//REMEMBER TO ADD .strict() to all Zod schemas. This disallows any unknown keys.

//ZOD ALLOWS UNKNOWN KEYS BY DEFAULT, BUT STRIPS THEM OUT DURING VALIDATION.

|  |  |  |
| --- | --- | --- |
| **Library** | **Joi** | **Express-Validator** |
| **Validators** | **array**() [Generates a schema object that matches an array data type.]  array.has() -> Verifies that a schema validates at least one of the values in the array.  array.length() -> Specifies the exact number of items in the array.  array.max(limit) -> Specifies the maximum number of items in the array.  array.min() -> Specifies the minimum no. of items in the array.  array.sort() -> Requires the array to comply with the specified sort order.  array.sparse() -> Allows this array to be sparse.  array.unique() -> Requires the array values to be unique.  **Supports Binary Data**  **boolean**() -> Generates a schema object that matches a boolean data type.  boolean.sensitive() -> Restrict the values provided to truthy and falsy as well as the 'true' and 'false' default conversions.  boolean.falsy() -> Allows for additional values to be considered valid booleans by converting them to false during validation.  boolean.truthy() -> Allows for additional values to be considered valid booleans by converting them to false during validation.  **number**() -> Generates a schema object that matches a number data type (as well as strings that can be converted to numbers).  number.integer() -> Requires the number to be an integer  number.greater() -> Specifies that the number has to be greater than a certain value.  number.less() -> Specifies that the number has to be less than a certain value.  number.max() -> Specifies that the maximum possible value of the number.  number.min() -> Specifis the minimum possible value of the number.  number.multiple() -> Specifies that the value must be a multiple of a value.  number.negative() -> Requires the number to be negative.  number.positive() -> Requires the number to be positive.  number.port() -> Requires the number to be a port between 0 and 65,535.  number.precision() -> Specifies the maximum number of decimal places.  number.unsafe() -> Allows unsafe numbers, i.e., those outside JS’s numeric safety range.  **date**() -> Generates a schema object that matches a date type.  date.greater() -> Specifies that the value must be greater than a given value.  date.iso() -> Requires the string value to be in valid ISO 8601 date format.  date.less() -> Specifies that the value must be less than the given date value.  date.max() -> Specifies the latest date allowed.  date.min() -> Specifies the oldest date allowed.  date.timestamp() -> Requires the value to be a timestamp interval.  **string()** -> Generates a schema object that matches a string data type. By default, empty strings aren’t allowed.  string.alphanum() -> Requires the string value to only contain a-z, A-Z, and 0-9.  string.base64() -> Requires the string to be a valid base64 string. Doesn’t check decoded value.  string.creditCard() -> Requires the number to be a credit card number.  string.dataUri() -> Requires the string value to be a valid data URI string.  string.domain() -> Requires the string value to be a valid domain name.  string.email() -> Requires the string value to be a valid email address.  string.guid() -> Requires the string value to be a valid GUID.  string.hostname() -> Requires the string value to be a valid hostname as per RFC1123.  string.insensitive() -> Allows the value to match any value in the allowed list or disallowed list in a case insensitive comparison.  string.ip() -> Requires the string value to be a valid ip address.  string.isoDate() -> Requires the string value to be in valid ISO 8601 date format.  string.isoDuration() -> Requires the string value to be in valid ISO 8601 duration format.  string.normalize() -> Requires the string value to be in a Unicode normalized form.  string.pattern() -> Defines a pattern rule.  string.uri() -> Requires the string value to be a valid RFC 3986 URI. | contains(str, seed) -> Checks if the string contains the seed.  equals(str, comparison) -> Checks if the string matches the comparison.  isAfter(str [, date]) -> Checks if the string is a date that's after the specified date.  isAlpha(str [, locale, options]) -> Checks if the string contains only letters (a-zA-Z).  isAlphanumeric(str [, locale, options]) -> Checks if the string contains only letters and numbers.  isAscii(str) -> Checks if the string contains only ASCII characters.  isBase32(str) -> Checks if the string is Base32 encoded.  isBase58(str) -> Checks if the string is Base58 encoded.  isBase64(str) -> Checks if the string is Base64 encoded.  isBefore(str [, date]) -> Checks if the string is a date that's before the specified date.  isBIC(str) -> Checks if the string is a valid BIC (Bank Identifier Code).  isBoolean(str) -> Checks if the string is a boolean.  isBtcAddress(str) -> Checks if the string is a valid Bitcoin address.  isByteLength(str, options) -> Checks if the string’s length (in bytes) falls within the specified range.  isCreditCard(str) -> Checks if the string is a credit card number.  isCurrency(str, options) -> Checks if the string is a valid currency amount.  isDataURI(str) -> Checks if the string is a data URI format.  isDate(str [, options]) -> Checks if the string is a valid date.  isDecimal(str [, options]) -> Checks if the string is a decimal number.  isDivisibleBy(str, number) -> Checks if the string is a number divisible by the given number.  isEAN(str) -> Checks if the string is a valid EAN (European Article Number).  isEmail(str [, options]) -> Checks if the string is an email.  isEmpty(str [, options]) -> Checks if the string is empty.  isEthereumAddress(str) -> Checks if the string is a valid Ethereum address.  isFQDN(str [, options]) -> Checks if the string is a fully qualified domain name.  isFloat(str [, options]) -> Checks if the string is a float.  isFullWidth(str) -> Checks if the string contains any full-width characters.  isHalfWidth(str) -> Checks if the string contains any half-width characters.  isHash(str, algorithm) -> Checks if the string is a hash of the specified algorithm.  isHexColor(str) -> Checks if the string is a hexadecimal color.  isHexadecimal(str) -> Checks if the string is a hexadecimal number.  isHSL(str) -> Checks if the string is a valid HSL color.  isIBAN(str) -> Checks if the string is a valid IBAN.  isIdentityCard(str [, locale]) -> Checks if the string is a valid identity card number.  isIMEI(str [, options]) -> Checks if the string is a valid IMEI.  isIP(str [, version]) -> Checks if the string is an IP (v4 or v6).  isIPRange(str) -> Checks if the string is a valid IP range.  isISBN(str [, version]) -> Checks if the string is a valid ISBN.  isISIN(str) -> Checks if the string is a valid ISIN (stock ticker).  isISO31661Alpha2(str) -> Checks if the string is a valid ISO 3166-1 alpha-2 country code.  isISO31661Alpha3(str) -> Checks if the string is a valid ISO 3166-1 alpha-3 country code.  isISO4217(str) -> Checks if the string is a valid ISO 4217 currency code.  isISO8601(str [, options]) -> Checks if the string is a valid ISO 8601 date.  isISRC(str) -> Checks if the string is a valid ISRC.  isISSN(str [, options]) -> Checks if the string is a valid ISSN.  isIn(str, values) -> Checks if the string is in an array of allowed values.  isInt(str [, options]) -> Checks if the string is an integer.  isJSON(str [, options]) -> Checks if the string is valid JSON.  isJWT(str) -> Checks if the string is a valid JWT.  isLatLong(str [, options]) -> Checks if the string is a valid latitude/longitude.  isLength(str, options) -> Checks if the string's length falls within a range.  isLicensePlate(str [, locale]) -> Checks if the string is a valid license plate.  isLocale(str) -> Checks if the string is a valid locale code.  isLowercase(str) -> Checks if the string is lowercase.  isMACAddress(str [, options]) -> Checks if the string is a MAC address.  isMagnetURI(str) -> Checks if the string is a valid magnet URI.  isMD5(str) -> Checks if the string is an MD5 hash.  isMimeType(str) -> Checks if the string is a valid MIME type.  isMobilePhone(str [, locale, options]) -> Checks if the string is a valid mobile phone number.  isMongoId(str) -> Checks if the string is a valid MongoDB ObjectId.  isMultibyte(str) -> Checks if the string contains one or more multibyte characters.  isNumeric(str [, options]) -> Checks if the string contains only numbers.  isOctal(str) -> Checks if the string is a valid octal number.  isPassportNumber(str, countryCode) -> Checks if the string is a valid passport number for the given country.  isPort(str) -> Checks if the string is a valid port number.  isPostalCode(str, locale) -> Checks if the string is a valid postal code.  isRFC3339(str) -> Checks if the string is a valid RFC 3339 date.  isRgbColor(str [, includePercentValues]) -> Checks if the string is a valid RGB color.  isSemVer(str) -> Checks if the string is a valid semantic version.  isSlug(str) -> Checks if the string is a valid slug.  isStrongPassword(str [, options]) -> Checks if the string is a strong password.  isSurrogatePair(str) -> Checks if the string contains any surrogate pairs.  isTaxID(str, locale) -> Checks if the string is a valid Tax Identification Number.  isTime(str [, options]) -> Checks if the string is a valid time format.  isURL(str [, options]) -> Checks if the string is a valid URL.  isUUID(str [, version]) -> Checks if the string is a UUID.  isUppercase(str) -> Checks if the string is uppercase.  isVariableWidth(str) -> Checks if the string contains both full-width and half-width characters.  isVAT(str, countryCode) -> Checks if the string is a valid VAT number.  isWhitelisted(str, chars) ->  Checks if the string contains only allowed characters. |
| **Sanitizers** | string.replace() -> Replace characters matching the given pattern with the specified replacement string.  string.trim() -> Requires the string value to contain no whitespace before or after. If the validation convert option is on (enabled by default), the string will be trimmed.  string.truncate() -> Requires the string value to contain no whitespace before or after. If the validation convert option is on (enabled by default), the string will be trimmed. | blacklist(input, chars) -> Removes characters from the input that match the blacklist.  escape(input) -> Replaces `<`, `>`, `&`, `'`, `"` with HTML entities.  unescape(input) -> Decodes HTML entities back into their corresponding characters.  ltrim(input [, chars]) -> Trims characters from the left side of the input.  rtrim(input [, chars]) -> Trims characters from the right side of the input.  trim(input [, chars]) -> Trims characters from both sides of the input.  normalizeEmail(email [, options]) -> Normalizes an email address (lowercases the domain, optionally Gmail-specific behavior).  stripLow(input [, keep\_new\_lines]) -> Removes ASCII control characters, optionally preserving new lines.  toBoolean(input [, strict]) -> Converts the input string to a boolean (`true` or `false`).  toDate(input) -> Converts the input string to a Date object, if possible.  toFloat(input) -> Converts the input string to a float.  toInt(input [, radix]) -> Converts the input string to an integer, optionally with a radix.  toString(input) -> Converts the input to a string.  whitelist(input, chars) -> Removes all characters from the input that are not in the whitelist. |