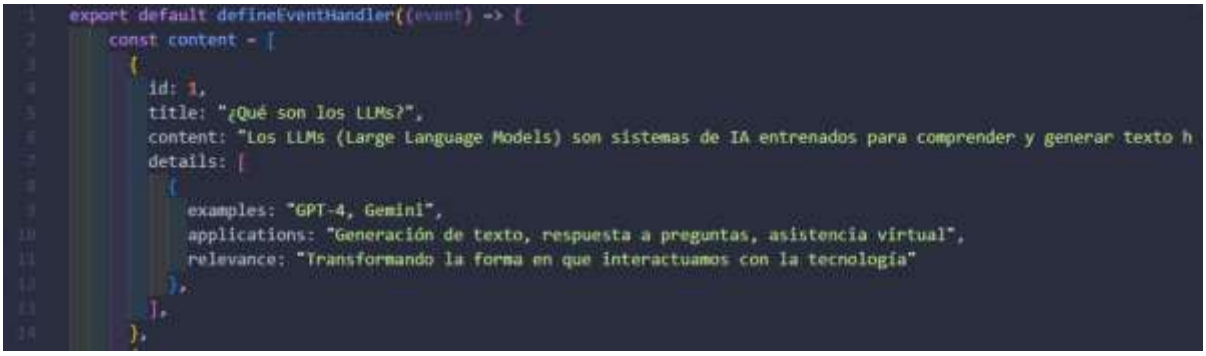
Análisis técnico de funcionalidades

A continuación, se detalla el funcionamiento del sistema en tres niveles clave: la API simulada, la gestión de estado con Pinia y la interfaz correspondiente en **pages/content.vue.**

API Local – server/api/contents.get.ts

Esta API entrega un listado estático de contenidos relacionados con los módulos mencionados. Cada elemento posee:

- **id:** identificador único
- **title:** título del módulo/contenido
- **content:** descripción principal
- **details**: objeto con subapartados o explicaciones complementarias



Esta estructura permite:

- Agilidad en el desarrollo (sin requerir base de datos).
- Pruebas dinámicas durante el diseño del OVA.
- Consistencia en la estructura de los módulos.

Gestión de estado – store/content.ts

El archivo content.ts es un **store de Pinia** que maneja toda la lógica de los contenidos del OVA. Proporciona almacenamiento centralizado y reactivo para:

- El listado de contenidos (contents)
- El contenido seleccionado por el usuario (currentContent)
- Estados de carga (isLoading) y errores (error)

Estructura principal:

```
export const useContentStore = defineStore('content', {
  state: () => ({
    contents: [] as Content[],
    currentContent: null as Content | null,
    isLoading: false,
    error: null as string | null
  }),
```

Funcionalidades clave:

fetchContents() : Esta es la que nos ayuda a realizar la petición a la API y esta carga los datos.

```
actions: {
  /**
   * Obtiene los contenidos desde la API
   */
  async fetchContents() {
    this.isLoading = true
    this.error = null

    try {
      const response = await $fetch('/api/content', { method: 'GET' }) as Content[]
      this.contents = response
    } catch (error) {
      console.error('Error fetching contents:', error)
      this.error = 'No fue posible cargar los contenidos'
    } finally {
      this.isLoading = false
    }
  },
```

setCurrentContent(content): Establece un contenido como el actual.

```
  setCurrentContent(content: Content) {
    this.currentContent = { ...content }
  },
```

setCurrentContentById(id): Busca un contenido por ID y lo asigna.

```
setCurrentContentById(id: number) {
  const content = this.getContentById(id)
  if (content) {
    this.currentContent = { ...content }
  }
},
```

updateContent(updatedContent) : Modifica un contenido existente.

```
updateContent(updatedContent: Content) {
  const index = this.contents.findIndex(c => c.id === updatedContent.id)
  if (index !== -1) {
    // Reemplazar el contenido completo para mantener la reactividad
    this.contents[index] = { ...updatedContent }

    // Si el contenido actual es el que se actualizó, también actualizamos currentContent
    if (this.currentContent && this.currentContent.id === updatedContent.id) {
      this.currentContent = { ...updatedContent }
    }
  }
},
```

clearCurrentContent(): Elimina la selección actual.

```
clearCurrentContent() {
  this.currentContent = null
}
```

Esto por supuesto tienes unas ventajas que son las siguientes:

- Separación entre lógica y presentación.
- Soporte para edición en tiempo real.
- Manejo robusto de errores y estados.

Vista de contenidos - pages/content.vue:

Esta es la página principal para visualizar, seleccionar y editar contenidos. Se estructura en dos columnas:

- Izquierda: lista de contenidos (ContentCard)
- Derecha: vista de edición (EditContent), visible solo si hay un contenido seleccionado.

Comportamiento:

- Al montarse, llama a fetchContents() para obtener datos.
- Si no hay contenidos, muestra una alerta.
- Permite al usuario seleccionar un módulo para editarlo.

Código clave:

```
<script setup>
const storeContent = useContentStore()

const contents = computed(() => storeContent.contents)
const currentContent = computed(() => storeContent.currentContent)

const handleSelectContent = (content) => {
  storeContent.setCurrentContent(content)
}

onMounted(() => {
  storeContent.fetchContents()
})
</script>
```

Flujo del usuario:

1. El usuario entra a /content
2. Se cargan automáticamente los módulos disponibles.
3. Al seleccionar uno, se abre el panel de edición.
4. Cualquier modificación puede guardarse en memoria.

Componente ContentCard.vue – Visualización y selección de contenidos

Durante el desarrollo del módulo de contenidos del OVA Leximind, uno de los elementos clave fue la creación de un componente reutilizable que permitiera mostrar cada contenido de manera clara, estética y funcional. Para ello, implementamos ContentCard.vue, un componente personalizado ubicado en el

directorio components/, el cual cumple la función de representar visualmente cada módulo del OVA como una tarjeta.

Este componente recibe como **propiedad (prop)** un objeto content, que representa uno de los módulos del OVA. Dentro de la tarjeta, mostramos:

- El **título** (content.title),
- La **descripción principal** (content.content),
- Una iteración sobre los **detalles complementarios** (content.details), que pueden ser conceptos, aplicaciones, ejemplos o definiciones clave.

Además, incluimos un botón con la etiqueta **"Editar Contenido"**, que al ser presionado emite un evento al componente padre (pages/content.vue), indicando cuál contenido fue seleccionado para su edición.

Código del componente ContentCard.vue:

```vue
<template>
  <v-card class="mb-4" elevation="2">
    <v-card-title>{{ content.title }}</v-card-title>
    <v-card-text>
      <p>{{ content.content }}</p>
      <div v-for="(detail, index) in content.details" :key="index" class="mb-2">
        <v-list dense>
          <v-list-item
            v-for="(value, key) in detail"
            :key="key"
          >
            <v-list-item-content>
              <strong>{{ formatKey(key) }}:</strong> {{ value }}
            </v-list-item-content>
          </v-list-item>
        </v-list>
      </div>
      <v-btn color="primary" @click="selectContent">Editar Contenido</v-btn>
    </v-card-text>
  </v-card>
</template>
```

Lógica interna del componente:

```vue
<script setup>
const props = defineProps({
  content: { type: Object, required: true },
})

const emit = defineEmits(['select-content'])

const selectContent = () => {
  emit('select-content', props.content)
}

</script>
```

Explicación del funcionamiento:

1. Recepción del contenido: El componente espera recibir un objeto con la información del contenido a mostrar.
2. Renderizado dinámico: Usamos un v-for doble: el primero recorre los detalles (details) y el segundo recorre las propiedades dentro de cada objeto detail.
3. Interacción: Al hacer clic en el botón “Editar Contenido”, se emite el evento select-content, pasando el objeto content al componente padre (content.vue), donde este evento se maneja para activar el panel de edición.

Conexión con pages/content.vue:

```
<ContentCard
  v-for="content in contents"
  :key="content.id"
  :content="content"
  @select-content="handleSelectContent"
/>
```

El componente padre escucha el evento y ejecuta la función:

```
const handleSelectContent = (content) => {
  storeContent.setCurrentContent(content)
}
```

Vista del OVA:

Conclusiones

El desarrollo del Objeto Virtual de Aprendizaje *Leximind* ha permitido sentar las bases para la construcción de un recurso educativo centrado en la comprensión y aplicación crítica de los **Modelos de Lenguaje de Gran Escala (LLMs)** en contextos pedagógicos. Aunque el proyecto no está terminado, los módulos implementados hasta ahora demuestran un enfoque claro y estructurado tanto desde el punto de vista técnico como didáctico.

Uno de los principales avances ha sido la implementación de una arquitectura modular basada en tecnologías modernas como **Nuxt 3**, **Vuetify** y **Pinia**, que ha permitido construir una aplicación escalable, dinámica y orientada al aprendizaje activo. La estructura de contenidos, la API simulada y la gestión del estado ya funcionan correctamente, sentando una base sólida para la incorporación de nuevas secciones como cuestionarios interactivos, actividades prácticas, y casos de uso en entornos reales.

A pesar de que el OVA no está terminado, su diseño permite una fácil expansión e integración de funcionalidades futuras. Se proyecta que, una vez finalizado, *Leximind* se convertirá en una herramienta útil para la formación de docentes y estudiantes en el uso responsable y efectivo de la inteligencia artificial en el ámbito educativo. Por ahora, el avance alcanzado refleja un trabajo coherente con los objetivos planteados y un compromiso con el aprendizaje significativo en la era digital.