

Gujarat vehicle registration

Vehicle plate registered in Gujarat — starts with GJ.

```
^GJ[\s\-]?\d{1,2}[\s\-]?[A-Z]{1,3}[\s\-]?\d{4}$
```

Token-by-token breakdown

Token	Meaning
^	start of string (or line in multiline mode)
GJ	literal text "GJ"
[\s\-]?	optional (zero or one): any of: whitespace, "-"
\d{1,2}	between 1 and 2 times: digit (0–9)
[\s\-]?	optional (zero or one): any of: whitespace, "-"
[A-Z]{1,3}	between 1 and 3 times: any of: uppercase letters
[\s\-]?	optional (zero or one): any of: whitespace, "-"
\d{4}	exactly 4 times: digit (0–9)
\$	end of string (or line in multiline mode)

Quick usage in 7 languages

JavaScript	<pre>const re = /^GJ[\s\-]?\d{1,2}[\s\-]?[A-Z]{1,3}[\s\-]?\d{4}\$/; re.test(value);</pre>
Python	<pre>import re re.match(r"^GJ[\s\-]?\d{1,2}[\s\-]?[A-Z]{1,3}[\s\-]?\d{4}\$", value)</pre>
Java	<pre>Pattern.compile("^GJ[\\s\\-]?\\d{1,2}[\\s\\-]?[A-Z]{1,3}[\\s\\-]?\\d{4}\$").matcher(value).matches();</pre>
C# / .NET	<pre>Regex.IsMatch(value, @"^GJ[\s\-]?\d{1,2}[\s\-]?[A-Z]{1,3}[\s\-]?\d{4}\$");</pre>
Go	<pre>regexp.MustCompile(`^GJ[\s\-]?\d{1,2}[\s\-]?[A-Z]{1,3}[\s\-]?\d{4}\$`).MatchString(value)</pre>
Ruby	<pre>/^GJ[\s\-]?\d{1,2}[\s\-]?[A-Z]{1,3}[\s\-]?\d{4}\$/ .match?(value)</pre>
PHP	<pre>preg_match('~^GJ[\s\-]?\d{1,2}[\s\-]?[A-Z]{1,3}[\s\-]?\d{4}\$~', \$value);</pre>

Common pitfalls

- **Anchored to the whole string.** This pattern uses ^ and \$, so it requires the entire input to match. To find it inside a longer text, drop the anchors and use the global (g) flag.
- **Uppercase only.** Ranges like [A-Z] won't match lowercase. Add a-z (or the i flag) if lowercase input should be accepted.
- **Escape it correctly per language.** In Java and JavaScript strings each backslash must be doubled (\d); in Python, Go, and C# use raw/verbatim strings so the backslashes survive.
- **Validate beyond format.** Matching the format doesn't guarantee the value is real. Confirm the gujarat vehicle registration against a source of truth (database, API, or checksum) where it matters.