1. Validation Patterns

1.1 Validate Email Addresses

- **Pattern**: `^[\w\.-]+@[a-zA-Z\d\.-]+\.[a-zA-Z]{2,6}$`
- **Example**: `"user@example.com"` (Valid), `"user@example"` (Invalid)

1.2 Match Valid IP Addresses

- **Pattern**: `^(?:\d{1,3}\.){3}\d{1,3}$`
- **Example**: `"192.168.1.1"` (Valid), `"256.256.256.256"` (Invalid)

1.3 Validate Dates in YYYY-MM-DD Format

- **Pattern**: `^(?:\d{4})-(?:0[1-9]|1[0-2])-(?:0[1-9]|[12]\d|3[01])$`
- **Example**: `"2023-05-21"` (Valid), `"2023-15-32"` (Invalid)

1.4 Validate Credit Card Numbers

- **Pattern**: `^(?:\d{4}[-\s]?){3}\d{4}$`
- **Example**: `"1234-5678-9012-3456"` (Valid), `"1234 5678 9012"` (Invalid)

1.5 Validate US ZIP Codes

- **Pattern**: `^\d{5}(-\d{4})?$`
- **Example**: `"12345"` (Valid), `"1234"` (Invalid)

---

2. Extraction Patterns

2.1 Extract Domain Names from URLs

- **Pattern**: `https?:\/\/(?:www\.)?([^\/]+)`

- **Example**: From `"https://www.example.com/page"`, extract `"example.com"`

2.2 Extract Quoted Strings

- **Pattern**: `['"](.*?)['"]`
- **Example**: In `'The "quick" fox'`, extract `"quick"`

2.3 Extract Phone Numbers

- **Pattern**: `\(?\d{3}\)?[-.\s]?\d{3}[-.\s]?\d{4}`
- **Example**: From `"Call (123) 456-7890"`, extract `"(123) 456-7890"`

2.4 Extract File Extensions from Filenames

- **Pattern**: `\.(\w+)$`
- **Example**: From `"photo.jpg"`, extract `"jpg"`

2.5 Extract Capitalized Words

- **Pattern**: `\b[A-Z][a-zA-Z]*\b`
- **Example**: From `"Hello World"`, extract `"Hello"`, `"World"`

---

3. Matching Patterns

3.1 Match Valid Hexadecimal Colors

- **Pattern**: `#([A-Fa-f0-9]{6}|[A-Fa-f0-9]{3})\b`
- **Example**: `"#FF5733"` (Match), `"123456"` (No match)

3.2 Match HTML Tags

- **Pattern**: `<\w+>`
- **Example**: `"<div>"` (Match), `"div"` (No match)

3.3 Match Balanced Parentheses Expressions

- **Pattern**: `\(([^()]*)\)`
- **Example**: From `"(a(bc)d)"`, extract `"(bc)"`

3.4 Match Time in HH

Format

- **Pattern**: `^(0[0-9]|1[0-9]|2[0-3]):([0-5][0-9])$`
- **Example**: `"23:59"` (Valid), `"25:00"` (Invalid)

3.5 Match Strong Passwords

- **Pattern**: `^(?=.*[A-Z])(?=.*[a-z])(?=.*\d)(?=.*[@$!%*?&])[A-Za-z\d@$!%*?&]{8,}$`
- **Example**: `"Passw0rd!"` (Valid), `"password"` (Invalid)

---

4. Pattern Variants

4.1 Find All Hashtags in Text

- **Pattern**: `#\w+`
- **Example**: From `"#love #coding"`, extract `"#love"`, `"#coding"`

**4.2

proper test string and then pattern

Certainly! Here are example test strings along with each regex pattern for validation and extraction:

---

1. Validation Patterns

1.1 Validate Email Addresses

- **Test String**: `"user@example.com"`, `"user@sub.example.com"`, `"user123@example-domain.org"`, `"user@.com"`
- **Pattern**: `^[\w\.-]+@[a-zA-Z\d\.-]+\.[a-zA-Z]{2,6}$`
- **Explanation**: This pattern checks for a standard email structure of `username@domain.extension` with allowed characters.

1.2 Match Valid IP Addresses

- **Test String**: `"192.168.1.1"`, `"255.255.255.255"`, `"256.100.100.100"`, `"123.456.789.0"`
- **Pattern**: `^(?:\d{1,3}\.){3}\d{1,3}$`
- **Explanation**: Matches IPv4 addresses with numbers between `0-255`. Does not verify ranges.

1.3 Validate Dates in YYYY-MM-DD Format

- **Test String**: `"2023-05-21"`, `"2023-12-31"`, `"2023-00-12"`, `"2023-11-31"`
- **Pattern**: `^(?:\d{4})-(?:0[1-9]|1[0-2])-(?:0[1-9]|[12]\d|3[01])$`
- **Explanation**: Ensures the year, month (01-12), and day (01-31) follow the correct date structure.

1.4 Validate Credit Card Numbers

- **Test String**: `"1234-5678-9012-3456"`, `"1234 5678 9012 3456"`, `"1234567890123456"`, `"1234-5678-901"`
- **Pattern**: `^(?:\d{4}[-\s]?){3}\d{4}$`
- **Explanation**: Matches credit card numbers in `#### #### #### ####` or `####-####-####-####` format.

1.5 Validate US ZIP Codes

- **Test String**: `"12345"`, `"12345-6789"`, `"123456"`, `"1234-567"`, `"ABCDE"`
- **Pattern**: `^\d{5}(-\d{4})?$`
- **Explanation**: Matches ZIP codes in `12345` or `12345-6789` formats.

---

2. Extraction Patterns

2.1 Extract Domain Names from URLs

- **Test String**: `"https://www.example.com/page"`, `"http://sub.example.co.uk/path"`, `"https://example.org"`
- **Pattern**: `https?:\/\/(?:www\.)?([^\/]+)`
- **Explanation**: Extracts the main domain name without `www` or paths.

2.2 Extract Quoted Strings

- **Test String**: `"She said, 'hello' and then left."`, `'The "quick" fox jumped.'`, `"It's a sunny day."`

- **Pattern**: `['"](.*?)['"]`
- **Explanation**: Captures any string enclosed in single (`' '`) or double (`" "`) quotes.

2.3 Extract Phone Numbers

- **Test String**: `"(123) 456-7890"`, `"123-456-7890"`, `"1234567890"`, `"123.456.7890"`
- **Pattern**: `\(?\d{3}\)?[-.\s]?\d{3}[-.\s]?\d{4}`
- **Explanation**: Matches phone numbers with formats like `(123) 456-7890`, `123-456-7890`, or `1234567890`.

2.4 Extract File Extensions from Filenames

- **Test String**: `"document.pdf"`, `"image.jpeg"`, `"archive.tar.gz"`, `"README"`
- **Pattern**: `\.(\w+)$`
- **Explanation**: Captures the file extension after the last `.` in filenames.

2.5 Extract Capitalized Words

- **Test String**: `"The Quick Brown Fox jumped Over the Lazy Dog."`
- **Pattern**: `\b[A-Z][a-z]*\b`
- **Explanation**: Matches words that start with a capital letter.