

Perfect 🍌 Let's build a **cheat-sheet table** that you can directly use when writing JSON Schema for RestAssured tests or in interviews.

JSON Schema Cheat-Sheet

Field in JSON	Schema Keyword	Example Schema
String	type: "string"	"name": { "type": "string" }
Integer	type: "integer"	"age": { "type": "integer", "minimum": 0 }
Number (decimal)	type: "number"	"price": { "type": "number", "minimum": 0.0 }
Boolean	type: "boolean"	"isActive": { "type": "boolean" }
Null	type: "null"	"middleName": { "type": "null" }
Object	type: "object", properties: { ... }	"user": { "type": "object", "properties": { "id": { "type": "integer" } } }
Array of strings	type: "array", items: { "type": "string" }	"tags": { "type": "array", "items": { "type": "string" } }
Array of objects	type: "array", items: { "type": "object" }	"products": { "type": "array", "items": { "type": "object", "properties": { "id": { "type": "integer" }, "title": { "type": "string" } }, "required": ["id", "title"] } }
Required fields	required: [...]	"required": ["id", "name"]
Regex pattern (string)	pattern	"email": { "type": "string", "pattern": "^[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\$" }
Enum (fixed values)	enum	"status": { "type": "string", "enum": ["active", "inactive"] }
String length	minLength , maxLength	"username": { "type": "string", "minLength": 3, "maxLength": 15 }
Number range	minimum , maximum	"age": { "type": "integer", "minimum": 18, "maximum": 99 }
Additional fields allowed?	additionalProperties	"additionalProperties": false (strict)
Date-time / Email / URI	format	"createdAt": { "type": "string", "format": "date-time" }

Example 1: Simple Object

```
json
{
  "id": 1,
  "name": "Sumit",
  "email": "sumit@example.com"
}
```

Schema:

```
json
{
  "type": "object",
  "properties": {
    "id": { "type": "integer" },
    "name": { "type": "string", "minLength": 1 },
    "email": { "type": "string", "format": "email" }
  },
  "required": ["id", "name", "email"],
  "additionalProperties": false
}
```

Example 2: Array of Objects

```
json
[
  { "id": 101, "title": "Movie A" },
  { "id": 102, "title": "Movie B" }
]
```

Schema:

```
json
{
  "type": "array",
  "items": {
    "type": "object",
    "properties": {
      "id": { "type": "integer" },
      "title": { "type": "string" }
    },
    "required": ["id", "title"],
    "additionalProperties": false
  }
}
```

✅ With this cheat-sheet, you can write **any schema** for API response validation in RestAssured.

Would you like me to also show you **how to integrate this schema into RestAssured test cases** step by step (like where to place `.body(JsonSchemaValidator.matchesJsonSchema(...))`)?

” Ask
ChatGPT