

EPF Member ID

Member ID format — establishment code + member number.

```
^[A-Z]{2}/[A-Z]{3}/\d{7}/\d{3}/\d{7}$
```

Token-by-token breakdown

Token	Meaning
^	start of string (or line in multiline mode)
[A-Z]{2}	exactly 2 times: any of: uppercase letters
/	literal text "/"
[A-Z]{3}	exactly 3 times: any of: uppercase letters
/	literal text "/"
\d{7}	exactly 7 times: digit (0–9)
/	literal text "/"
\d{3}	exactly 3 times: digit (0–9)
/	literal text "/"
\d{7}	exactly 7 times: digit (0–9)
\$	end of string (or line in multiline mode)

Quick usage in 7 languages

JavaScript	<pre>const re = /^[A-Z]{2}\/[A-Z]{3}\d{7}\d{3}\d{7}\$/; re.test(value);</pre>
Python	<pre>import re re.match(r"^[A-Z]{2}/[A-Z]{3}/\d{7}/\d{3}/\d{7}\$", value)</pre>
Java	<pre>Pattern.compile("^[A-Z]{2}/[A-Z]{3}/\d{7}/\d{3}/\d{7}\$").matcher(value).matches();</pre>
C# / .NET	<pre>Regex.IsMatch(value, @"^[A-Z]{2}/[A-Z]{3}/\d{7}/\d{3}/\d{7}\$");</pre>
Go	<pre>regexp.MustCompile(`^[A-Z]{2}/[A-Z]{3}/\d{7}/\d{3}/\d{7}\$`).MatchString(value)</pre>
Ruby	<pre>/^[A-Z]{2}\/[A-Z]{3}\d{7}\d{3}\d{7}\$/ .match?(value)</pre>
PHP	<pre>preg_match('~^[A-Z]{2}/[A-Z]{3}/\d{7}/\d{3}/\d{7}\$~', \$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 epf member id against a source of truth (database, API, or checksum) where it matters.