Webhook Handler API

This is a simple Node.js + Express API that receives webhook notifications, processes them, and forwards them to a third-party service. It demonstrates webhook handling, data processing, and secure forwarding using native Node.js modules.

What you'll get to know

- How to run a Node.js/Express server
- Receiving and processing webhook payloads
- Forwarding webhooks to third-party services
- Using native Node.js `https` module for HTTP requests
- Input validation and error handling
- Request logging middleware

Prerequisites

- Node.js (recommend v20+)
- npm (comes with Node)

Quick start

1. Install dependencies

```
npm install
```

2. Start the server

```
npm start
# Or you can start directly using node:
node server.js
```

3. The API will be available at: `http://localhost:3000`

Configuration

The webhook target URL is hardcoded in the server code as `https://third-party-webhook.example.com/notify`. To change it, edit the `WEBHOOK_TARGET_URL` constant in `server.js`.

Set the following environment variables:

- `PORT`: Port to run the server on (optional, default: 3000)

Endpoints

Webhook Receiver

- **POST /webhook** — Receive and forward webhook notifications
    - **Content-Type:** `application/json`
    - **Body:** JSON payload from the webhook source
    - **Returns:** JSON response with processing status

Example Usage

POST /webhook

Request body:

```
{
  "event": "user.created",
  "user": {
    "id": 123,
    "name": "John Doe",
    "email": "john@example.com"
  },
  "timestamp": "2023-11-01T12:00:00Z"
}
```

Response:

```
{
  "status": "webhook processed and forwarded",
  "thirdPartyStatus": 200,
  "thirdPartyResponse": "{\"received\": true}"
}
```

Features

- **Webhook Processing**: Receives JSON payloads and logs them for processing
- **Secure Forwarding**: Forwards webhooks to configured third-party endpoints
- **Native Node.js**: Uses built-in `https` module (no external HTTP libraries needed)
- **Request Logging**: All requests are logged to the console with timestamps
- **Error Handling**: Proper HTTP status codes and error messages
- **CORS Support**: Allows cross-origin requests for webhook integrations

Try it with curl (Windows PowerShell examples)

```
# Send a test webhook
curl -X POST http://localhost:3000/webhook `
  -H "Content-Type: application/json" `
  -d '{"event":"test","message":"Hello World"}' | ConvertFrom-Json

# Check server logs for processing details
```

Try it with PowerShell

```
# Send webhook using Invoke-RestMethod
$body = @{
  event = "user.created"
  user = @{
    id = 123
    name = "John Doe"
  }
} | ConvertTo-Json

Invoke-RestMethod -Uri "http://localhost:3000/webhook" -Method Post -Body $body -ContentType
"application/json"
```

What you can try next

1. Add webhook signature verification for security
2. Implement retry logic for failed third-party requests
3. Add rate limiting to prevent webhook spam
4. Store webhook history in a database
5. Add webhook filtering and transformation logic
6. Implement webhook queuing for high-volume scenarios
7. Add monitoring and alerting for webhook failures
8. Support multiple third-party targets with routing logic

Technical Notes

- Uses native Node.js `https` module for secure HTTP requests
- JSON payloads are parsed and forwarded as-is to the third party
- All incoming webhooks are logged to the console for debugging
- Third-party responses are captured and included in the API response
- Error handling covers both incoming request errors and forwarding failures