# Endpoint: Edit Agent

The `https://swarms.world/api/edit-agent` endpoint allows users to edit an existing agent on the Swarms platform. This API accepts a POST request with a JSON body containing the agent details to be updated, such as its id, name, description, use cases, language, tags and requirements. The request must be authenticated using an API key.

## Endpoint

- **URL:** `https://swarms.world/api/edit-agent`
- **Method:** POST
- **Content-Type:** `application/json`
- **Authorization:** Bearer token required in the header

## Request Parameters

The request body should be a JSON object with the following attributes:

Attribute	Type	Description	Required
`id`	`string`	The ID of the agent to be edited.	Yes
`name`	`string`	The name of the agent.	Yes
`agent`	`string`	The agent text.	Yes
`description`	`string`	A brief description of the agent.	Yes
`language`	`string`	The agent's syntax language	No

`useCases`	`array`	An array of use cases, each containing a title and description.	Yes
`requirements`	`array`	An array of requirements, each containing a package name and installation.	Yes
`tags`	`string`	Comma-separated tags for the agent.	No

### `useCases` Structure

Each use case in the `useCases` array should be an object with the following attributes:

Attribute	Type	Description	Required
`title`	`string`	The title of the use case.	Yes
`description`	`string`	A brief description of the use case.	Yes

### `requirements` Structure

Each requirement in the `requirements` array should be an object with the following attributes:

Attribute	Type	Description	Required
`package`	`string`	The name of the package.	Yes
`installation`	`string`	Installation command for the package	Yes

## Example Usage

### Python

```python
import requests
import json

url = "https://swarms.world/api/edit-agent"
headers = {
    "Content-Type": "application/json",
    "Authorization": "Bearer {apiKey}"
}
data = {
    "id": "agent_id",
    "name": "Updated agent",
    "agent": "This is an updated agent from an API route.",
    "description": "Updated description of the agent.",
    "language": "javascript",
    "useCases": [
        {"title": "Updated use case 1", "description": "Updated description of use case 1"},
        {"title": "Updated use case 2", "description": "Updated description of use case 2"}
    ],
    "requirements": [
        { "package": "express", "installation": "npm install express" },
        { "package": "lodash", "installation": "npm install lodash" },
    ],
    "tags": "updated, agent"
}


response = requests.post(url, headers=headers, data=json.dumps(data))
print(response.json())
```

### Node.js

```javascript
const fetch = require("node-fetch");


async function editAgentHandler() {
  try {
    const response = await fetch("https://swarms.world/api/edit-agent", {
      method: "POST",
      headers: {
        "Content-Type": "application/json",
        Authorization: "Bearer {apiKey}",
      },
      body: JSON.stringify({
        id: "agent_id",
        name: "Updated agent",
        agent: "This is an updated agent from an API route.",
        description: "Updated description of the agent.",
        language: "javascript",
        useCases: [
     {
      title: "Updated use case 1",
      description: "Updated description of use case 1",
     },
     {
      title: "Updated use case 2",
      description: "Updated description of use case 2",
     },
    ],
    requirements: [
     { package: "express", installation: "npm install express" },
     { package: "lodash", installation: "npm install lodash" },
    ],
    tags: "updated, agent",
   }),
  });

  const result = await response.json();
  console.log(result);
 } catch (error) {
  console.error("An error has occurred", error);
 }
}

editAgentHandler();
```

Go

```go
package main

import (
    "bytes"
    "encoding/json"
    "fmt"
    "net/http"
)

func main() {
    url := "https://swarms.world/api/edit-agent"
    payload := map[string]interface{}{
        "id":          "agent_id",
        "name":        "Updated Agent",
        "agent":       "This is an updated agent from an API route.",
        "description": "Updated description of the agent.",
        "language": "javascript",
        "useCases": []map[string]string{
            {"title": "Updated use case 1", "description": "Updated description of use case 1"},
            {"title": "Updated use case 2", "description": "Updated description of use case 2"},
        },
        "requirements": []map[string]string{
            {"package": "express", "installation": "npm install express"},
            {"package": "lodash", "installation": "npm install lodash"},
        },
        "tags": "updated, agent",
    }
    jsonPayload, _ := json.Marshal(payload)

    req, _ := http.NewRequest("POST", url, bytes.NewBuffer(jsonPayload))
    req.Header.Set("Content-Type", "application/json")
    req.Header.Set("Authorization", "Bearer {apiKey}")

    client := &http.Client{}
    resp, err := client.Do(req)
    if err != nil {
        fmt.Println("An error has occurred", err)
        return
    }
    defer resp.Body.Close()

    var result map[string]interface{}
    json.NewDecoder(resp.Body).Decode(&result)
    fmt.Println(result)
}
```

### cURL

```bash
curl -X POST https://swarms.world/api/edit-agent \
-H "Content-Type: application/json" \
-H "Authorization: Bearer {apiKey}" \
-d '{
   "id": "agent_id",
   "name": "Updated agent",
   "agent": "This is an updated agent from an API route.",
   "description": "Updated description of the agent.",
   "language": "javascript",
   "useCases": [
      {"title": "Updated use case 1", "description": "Updated description of use case 1"},
      {"title": "Updated use case 2", "description": "Updated description of use case 2"}
   ],
   "requirements": [
      { "package": "express", "installation": "npm install express" },
      { "package": "lodash", "installation": "npm install lodash" },
   ],
   "tags": "updated, agent"
}'
```

## Response

The response will be a JSON object containing the result of the operation. Example response:

```json
{
  "success": true,
  "message": "Agent updated successfully",
  "data": {
    "id": "agent_id",
    "name": "Updated agent",
    "agent": "This is an updated agent from an API route.",
    "description": "Updated description of the agent.",
    "language": "javascript",
    "useCases": [
      {
        "title": "Updated use case 1",
        "description": "Updated description of use case 1"
      },
      {
        "title": "Updated use case 2",
        "description": "Updated description of use case 2"
      }
    ],
    "requirements": [
      { "package": "express", "installation": "npm install express" },
      { "package": "lodash", "installation": "npm install lodash" }
    ],
    "tags": "updated, agent"
```

  }
}
```

In case of an error, the response will contain an error message detailing the issue.

## Common Issues and Tips

- **Authentication Error:** Ensure that the `Authorization` header is correctly set with a valid API key.
- **Invalid JSON:** Make sure the request body is a valid JSON object.
- **Missing Required Fields:** Ensure that all required fields (`name`, `agent`, `description`, `useCases`, `requirements`) are included in the request body.
- **Network Issues:** Verify network connectivity and endpoint URL.

## References and Resources

- [API Authentication Guide](https://swarms.world/docs/authentication)
- [JSON Structure Standards](https://json.org/)
- [Fetch API Documentation (Node.js)](https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API)
- [Requests Library (Python)](https://requests.readthedocs.io/)
- [Net/HTTP Package (Go)](https://pkg.go.dev/net/http)

This comprehensive documentation provides all the necessary information to effectively use the `https://swarms.world/api/add-agent` and `https://swarms.world/api/edit-agent` endpoints, including details on request parameters, example code snippets in multiple programming languages, and
```

troubleshooting tips.