

CHATBOT CON EXPRESIONES REGULARES PARA VALIDACIÓN DEL VERBO TO BE

Daniel David Henriquez Julio

Juan David Bolaños Lopez

Víctor Daniel Marrugo Aguilar

Jesús David Caraballo Nieto

Wilmer Andrés Iriarte Camargo

Gabriel Jose Buelvas Morales

FUNDACIÓN TECNOLÓGICO COMFENALCO DE CARTAGENA

Profesor: Ing. Carlos García Castro

Ing.Sistemas

Fecha: 11 De Septiembre De 2025

Cartagena/Bolivar/Colombia

1. INTRODUCCIÓN

Este proyecto implementa un chatbot interactivo que utiliza expresiones regulares para validar frases en inglés que contienen el verbo TO BE en sus formas de tiempo presente y pasado. Las expresiones regulares son una herramienta fundamental en ciencias de la computación para el reconocimiento de patrones y procesamiento de texto, permitiendo definir de manera precisa y eficiente las reglas gramaticales que debe seguir una frase válida.

La importancia de las expresiones regulares en este contexto radica en su capacidad para:

- Definir patrones complejos de manera concisa
- Validar estructuras gramaticales específicas
- Procesar texto de forma eficiente
- Implementar autómatas finitos para reconocimiento de lenguajes

Este sistema educativo permite a los usuarios practicar la construcción correcta de oraciones en inglés, proporcionando retroalimentación inmediata sobre la validez gramatical de sus frases.

2. MARCO TEÓRICO

2.1 Expresiones Regulares

Definición: Una expresión regular es una secuencia de caracteres que define un patrón de búsqueda. Formalmente, es una cadena que describe un conjunto de cadenas según ciertas reglas sintácticas.

Operadores Básicos:

- **Concatenación:** `ab` - coincide con 'a' seguido de 'b'
- **Alternación:** `a|b` - coincide con 'a' o 'b'
- **Repetición:** `a*` - cero o más repeticiones de 'a'
- **Repetición positiva:** `a+` - una o más repeticiones de 'a'
- **Opcional:** `a?` - cero o una repetición de 'a'
- **Agrupación:** `(ab)+` - una o más repeticiones del grupo 'ab'
- **Clases de caracteres:** `[a-z]` - cualquier letra minúscula
- **Anclas:** `^` inicio de línea, `$` final de línea

Ejemplos aplicados al proyecto:

<code>/^i\s+am\s+.\s+/i</code>	<code># "I am happy"</code>
<code>/^(he she it)\s+is\s+.\s+/i</code>	<code># "He is tall"</code>
<code>/^[A-Z][a-z]+\s+was\s+.\s+/</code>	<code># "Maria was here"</code>

2.2 Chatbots

Definición: Un chatbot es un programa de software diseñado para simular conversaciones con usuarios humanos, especialmente a través de internet.

Aplicación en el Proyecto: El chatbot implementado actúa como un tutor virtual que:

- Mantiene una conversación natural con el usuario
- Recibe frases en inglés para validar

- Proporciona retroalimentación educativa inmediata
- Guía al usuario en el proceso de aprendizaje

2.3 Gramática de las Frases con TO BE

Estructura: Sujeto + Verbo TO BE + Predicado

Tipos de Sujeto:

1. **Pronombres Personales:** I, You, He, She, It, We, They
2. **Sustantivos Propios:** Maria, Carlos, Cartagena
3. **Sustantivos Comunes con Artículo:** The car, The students
4. **Pronombres Demostrativos:** This book, Those pencils

Formas del Verbo TO BE:

- **Presente:** am, are, is
- **Pasado:** was, were

Tipos de Oraciones:

- **Afirmativas:** "She is happy"
- **Negativas:** "She is not happy" / "She isn't happy"
- **Interrogativas:** "Is she happy?"

3. SUSTENTACIÓN DEL LENGUAJE DE PROGRAMACIÓN SELECCIONADO

TypeScript + React

Justificación de la Selección:

1. TypeScript:

- **Tipado Estático:** Permite detectar errores en tiempo de compilación
- **Mejor IntelliSense:** Autocompletado y documentación mejorada
- **Mantenibilidad:** Código más legible y mantenible
- **Escalabilidad:** Facilita el crecimiento del proyecto

2. React:

- **Componentes Reutilizables:** Arquitectura modular
- **Estado Reactivo:** Manejo eficiente del estado de la aplicación
- **Ecosistema Maduro:** Amplia comunidad y recursos
- **Experiencia de Usuario:** Interfaz moderna e interactiva

3. Ventajas para el Proyecto:

- Separación clara entre lógica de validación y presentación
- Interfaz de chat en tiempo real
- Manejo eficiente del estado de la conversación
- Fácil extensibilidad para futuras funcionalidades

4. IMPLEMENTACIÓN

4.1 Expresiones Regulares Utilizadas

4.1.1 Tiempo Presente - Afirmativo

```
// Pronombre "I"
i: /^i\s+am\s+.+/i

// Pronombres plurales y "You"
youWe: /^(you|we|they)\s+are\s+.+/i

// Pronombres singulares tercera persona
heSheIt: /^(he|she|it)\s+is\s+.+/i

// Sustantivos propios singulares
properNounSingular: /^[A-Z][a-z]+\s+is\s+./i

// Sustantivos comunes con artículo
commonNounSingular: /^the\s+[a-z]+\s+is\s+.+/i
commonNounPlural: /^the\s+[a-z]+\s+are\s+.+/i

// Pronombres demostrativos
demonstrativeThis: /^(this|that)\s+[a-z]+\s+is\s+.+/i
demonstrativeThese: /^(these|those)\s+[a-z]+\s+are\s+.+/i
```

4.1.2 Tiempo Presente - Negativo

```
// Negación con "I"
i: /^i\s+(am\s+not|'m\s+not)\s+.+/i

// Negación con contracciones
youWe: /^(you|we|they)\s+(are\s+not|aren't)\s+.+/i
heSheIt: /^(he|she|it)\s+(is\s+not|isn't)\s+.+/i
```

4.1.3 Tiempo Pasado - Afirmativo

```
// Sujetos que usan "was"
iHeSheIt: /^(i|he|she|it)\s+was\s+.+/i

// Sujetos que usan "were"
youWeThey: /^(you|we|they)\s+were\s+.+/i
```

4.1.4 Tiempo Pasado - Negativo

```
// Negación en pasado
iHeSheIt: /^(i|he|she|it)\s+(was\s+not|wasn't)\s+.+/i
youWeThey: /^(you|we|they)\s+(were\s+not|weren't)\s+.+/i
```

4.1.5 Preguntas

```
// Preguntas en presente  
toBeQuestionPresent: /^(am|are|is)\s+.+\?$/i  
  
// Preguntas en pasado  
toBeQuestionPast: /^(was|were)\s+.+\?$/i
```

4.2 Módulos Principales

4.2.1 Módulo Validador (validator.ts)

- **Función:** Contiene toda la lógica de validación
- **Responsabilidades:**
 - Definir expresiones regulares
 - Validar frases según los patrones
 - Generar mensajes de retroalimentación
 - Proporcionar correcciones sugeridas

4.2.2 Módulo Chatbot (App.tsx)

- **Función:** Interfaz de usuario y manejo de conversación
- **Responsabilidades:**
 - Gestionar el flujo de conversación
 - Manejar el estado de la aplicación
 - Proporcionar interfaz de chat interactiva
 - Integrar la validación con la experiencia de usuario

4.2.3 Módulo de Pruebas (validator.test.ts)

- **Función:** Asegurar la calidad del código
- **Responsabilidades:**
 - Validar funcionamiento de expresiones regulares
 - Probar casos límite
 - Garantizar cobertura de pruebas

4.3 Ejemplos de Ejecución

Ejemplo 1: Validación Exitosa de nombre

This screenshot shows a chat interface with a light gray background. On the left, a white chat bubble from 'BOTSITO' (01:45 a.m.) contains the text: "Hello! I'm Botsito, an English Grammar Bot specializing in the verb TO BE. What's your name?". On the right, a blue chat bubble from 'Daniel' (01:45 a.m.) contains the text: "Daniel". Below this, another white bubble from 'BOTSITO' (01:45 a.m.) says: "Nice to meet you, Daniel! I can help you validate English sentences using the verb TO BE in present and past tense. Would you like to start practicing? (Type 'yes' to begin or 'examples' to see some examples)". A final blue bubble from 'Daniel' (01:45 a.m.) contains the text: "yes". Each bubble has a small circular icon with a 'D' or 'B' and a checkmark.

Ejemplo 2: Validación Exitosa

This screenshot shows a chat interface. On the left, a white chat bubble from 'BOTSITO' (01:45 a.m.) contains the text: "Excellent! Now you can send me English sentences using the verb TO BE (am, are, is, was, were) and I'll validate them for grammatical correctness. Please enter a sentence to validate." On the right, a green chat bubble from 'Daniel' (01:45 a.m.) contains the text: "She is my girlfriend." Below this, a white bubble from 'BOTSITO' (01:45 a.m.) contains the text: "✓ Correct affirmative sentence with TO BE in present tense! Excellent! That's a affirmative sentence in present tense. Please try another sentence or type 'stop' to end the session." The green bubble on the right also contains the text: "VALID - AFFIRMATIVE (PRESENT)" and has a checkmark.

Ejemplo 2: Error Detectado

This screenshot shows a chat interface. On the left, a white chat bubble from 'BOTSITO' (02:49 a.m.) contains the text: "X Error: 'I' always uses 'am', never 'are'. Let me help you improve this sentence: Correct version: 'I am a student' Reminder: Check subject-verb agreement with TO BE verbs. Please try another sentence or type 'stop' to end the session." On the right, a red chat bubble from 'Daniel' (02:49 a.m.) contains the text: "I are student". Below the red bubble, a white bubble from 'BOTSITO' (02:49 a.m.) contains the text: "INVALID - AFFIRMATIVE (PRESENT)" and has a checkmark.

Ejemplo 3: Validación en Pasado

This screenshot shows a chat interface. On the right, a green chat bubble from 'Daniel' (02:51 a.m.) contains the text: "They were in Cartagena yesterday." Below this, a white chat bubble from 'BOTSITO' (02:51 a.m.) contains the text: "✓ Correct affirmative sentence with TO BE in past tense! Excellent! That's a affirmative sentence in past tense. Please try another sentence or type 'stop' to end the session." The green bubble on the right also contains the text: "VALID - AFFIRMATIVE (PAST)" and has a checkmark.

5. CONCLUSIONES

5.1 Aprendizajes sobre Expresiones Regulares

Durante el desarrollo de este proyecto, se adquirieron conocimientos profundos sobre:

1. **Diseño de Patrones:** La importancia de crear expresiones regulares precisas y eficientes que capturen exactamente los casos deseados sin generar falsos positivos.
2. **Optimización:** La necesidad de balancear la complejidad de las expresiones con su rendimiento, especialmente al manejar múltiples patrones simultáneamente.
3. **Mantenibilidad:** La importancia de estructurar las expresiones regulares de manera modular y documentada para facilitar futuras modificaciones.
4. **Casos Límite:** La identificación y manejo de casos especiales como contracciones, sustantivos plurales, y combinaciones complejas de sujetos.

5.2 Enriquecimiento Personal, Laboral y Académico

Personal:

- Desarrollo de habilidades de pensamiento lógico y resolución de problemas
- Mejora en la comprensión de la gramática inglesa
- Satisfacción de crear una herramienta educativa funcional

Laboral:

- Experiencia práctica con tecnologías modernas (TypeScript, React)
- Comprensión profunda de expresiones regulares aplicables en validación de datos
- Desarrollo de software con enfoque en experiencia de usuario
- Práctica en testing y aseguramiento de calidad

Académico:

- Aplicación práctica de conceptos de autómatas y lenguajes formales
- Integración de teoría de compiladores con desarrollo de software
- Comprensión de la relación entre gramáticas formales y expresiones regulares
- Experiencia en documentación técnica y presentación de proyectos

5.3 Impacto Educativo

Este proyecto demuestra cómo los conceptos teóricos de autómatas y lenguajes pueden aplicarse para crear herramientas educativas prácticas. La implementación de un chatbot que enseña gramática inglesa mediante expresiones regulares representa una convergencia exitosa entre:

- Teoría de autómatas finitos
- Procesamiento de lenguaje natural básico
- Desarrollo de software moderno
- Educación interactiva

***Nota:** Este informe documenta completamente la implementación del proyecto "Chatbot con Expresiones Regulares" desarrollado para la materia de Autómatas, Gramáticas y Lenguajes, cumpliendo con todos los requisitos especificados en las instrucciones del proyecto.*