Junior Backend Engineer - Technical Assessment

Fix & Improve Challenge: Webhook Receiver Service

Overview

You are inheriting a webhook receiver service that works but has several issues. Your job is to **identify problems, prioritize them, fix the most critical ones, and document your decisions**.

This tests your ability to:

- Read and understand existing code.
- Identify technical issues and security concerns.
- Prioritize improvements.

The Scenario

A previous developer built a webhook receiver service for handling incoming webhooks from external services (e.g., payment processors, third-party APIs, etc.). The service is currently running in development, and you have been asked to review and improve it.

The service currently:

- Can receive webhooks via POST endpoint.
- Can store webhook data.
- Can provide endpoints to retrieve webhooks.
- Has multiple issues that make it unsuitable for production.

Your Task

1. Given sample code

```
// types.ts

export interface Webhook {
  id: string;
  source: string;
  event: string;
  payload: any;
  receivedAt: Date;
}

export interface WebhookInput {
  source: string;
  event: string;
  payload: any;
}
```

```ts
// storage.ts

import { Webhook } from './types';

let webhooks: Webhook[] = [];

export const storage = {
  save(webhook: Webhook): void {
    webhooks.push(webhook);
  },

  getAll(): Webhook[] {
    return webhooks;
  },

  getById(id: string): Webhook | undefined {
    return webhooks.find(w => w.id === id);
  },

  count(): number {
    return webhooks.length;
  },

  clear(): void {
    webhooks = [];
  }
};
```

```ts
// main.ts

import express, { Request, Response } from 'express';
import dotenv from 'dotenv';
import { Webhook, WebhookInput } from './types';
import { storage } from './storage';

dotenv.config();

const app = express();
const PORT = process.env.PORT || 3000;

app.use(express.json());

app.post('/webhooks', (req: Request, res: Response) => {
  const input = req.body as WebhookInput;

  const id = Math.random().toString(36).substring(7);

  const webhook: Webhook = {
    id,
    source: input.source,
    event: input.event,
```

```
    payload: input.payload,
    receivedAt: new Date()
  };

  storage.save(webhook);

  res.json({
    id: webhook.id,
    message: 'Webhook received'
  });
});

app.get('/webhooks', (req: Request, res: Response) => {
  const allWebhooks = storage.getAll();

  res.json({
    webhooks: allWebhooks,
    count: allWebhooks.length
  });
});

app.get('/webhooks/:id', (req: Request, res: Response) => {
  const { id } = req.params;

  const webhook = storage.getById(id);

  if (!webhook) {
    return res.status(404).json({ error: 'Webhook not found' });
  }

  res.json(webhook);
});

app.use((err: Error, req: Request, res: Response, next: any) => {
  console.log('Error:', err);
  res.status(500).json({ error: 'Something went wrong' });
});

app.listen(PORT, () => {
  console.log(`Webhook service running on port ${PORT}`);
});
```

Available endpoints:

- `POST /webhooks` - Receive and store a webhook.
- `GET /webhooks` - Retrieve all stored webhooks.
- `GET /webhooks/:id` - Retrieve a webhook by ID.

2. Code Review & Implementation

Review the provided codebase and identify issues. Think about how you would identify and prioritize problems in a real-world scenario. Finally, implement fixes for the top issues you find.

Write a brief analysis document in the `ANALYSIS.md` file that:

1. Lists all issues you found.
2. Categorizes them (security, scalability, reliability, code quality, etc.).
3. Rates their severity (Critical, High, Medium, Low).

Requirements & Constraints

Technical Constraints

- Must use TypeScript.
- Can choose any NodeJS framework (Express, NestJS, Fastify, etc.) to implement your code.
- Can add new dependencies if needed for your fixes.
- Can refactor code structure if it helps your fixes.

Plus Points

- Use NestJS framework.
- Apply design patterns where appropriate.
- Implement unit tests.

Submission Requirements

- Please create a Git repository (e.g., GitHub, GitLab, or Bitbucket) for your source code and share the link with us.
- Have a well-documented `README.md` explaining how to run your code and the changes you made.
- **If your codebase is not running properly, your assessment will be marked as failed.**