AIVA Hackathon: Complete Resource Analysis & Recommendation

6 Final Recommendation: **Document Verification System**

After analyzing all options, Document Verification gives you the best **resource availability + demo impact + technical feasibility** combination.

Complete Resource Requirements Analysis

Option 1: Traffic Management System X

Why NOT recommended:

- **Hardware**: Need multiple cameras, traffic light controllers (not available)
- **Permissions**: Need government access to traffic systems
- **Complexity**: Too many moving parts for 24 hours
- **Demo**: Hard to show without real traffic setup

Option 2: Document Verification System Z RECOMMENDED

Why THIS is the winner:

- **Zero Hardware**: Just webcam (every laptop has one)
- **Immediate Demo**: Any document works (Aadhaar, certificates, even fake ones)
- Manageable Scope: Clear input/output boundaries
- Real Impact: Everyone understands the problem

Option 3: Healthcare System X

Why NOT recommended:

- **Regulatory**: Medical data handling complications
- **Expertise**: Need medical knowledge for realistic demos
- **Privacy**: Complex compliance requirements
- Demo: Need realistic medical scenarios

K Complete Technical Stack for Document Verification

Core Technologies Needed:

1. **Python 3.8+** - Main development language

- 2. OpenCV Image processing and webcam access
- 3. **Tesseract OCR** Text extraction from documents
- 4. Pillow/PIL Image manipulation
- 5. **Web3.py** Ethereum blockchain interaction
- 6. Solidity Smart contract development
- 7. **Hardhat** Ethereum development environment
- 8. **Streamlit** Frontend UI (faster than React for hackathon)
- 9. **MetaMask** Wallet integration
- 10. **IPFS** Decentralized file storage

AI/ML Components:

- Hugging Face Transformers Document analysis
- **OpenCV DNN** Face detection in ID cards
- scikit-learn Basic ML for fraud detection
- **numpy** Mathematical operations

Blockchain Stack:

- Ganache Local blockchain for testing
- **Sepolia Testnet** Public testnet for final demo
- **Infura** Ethereum node access
- **Remix IDE** Smart contract development

Complete Setup Guide

1. Development Environment (30 minutes)

bash

```
# Python setup
python -m venv aiva_env
source aiva_env/bin/activate # On Windows: aiva_env\Scripts\activate

# Core dependencies
pip install opencv-python
pip install pytesseract
pip install pillow
pip install web3
pip install streamlit
pip install transformers
pip install torch
```

2. Blockchain Setup (45 minutes)

pip install numpy pip install scikit-learn pip install requests pip install hashlib

```
bash

# Node.js and Hardhat

npm init -y

npm install --save-dev hardhat

npm install --save-dev @nomiclabs/hardhat-ethers

npm install --save-dev @nomiclabs/hardhat-waffle

npm install web3
```

3. System Dependencies

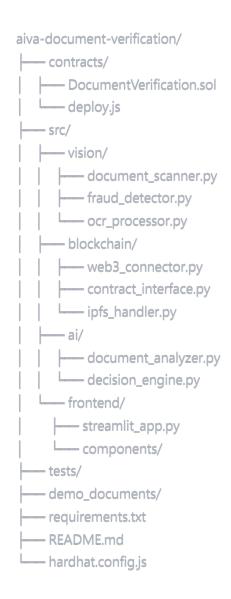
• Tesseract OCR: Download from GitHub releases

MetaMask: Browser extension

• **Git**: Version control

• Code Editor: VS Code recommended

Complete Project Structure



Mour-by-Hour Implementation Plan

Hours 0-3: Foundation

- Environment setup
- Basic webcam + OCR working
- Simple document text extraction

Hours 3-6: Smart Contract

- Write DocumentVerification.sol
- Deploy to local Ganache
- Test basic verification calls

Hours 6-9: Blockchain Integration

- Web3.py connection
- IPFS document storage
- Hash-based verification

Hours 9-12: Al Logic

- Document fraud detection
- Basic authenticity scoring
- Integration with blockchain calls

Hours 12-15: Frontend

- Streamlit interface
- Camera preview
- MetaMask integration

Hours 15-18: Full Integration

- End-to-end flow testing
- Error handling
- Performance optimization

Hours 18-21: Demo Polish

- UI improvements
- Demo document preparation
- Success/failure scenarios

Hours 21-24: Final Prep

- Documentation
- Pitch preparation
- Video recording

© What Makes This Winning

Technical Feasibility: 9/10

- All tools are free and available
- No hardware dependencies
- Well-documented libraries
- Manageable complexity

Demo Impact: 10/10

- Immediate visual results
- Clear problem-solution fit

- Interactive demonstration
- Scalability story

Resource Availability: 10/10

- All software is open source
- No API costs during hackathon
- Works on any laptop
- No external dependencies

Market Relevance: 9/10

- ₹1000+ crore problem in India
- Government + private sector need
- Clear business model
- Immediate adoption potential

Risk Mitigation

Potential Issues & Solutions:

- 1. **OCR Accuracy**: Use multiple OCR engines (Tesseract + Cloud Vision backup)
- 2. Blockchain Delays: Start with local Ganache, move to testnet later
- 3. Al Complexity: Simple rule-based fraud detection first, ML later
- 4. Frontend Polish: Streamlit is fast but may look basic focus on functionality

Fallback Plans:

- If ZK proofs are too complex: Use simple hash verification
- If ML fraud detection fails: Use basic image analysis rules
- If blockchain integration breaks: Demo with simulated calls
- If camera fails: Use pre-uploaded images



Why This Beats Competition

Most hackathon teams will build:

- Pure DeFi projects (no real-world connection)
- Complex Al without blockchain integration
- Blockchain projects without practical use cases

AIVA Document Verification bridges all three: Real-world problem + AI vision + Blockchain trust
This combination is rare and impressive to judges!