Perfect — let's extend what I did for **Phase 1** to **all phases** so you have a **roadmap + folder structure + required stuff** for every step.
This way, you'll know exactly *what to build per phase* and won't get stuck.

🏗️ Complete Roadmap with Details

✅ Phase 1 – MVP (Basic File Upload + Hash Analyzer)

📂 Folder structure: *(already explained in last message)*

- `frontend/` (React + Clerk login + upload + dashboard)
- `backend/` (Express + Mongo + file upload + SHA256)
- `storage/` (uploaded files)

You build:

- Clerk login flow
- File upload form
- Express `/upload` + `/files` endpoints
- SHA256 hash util
- MongoDB schema for files

👉 End result = Upload a file → backend stores metadata → dashboard shows filename + hash.

✅ Phase 2 – Static Analysis (Prove Cyber Skills)

📂 Extra folders to add:

```
backend/
├── analysis/
```

```
│   ├── static/
│   │   ├── entropy.js
│   │   ├── strings.js
│   │   ├── yaraScan.js
│   │   └── peParser.js
```

You build:

- `entropy.js` → Shannon entropy for file.
- `strings.js` → extract readable ASCII strings → regex for IPs/domains.
- `yaraScan.js` → run YARA rules (via child_process).
- `peParser.js` → parse Windows executables (PE headers).

Backend changes:

- `/analyze/:id` → runs all static analyzers on file → saves results to Mongo.

File schema add:

```
 staticAnalysis: {
  entropy: Number,
  strings: [String],
  iocs: [String],
  yaraMatches: [String],
  peMetadata: Object
}
```

-

Frontend changes:

- Dashboard shows **analysis tab** per file (accordion/table).

👉 End result = After upload, you can click *Analyze* → see entropy, strings, IOCs, YARA matches.

✅ Phase 3 – Threat Intelligence Integration

📂 Extra folders to add:

```
backend/
├── analysis/
│   └── threatIntel/
│       ├── virusTotal.js
│       ├── abuseCh.js
│       └── otx.js
│
```

You build:

- `virusTotal.js` → call VT API by hash (free tier).
- `abuseCh.js` → pull malware domain/IP feeds (CSV/JSON).
- `otx.js` → AlienVault OTX lookup for indicators.

Backend changes:

- `/intel/:id` → look up hash + IOCs in threat intel sources.

File schema add:

```
 threatIntel: {
  vtReport: Object,
  abuseCHMatches: [String],
  otxMatches: [String]
}
```

-

Frontend changes:

- Dashboard shows threat score + intel matches.
- Example: `VirusTotal: 15/70 engines flagged`

👉 End result = File analysis enriched with *global threat intel*.

✅ Phase 4 – Dynamic Analysis (Optional Advanced)

📂 Extra folders to add:

```
backend/
```

```
├── analysis/
│   └── dynamic/
│       └── cuckoo.js
```

You build:

- Setup **Cuckoo Sandbox** (or any open-source sandbox VM).
- `cuckoo.js` → API client to submit file → poll report → parse results.

Backend changes:

- `/dynamic/:id` → submits file → waits for sandbox results → saves process tree, network IOCs, dropped files.

File schema add:

```
 dynamicAnalysis: {
  processes: [String],
  network: [String],
  droppedFiles: [String]
}
```

-

Frontend changes:

- Dashboard adds a **Timeline view** (how malware behaved).

👉 End result = Full behavioral analysis, like professional malware labs.

✅ Phase 5 – Reporting

📂 Extra folders to add:

```
backend/
├── reports/
│   └── reportGenerator.js
```

You build:

- `reportGenerator.js` → use Puppeteer or PDFKit to generate PDF.
- Report includes:
  - File metadata (hash, size, type)
  - Static analysis results
  - Threat intel results
  - Dynamic sandbox results (if available)
  - Final verdict (benign/malicious).

Backend changes:

- `/report/:id` → returns downloadable PDF.

Frontend changes:

- Dashboard “Download Report” button → calls `/report/:id`.

👉 End result = You can deliver a **SOC-style PDF report** to management/HR/interviewers.

🚦 Summary of Phases

Phase	What You Prove	What to Build
1. MVP	Full-stack basics	Upload + hash analyzer
2. Static	Malware RE basics	Entropy, strings, YARA, PE parsing
3. Threat Intel	SOC/Threat hunting	VT, Abuse.ch, OTX lookups
4. Dynamic	Advanced malware RE	Sandbox + behavior logging
5. Reporting	Professionalism	Auto PDF reports

👉 If you stop at **Phase 2 or 3**, it’s already *resume-worthy*.
👉 If you finish **Phase 5**, it’s **industry-grade** and will blow interviewers away.

Do you want me to now **write a combined “final folder structure” tree** (all phases included together) so you know how the repo will look when it’s 100% complete?