Spezifikation des REST API „Werkstatt Auftragsverwaltung“

Kurzbeschreibung

Die Werkstatt Auftragsverwaltung API ermöglicht die Verwaltungvon Reparaturaufträgen (Work Orders). Jeder Auftrag besitzt eine eindeutige ID, ein Kennzeichen, eine Beschreibung, einen Status und ein Fälligkeitsdatum. Die API unterstützt Operationen zum Anlegen neuer Aufträge, Löschen bestehender Aufträge, Aktualisieren von Auftragsdaten sowie zum Abfragen spezifischer oder aller Aufträge (inklusive Filterung).

Datenstrukturen / Schema:

Name Datenstruktur	WorkOrder	**Allgemeine Beschreibung**	Ein Auftrag (WorkOrder) repräsetiert eine Arbeit an einem Fahrzeug	
Name Komponente	**Datentyp**	**Einschränkungen**	**Kategorie**	**Bemerkung**
id	Integer	Zwischen 0 Integer.MAX	Optional	Eindeutige ID, servergeneriert
licensePlate	String	Not Empty, maximal 20 Zeichen	Mandatory	KFZ-Kennzeichnen
description	String	Not Empty, maximal 255 Zeichen	Mandatory	Beschreibung der Arbeit
status	String	Enum: PENDING, IN_PROGRESS, COMPLETED	Mandatory	Bearbeitungsstatus
dueDate	String	Format: YYYY-MM-DD	Mandatory	Fälligkeitsdatum
Beispiel	“SB-XY-123", “Ölwechsel”, “PENDING”, “2025-12-24"			

Tabelle 1: Datenstrukturbeschreibung der Ressource „WorkOrder“

Format der Datenübertragung

Content-Type: `application/json`

Daten werden im JSON-Format im Body übertragen.

Beispiel für ein Request-Body (zum Erstellen)

```
{
  "licensePlate": "SB-XY-123",
  "description": "Bremsscheiben wechseln",
  "status": "PENDING",
  "dueDate": "2025-
10-15"
}
```

Beispiel für ein Response-Body

```
{
  "id": 101,
  "licensePlate": "SB-XY-123",
  "description": "Bremsscheiben wechseln",
  "status": "PENDING",
  "dueDate": "2025-
10-15"
}
```

JSON Schema (Auszug)

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "title": "WorkOrder-Objekt",
  "description": "Struktur eines einzelnen Reparaturauftrags",
  "type": "object",
  "required": [
    "licensePlate",
    "description",
    "status",
    "dueDate"
  ],
  "properties": {
    "id": {
      "type": "integer",
      "description": "servergenerierte eindeutige ID",
      "minimum": 0
    },
    "licensePlate": {
      "type": "string",
      "description": "KFZ - Kennzeichen des Kundenfahrzeugs",
      "minLength": 1,
      "maxLength": 20
    },
    "description": {
      "type": "string",
      "description": "Beschreibung der durchzufuehrenden Arbeit",
      "minLength": 1,
      "maxLength": 255
    },
    "status": {
      "type": "string",
      "description": "Aktueller Status des Auftrags",
      "enum": [
        "PENDING",
        "IN_PROGRESS",
        "COMPLETED"
      ]
    },
    "dueDate": {
      "type": "string",
      "description": "Faelligkeitsdatum (ISO 8601)",
      "format": "date"
    }
  },
  "additionalProperties": false
}
```

Quellcode 1: JSON Schema eines workorder

Übersicht über die Ressourcen und Methoden

Path / Ressource/ end point	Methode			
	POST (create)	GET (read)	PUT (update)	DELETE (delete)
/workorders	X	X	-	-
/workorders /{id}	-	X	X	X

Tabelle 2: Dienst-Übersicht des REST-API „Werkstatt Auftragsverwaltung"

Fehlerbehandlung

Name Datenstruktur	Error	**Allgemeine Beschreibung**	Standardisierte Struktur von Fehlermeldungen des APIs	
Name Komponente	**Datentyp**	**Einschränkungen**	**Kategorie**	**Bemerkung**
message	String	Not empty	Mandatory	Kurze, allgemeine Fehlermeldung zur Art des Problems
Detail	String	Not Empty	Mandatory	Detailliertere Beschreibung des Fehlers oder Hinweise zur Behebung.
path	String	Not Empty	Mandatory	der API-Endpunkt, bei dem der Fehler aufgetreten ist
Beispiel	message = date is in wrong format detail = indicate a valid date path = /workorders			

Fehlermeldungen werden im JSON-Format im HTTP-Response übermittelt. Das Fehlerobjekt enthält die Felder `message`, `detail` und `path`.

Beispiel für einen Fehler-Response (400 Bad Request)

```
HTTP/1.1 400  Bad Request
Content-Type: application/json
{
    "message":"Wrong format for dueDate",
    "detail":"Date must be in YYYY-MM-DD format",
    "path":"/workorders"
}
```

Quellcode 2: Beispiel für einen Fehler-Response

Nicht erlaubte Methoden (405 Method Not Allowed)

Wird eine nicht unterstützte HTTP-Methode oder eine nicht erlaubte HTTP-Methode für eine Ressource aufgerufen, so wird der Fehlercode 405 zurückgegeben und der `Allow`-Header hinzugefügt.

```
HTTP/1.1 405 Method Not Allowed
Allow: GET, POST
Content-Type: application/json
{
      "message": "Method 'PUT' not allowed for resource '/workorders'",
      "detail": "Supported methods are GET, POST",
      "path": "/workorders"
}
```

Quellcode 3: Bespiel Method Not Allowed

Resource nicht gefunden (404 not Found)

Bei einer Anfrage einer Ressource, eines Pfads, der nicht existiert, wird 404 Font Found geantwortet.

```
HTTP/1.1 404 Not Found
Content-Type: application/json
{
  "message": "Resource not found",
  "detail": "The requested URL '/workorder' does not exist on this
server.",
  "path": "/workorder"
}
```

Sicherheitsmechanismen

Die Authentifizierung und Autorisierung erfolgen über **OAuth 2.0**. Alle Anfragen müssen verschlüsselt über **HTTPS** übertragen werden.

Spezifikation der einzelnen Ressourcen

1. API-Methoden der Ressource `/workorders`

Method: POST, Path: /workorders

Es wird ein neuer Workorder in die Liste eingetragen

Request

http-Body	Beschreibung	Content-Type
Workorder	Es wird ein neuer Auftrag angelegt	application/json

Response

Response Header	Response Body	Beschreibung
http-Status Code 201	WorkOrder mit ID	Ressource erfolgreich erzeugt.
location : <URL>		URL der neu erzeugten Ressource
400 Bad Request	Fehlerobjekt	Ungültige Eingabedaten.

Beispiel

Request

```
POST /workorders HTTP/1.1
Host: localhost: 8080
Content-Type: application/json

{
    "licensePlate": "HH-OP-42",
    "description": "Reifenwechsel",
    "status": "PENDING",
    "dueDate": "2025-11-01"

}
```

Response

```
HTTP/1.1 201 Created
Location: workorder/1
{
    "id": 1,
    "licensePlate": "HH-OP-42",
    "description": "Reifenwechsel",
    "status": "PENDING",
    "dueDate": "2025-11-01"
}
```

Method: GET, Path: /workorders

Gibt die Liste aller WorkOrders zurück. Unterstützt Filterung

Request

Query Params (Optional):

- **status**: Filter nach Status.
- **licensePlate**: Filter nach Kennzeichen.
- **dueDate**: Filter nach Datum.

http-Body	**Beschreibung**
leer	

Response

Response Header	**Response Body**	**Beschreibung**
http-Status Code 200 OK	Liste der Workorder	Liste der workorder
http-Status Code 404 Not Found	Fehlerobjekt	Es sind keine Workorder vorhanden.

Beispiel

Request	`GET /workorders HTTP/1.1` `Host: localhost:4711`

Response	`HTTP/1.1 200 OK`
	`Content-Type: application/json`
	
	`[`
	
	`  {`
	`  "id": 2,`
	`  "licensePlate": "KL-AA-99",`
	`  "description": "Inspektion",`
	`  "status": "IN_PROGRESS",`
	`  "dueDate": "2025-10-20"`
	`} ]`

2. API-Methoden der Ressource `/workorders/{id}`

Method: GET, Path: /workorders/{id}

Gibt den Workorder mit der ID {id} zurück, sofern vorhanden.

Request

Path Parameter:

Name	Beschreibung	Datentyp	Einschränkungen
id	Eindeutige ID eines Benutzers	Ganze Zahl	zwischen 0 und INTEGER.MAX

http-Body	**Beschreibung**
leer	

Response

Response Header	**Response Body**	**Beschreibung**
http-Status Code 200 OK	WorkOrder	Auftrag gefunden
http-Status Code 404 Not Found	Fehlerobjekt	ID existiert nicht

Beispiel

Request	`GET/workorders /1 HTTP/1.1`
	`Host: localhost:8080`
	`Content-Type: application/json`

Response	`HTTP/1.1 200 Ok`
	`Content-Type: application/json`
	`{`
	`"licensePlate": "HH-OP-42",`
	`"description": "Reifenwechsel",`
	`"status": "COMPLETED",`
	`"dueDate": "2025-11-01"`
	`}`

Method: PUT, Path: /workorders/{id}

Aktualisiert Daten des Workorders mit der ID {id} sofern vorhanden.

Request

Path Parameter:

Name	Beschreibung	Datentyp	Einschränkungen
id	WorkOrder mit aktualisierten Daten.	Ganze Zahl	zwischen 0 und INTEGER.MAX

http-Body	**Beschreibung**	**Content-Type**
WorkOrder	aktualisierte Daten des Workorder mit der ID {id}, die id im Body wird ignoriert.	application/json

Response

Response Header	**Response Body**	**Beschreibung**
200 Ok	leer	Daten wurden aktualisiert
400 Bad Request	Fehlerobjekt	1. id ist keine ganze Zahl 2. id ist eine Zahl < 0 oder > INTEGER.MAX 3. Workorder im falschen Format, hat fehlende Felder oder falsche Werte
404 Not Found	Fehlerobjekt	Workorder mit der ID id existiert nicht.

Beispiel

Request	PUT /workorders /67 HTTP/1.1 Host: localhost: 8080 Content-Type: application/json; charset=utf-8 } "licensePlate": "HH-OP-42", "description": "Reifenwechsel", "status": "COMPLETED", "dueDate": "2025-11-01" }
Response	HTTP/1.1 200 Ok Content-Type: application/json

Request	PUT /workorders /xx HTTP/1.1 Host: localhost: 4711

Response	HTTP/1.1 400 Bad Request
	Content-Type: application/json
	{
	"message":"wrong format for id",
	"detail":"use a number between 0 and INTEGER.MAX"
	"path":"/workorders/xx"
	}

Request	PUT /workorders /21 HTTP/1.1 Host: localhost: 8080

Response	HTTP/1.1 404 Not Found
	Content-Type: application/json
	{
	"message":"workorder with id 21 does not exist",
	"detail":"use a valid workorder id"
	"path":"/workorders/21"
	}

Method: DELETE, Path: /workorders/{id}

Löscht den Workorder mit der ID {id}, sofern vorhanden.

Request

Path Parameter:

Name	Beschreibung	Datentyp	Einschränkungen
id	Eindeutige ID eines Workorders	Ganze Zahl	zwischen 0 und INTEGER.MAX

http-Body	**Beschreibung**
leer	Der Request-Body ist leer

Response

Response Header	**Response Body**	**Beschreibung**
200 Ok	leer	Workorder mit angegebener id wurde gelöscht
400 Bad Request	Fehlerbeschreibung	id im Pfad ist keine Zahl oder ungültig
404 Not Found	Fehlerbeschreibung	Workorder mit der angegebenen id existiert nicht.

Beispiel

Request	`DELETE /workorders /67 HTTP/1.1` `Host: localhost: 8080`
Response	`HTTP/1.1 200 Ok` `Content-Type: application/json`

Request	`DELETE /workorders /xx HTTP/1.1` `Host: localhost: 8080`

Response	`HTTP/1.1 400 Bad Request`
	`Content-Type: application/json`
	`{`
	`"message":"wrong format for id",`
	`"detail":"use a number between 0 and INTEGER.MAX"`
	`"path":"/workorders/xx"`
	`}`

Request	`DELETE /workorders /21 HTTP/1.1` `Host: localhost: 8080`

Response	`HTTP/1.1 404 Not Found`
	`Content-Type: application/json`
	`{`
	`"message":"workorder with id 21 does not exist",`
	`"detail":"use a valid workoder id"`
	`"path":"/workorders/21"`
	`}`