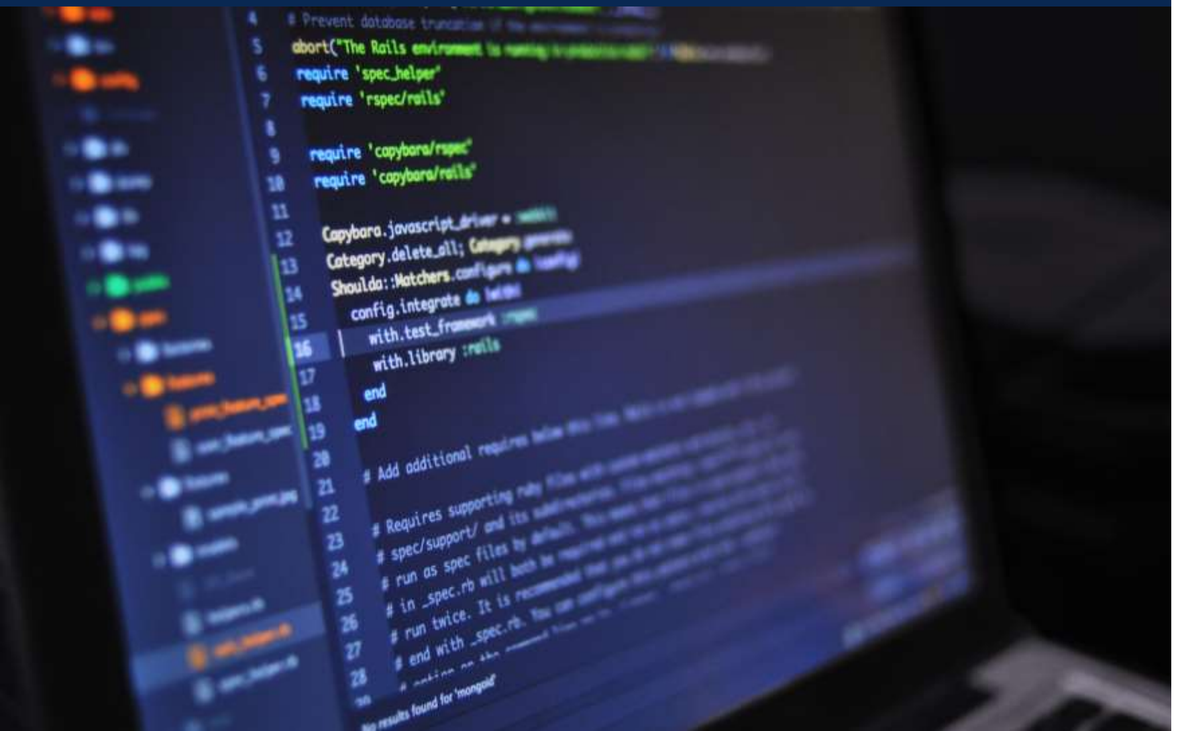
PROYECTO FINAL

SIMULADOR DE EXAMEN

SERGIO CARVAJAL
FABIAN BUELVAS
STEEVEN HARVEY

SIMULADOR DE EXAMEN



Un simulador de exámenes es una herramienta tecnológica diseñada para replicar las condiciones de un examen real, proporcionando a los usuarios una experiencia de práctica cercana a la evaluación oficial. Estos simuladores son ampliamente utilizados en contextos académicos, profesionales y de certificación, permitiendo a los estudiantes y candidatos prepararse de manera efectiva al enfrentar preguntas y formatos similares a los del examen final. El propósito principal de un simulador de exámenes es ayudar a los usuarios a familiarizarse con el tipo de preguntas, la estructura del examen, los límites de tiempo y las estrategias necesarias para maximizar su desempeño. Además, suelen incluir funciones como retroalimentación inmediata, análisis de resultados y recomendaciones personalizadas para reforzar las áreas de mejora.

QUESTIONCARD.VUE

QuestionCard.vue

components > QuestionCard.vue > {} script setup > handleAnswer

```vue
<template>
  <v-card class="mb-4 pa-4">
    <v-card-title class="text-h6">
      Pregunta {{ questionNumber }}
    </v-card-title>
    <v-card-text>
      <div class="question-text mb-4">{{ question.text }}</div>
      <v-radio-group v-model="selectedAnswer" @change="handleAnswer">
        <v-radio
          v-for="(option, index) in question.options"
          :key="index"
          :label="option"
          :value="index"
          color="primary"
        ></v-radio>
      </v-radio-group>
    </v-card-text>
  </v-card>
</template>

<script setup>
import { ref } from 'vue'

const props = defineProps({
  question: {
    type: Object,
    required: true
  },
  questionNumber: {
    type: Number,
    required: true
  }
})

const emit = defineEmits(['answer'])

const selectedAnswer = ref(null)

const handleAnswer = () => {
  emit('answer', {
    questionId: props.question.id,
    selectedAnswer: selectedAnswer.value
  })
}
</script>
```

QUESTIONCARD.VUE

El código proporciona un componente reutilizable para mostrar preguntas de un examen interactivo con opciones de respuesta, que es especialmente útil en aplicaciones como simuladores de exámenes, cuestionarios educativos, o encuestas.
Específicamente:

1.Presentar preguntas dinámicamente:
- •Cada pregunta se configura mediante las propiedades question y questionNumber, lo que permite que el componente sea reutilizado para diferentes preguntas.
- •Incluye un sistema de opciones múltiples (radio buttons) que asegura que solo una opción sea seleccionada por el usuario.

2.Recolectar respuestas del usuario:
- •El evento answer permite enviar los datos de la respuesta seleccionada (ID de la pregunta y la opción elegida) al componente principal que orquesta la lógica del cuestionario.

3.Interactividad:
- •Facilita la interacción del usuario al proporcionar una interfaz limpia, ordenada y fácil de usar mediante Vuetify, un framework de diseño material.

Importancia del Código

1.Reutilización y Escalabilidad:
1. Este componente es reutilizable, lo que significa que puede integrarse en un sistema con múltiples preguntas sin necesidad de escribir código repetitivo.
2. En sistemas de exámenes con decenas o cientos de preguntas, la reutilización mejora significativamente la productividad y el mantenimiento del código.

2.Estandarización:
1. Garantiza que todas las preguntas se muestren con un diseño consistente y profesional, mejorando la experiencia del usuario.

3.Centralización del Manejo de Respuestas:
1. Al emitir un evento con los datos de la respuesta seleccionada, permite que la lógica de evaluación o registro de respuestas sea manejada en un nivel superior, manteniendo el componente limpio y enfocado en su propósito principal: la presentación de preguntas.

4.Adaptabilidad:
1. Es fácilmente modificable para integrar nuevas funcionalidades como:
    1. Validación de respuestas.
    2. Indicaciones correctas/incorrectas.
    3. Contadores de tiempo por pregunta.

EXAM.VUE

Este código representa un simulador interactivo de examen de matemáticas diseñado con Vue 3 y Vuetify, que incluye las siguientes características:

1. Presentación del examen:
   - Muestra una serie de preguntas de opción múltiple extraídas de un banco predefinido (questionBank).
   - Cada pregunta se presenta en un componente reutilizable (QuestionCard).
2. Gestión de tiempo:
   - El temporizador cuenta 30 minutos (timeRemaining) para completar el examen.
   - Si el tiempo se agota, el examen se finaliza automáticamente.
3. Progreso visual:
   - Un indicador de progreso (v-progress-linear) muestra cuántas preguntas han sido respondidas en relación con el total.
4. Evaluación y retroalimentación:
   - Al finalizar, muestra la puntuación total obtenida (score) junto con un desglose de las respuestas correctas e incorrectas, indicando con íconos el estado de cada pregunta.
5. Reinicio del examen:
   - Permite al usuario reiniciar el examen desde el principio con el botón "Intentar Nuevo Examen".

IMPORTANCIA

- **Experiencia Realista de Exámenes**:
Simula las condiciones de un examen real con límite de tiempo y seguimiento del progreso, lo cual es útil para estudiantes que necesitan prepararse para evaluaciones bajo presión.

- **Interactividad y Dinamismo**:
Las preguntas y respuestas se gestionan de manera interactiva, lo que fomenta el compromiso del usuario.

- **Retroalimentación Inmediata**:
Al finalizar, el usuario recibe un desglose detallado de su desempeño, identificando áreas donde necesita mejorar.

- **Reutilización y Escalabilidad**:
Es fácilmente ampliable a exámenes más complejos. Solo es necesario agregar más preguntas al questionBank o integrarlo con una base de datos externa.

- **Adaptación a Distintos Usos**:
Puede usarse no solo para matemáticas, sino para cualquier otra materia, modificando el banco de preguntas y el contenido.

EXAM.JS

JS exam.js 9+

stores > JS exam.js > ...

```
<script setup>
import { defineStore } from 'pinia'

export const useExamStore = defineStore('exam', {
  state: () => ({
    currentExam: null,
    examHistory: [],
    userProgress: {
      totalExams: 0,
      averageScore: 0
    }
  }),

  actions: {
    startNewExam() {
      this.currentExam = {
        startTime: new Date(),
        answers: {},
        completed: false
      }
    },

    submitAnswer(questionId, answer) {
      if (this.currentExam) {
        this.currentExam.answers[questionId] = answer
      }
    },

    finishExam(score) {
      if (this.currentExam) {
        this.currentExam.completed = true
        this.currentExam.endTime = new Date()
        this.currentExam.score = score

        this.examHistory.push({ ...this.currentExam })
        this.updateProgress()
      }
    },

    updateProgress() {
      this.userProgress.totalExams = this.examHistory.length
      this.userProgress.averageScore = this.examHistory.reduce((acc, exam) =>
        acc + exam.score, 0) / this.examHistory.length
    }
  }
})
</script>
```

EXAM.JS

Este código define un store en Pinia para gestionar el estado global relacionado con un simulador de exámenes. Se encarga de almacenar y administrar la información del examen actual, el historial de exámenes completados y el progreso general del usuario.

- Estado Inicial (state):
- currentExam: Contiene los datos del examen que el usuario está realizando actualmente.
- examHistory: Almacena un historial de todos los exámenes completados por el usuario.
- userProgress: Rastrea estadísticas del usuario, como el número total de exámenes realizados y el puntaje promedio.
- Acciones (actions):
- startNewExam():
  - Inicia un nuevo examen, estableciendo su hora de inicio y configurando un espacio para almacenar respuestas.
- submitAnswer(questionId, answer):
  - Guarda las respuestas del usuario para el examen actual.
- updateProgress():
  - Calcula y actualiza las estadísticas del usuario, como el número total de exámenes y el puntaje promedio.
- finishExam(score):
  - Marca el examen actual como completado, guarda la puntuación obtenida, registra la hora de finalización y lo añade al historial.

I M P O R T A N C I A

- Gestión Centralizada del Estado:

Mantiene todos los datos relevantes de los exámenes en un único lugar, lo que facilita el acceso y la modificación desde cualquier parte de la aplicación.

- Seguimiento del Usuario:

Permite almacenar el historial de exámenes y calcular métricas clave como el promedio de puntuación y el número total de exámenes realizados.

- Estructura Reutilizable:

Al encapsular la lógica relacionada con los exámenes, se puede integrar fácilmente en diferentes partes de la aplicación sin duplicar código.

- Consistencia y Escalabilidad:

Asegura que los datos del examen actual y el progreso del usuario se mantengan consistentes, incluso si la aplicación crece en complejidad.