CyberXplore Developer Challenge

Project Title: Secure File Upload and Malware Scanning Dashboard

CyberXplore is building a scalable backend system to allow users to upload documents and simulate malware scanning asynchronously in the background. Your task is to create a **full-stack application** that allows file uploads, queues scanning jobs, and displays results in a real-time dashboard.

Objective

Design and implement a system that:

- Accepts PDF/DOCX/Image file uploads from users.
- Stores the uploaded file metadata.
- Simulates malware scanning in the background via a job queue.
- Displays scanned results on a responsive web dashboard.
- Optionally notifies security channels (via Slack/webhook) if a file is malicious.

Tech Stack

You're free to use any frameworks/tools you're comfortable with, but the following is recommended:

- **Backend**: Node.js (TypeScript preferred), Express.js
- **Frontend**: React or Next.js
- **Database**: MongoDB
- **Queue (Required)**: RabbitMQ or custom in-memory queue
- **Optional**: Redis (for file deduplication or caching)

Backend Requirements

1. `POST /upload` – File Upload API

- Accept file types: `.pdf`, `.docx`, `.jpg`, `.png` (Max 5MB)
- Store the file in a local `uploads/` directory or simulate cloud storage.
- Save metadata in MongoDB:

```
{
  "filename": "invoice.pdf",
  "path": "/uploads/invoice.pdf",
  "status": "pending",
  "uploadedAt": "ISO timestamp",
  "scannedAt": null,
  "result": null
}
```

- Enqueue a background job to scan the file.

---

2. Malware Scanning Worker

- Dequeue file scanning jobs from RabbitMQ or a custom queue.
- Simulate malware scanning:
  - Wait 2–5 seconds using `setTimeout()`
  - If the file contains dangerous keywords (e.g., `rm -rf`, `eval`, `bitcoin`) → mark as `infected`
  - Else, mark as `clean`
- Update MongoDB with:
  - `status`: "scanned"
  - `result`: "clean" or "infected"
  - `scannedAt`: ISO timestamp

---

3. `GET /files` – Fetch Scanned Files

- Returns a list of uploaded files with metadata:
  - Filename
  - Status: `pending`, `scanned`
  - Result: `clean`, `infected`
  - Uploaded/Scanned timestamps

---

4. (Optional) Webhook/Slack Alert

- Send a POST webhook or Slack alert when a file is marked as `infected`.

Frontend Requirements

Build a minimal, responsive dashboard (React/Next.js preferred).

Pages

1. File Upload Page

- Drag and drop or file input field
- Show upload progress
- Display message: “Scan in progress…” after upload completes

2. Dashboard Page

- List of all uploaded files:
  - Filename
  - Scan Status: `Pending`, `Scanning`, `Clean`, `Infected`
  - Timestamps
- Auto-refresh every 5–10 seconds to reflect new scan results
- Color indicators:
  - Clean
  - Infected
  - Pending

Bonus Features

- Filter by status (`clean` , `infected` , `pending`)
- View details modal for each file
- Pagination or infinite scroll
- Toast alerts for newly scanned files

Submission Guidelines

- Push your code to a **public GitHub repository**.
- Include a `README.md` with:
  - Setup instructions (backend, frontend, MongoDB, queue)
  - How you simulate scanning
  - How to run frontend + backend locally
- Add sample screenshots or a video if possible.

Hosted Services (Optional)

If you don't want to set up Redis or RabbitMQ locally, you may use:

- **Redis (Free 1-Week Trial)**: https://rclusters.com
- **RabbitMQ (Free Hosted Queue)**: https://www.cloudclusters.io/cloud/rabbitmq/

Grading Criteria (100 Points)

Category	Points
Backend: Upload API, queue logic	20
Worker: Async scanning & DB update	20
Frontend: Upload UI + dashboard	20
MongoDB schema & data handling	15
Code structure, modularity, error handling	10
README clarity & local setup instructions	10
Bonus (Slack, Redis, Webhooks, Filters)	5

What This Assignment Tests

- Full-stack capability (backend, frontend, DB, queues)
- Secure file handling
- Real-world async processing
- UI reactivity with background state changes
- System design and scalability understanding

Ready?

Build something you're proud of. Clean code, thoughtful UX, and well-explained backend logic will stand out.

We’re excited to see what you create. Good luck!