**💡 TIXIFY – The Ultimate Ticket Fraud Detection Platform**

**🏷️ What is TIXIFY?**

TIXIFY is an **AI-powered, cloud-based ticket verification system** that ensures event tickets (concerts, sports, festivals, etc.) are **legitimate**, **safe**, and **valid**.

It checks:  
✅ If the ticket is original.  
✅ If the ticket is bought from authorized sellers (no shady third-party websites).  
✅ If the ticket is edited/tampered.  
✅ If the seller is genuine.  
✅ If the website link where you are purchasing is authentic.

**🎯 Why are we building this?**

The world is getting scammed with fake tickets:  
❌ Counterfeit digital tickets.  
❌ Unauthorized resales.  
❌ Edited QR codes and barcodes.  
❌ Overpriced third-party sites.

People lose money, can't enter events, and trust is broken.  
**Tixify exists to stop all this.**

**🏗️ How does TIXIFY work?**

**Step-by-step Flow:**

**User uploads a ticket** (PDF, image, QR code, barcode).  
Tixify scans the ticket and checks:

* Metadata (dates, serial numbers).
* QR code validity.
* Signature of authorized issuers.
* Source of purchase (we check the domain or seller info).

**AI fraud detection** model analyzes:

* Common tampering patterns.
* Invalid modifications.
* Pricing anomalies.

Shows the result:

* ✅ Genuine and Valid.
* ❌ Fake, Edited, or Unauthorized.

(Optional) Save the check history for future reports.

**Bonus**: Real-time warnings when a ticket website is unauthorized.

**🚀 Architecture + Tech Stack:**

| **Layer** | **Tech** | **Description** |
| --- | --- | --- |
| 🧠 Backend | Python (FastAPI) | Build APIs to handle ticket uploads, verification, and results |
| 🎨 Frontend | React (optional) | Simple upload page + result page |
| 🐳 Containerization | Docker | To run backend as a container |
| ☸️ Orchestration | Kubernetes (AKS) | To scale and manage our app |
| ☁️ Cloud | Azure | Hosting and all backend resources |
| 📦 Database | Azure Cosmos DB/Postgres | To store user checks and ticket logs |
| 📂 Storage | Azure Blob Storage | To store uploaded files securely |
| ⚙️ Infrastructure | Terraform | To deploy all infra via code |
| 🔄 CI/CD | GitHub Actions or GitLab CI | Auto-deploy infra and backend |
| 🧠 AI | Python (Sklearn/Tensorflow) | Analyze tickets and detect fraud |

**🏆 Why is TIXIFY special?**

✨ Full end-to-end automation (infra + app + AI).  
✨ Real-world use case.  
✨ Industry-demanding skills: Azure, Kubernetes, Terraform, AI, FastAPI.  
✨ Perfect for resume + interview discussions.  
✨ Shows architecture thinking and execution.

**💼 Target Audience:**

🎟️ Event-goers  
🎟️ Concert lovers  
🎟️ Sports fans  
🎟️ Event organizers wanting fraud checks

**🎁 Future scope:**

* Mobile app.
* Integration with official ticket vendors.
* Blockchain for ticket tracking.
* Real-time market price analysis.
* Automatic alerts for trending scams.

**TIXIFY - FULL ROADMAP**

(aka how we build our tech empire, step by step)

**1. Frontend (User Interface)**

* Tech: **ReactJS** or **Next.js**
* Purpose:
  + Users upload tickets (PDF, images).
  + Check status of ticket validation.
  + Simple, clean dashboard to see results.

**2. Backend (API + Logic)**

* Tech: **Python (FastAPI)** or **Node.js**
* Purpose:
  + Accept uploaded tickets.
  + Call AI model to check ticket authenticity.
  + Talk to database for ticket history.

**3. AI Integration**

* Tech: **OpenAI / AWS Textract / Azure AI / custom ML model**
* Purpose:
  + Extract details from ticket (name, event, date).
  + Verify if it matches genuine patterns.
  + Flag if it's fake, tampered, or third-party.

**4. Database**

* Tech: **PostgreSQL / MongoDB / Azure SQL**
* Purpose:
  + Store uploaded ticket data.
  + Save validation results.
  + Keep user info and logs.

**5. Cloud Infrastructure (where all this runs)**

* Platform: **Azure** (or AWS, depending on your choice)
* Tools:
  + **Terraform** (for Infrastructure as Code).
  + **Kubernetes** (for container orchestration).
  + **Docker** (for containerizing backend + frontend).

**6. CI/CD Pipelines**

* Tool: **GitLab CI/CD**
* Purpose:
  + Automate deployment.
  + Run tests.
  + Push changes live without manual steps.

**7. Monitoring & Logging**

* Tools:
  + **Grafana** + **Prometheus** (metrics).
  + **ELK Stack** (logs).
  + Azure Monitor (if fully on Azure).

**8. Security**

* JWT tokens for user auth.
* API security best practices.
* Input validation and protection.

**9. Deployment**

* Everything runs on **Kubernetes** cluster.
* **Terraform** creates all infra (VMs, DBs, Storage).
* GitHub/GitLab manages source code.

**Deliverables:**

* Full working product.
* Upload, validate, and flag fake tickets.
* Scalable, cloud-native, and secure.
* Resume-ready, LinkedIn-worthy masterpiece.