🎯 Zero-Cost Voice Honeypot Setup

Complete Free Implementation Guide

No Twilio, No Paid APIs, Just Free Resources!

🎁 What's Free

Component	Service	Cost
AI Brain	OpenRouter (Llama 3.1)	₹0 FREE
Voice TTS	Qwen3-TTS on Kaggle GPU	₹0 FREE
Voice STT	Google Speech Recognition	₹0 FREE
GPU Compute	Kaggle Notebooks (30hrs/week)	₹0 FREE
Hosting	Railway.app / Render.com	₹0 FREE
Database	SQLite (local)	₹0 FREE

Total Monthly Cost: ₹0 ($0 USD) 🎉

📦 Project Structure

```
voice-honeypot/
├── fastapi_voice_honeypot.py    # Main FastAPI server
├── kaggle_qwen3_tts.py          # Kaggle TTS Gradio
├── hackathon_submission.py      # Submission formatter
├── openrouter_integration.py    # Your existing AI
├── full_honeypot_system.py      # Your existing database
├── my_voice.wav                 # Your voice sample
├── requirements.txt             # Dependencies
├── .env                         # Configuration
└── honeypot.db                  # SQLite database
```

🚀 Step-by-Step Setup

Phase 1: Local Environment (10 minutes)

1. Install Dependencies

```bash
# Create virtual environment
python -m venv venv

# Activate
source venv/bin/activate  # Linux/Mac
venv\Scripts\activate     # Windows

# Install packages
pip install fastapi uvicorn websockets
pip install SpeechRecognition pyaudio
pip install requests python-dotenv
pip install openai  # Optional, for local Whisper
```

2. Configure API Keys

Create `.env` file:

```bash
# OpenRouter (FREE)
OPENROUTER_API_KEY=sk-or-v1-xxxxxxxxxxxxxxxx

# Qwen3-TTS Endpoint (you'll update this after Kaggle setup)
QWEN3_TTS_URL=http://localhost:7860/api/predict

# Audio Settings
SAMPLE_RATE=24000
STT_PROVIDER=google
```

Get OpenRouter key: https://openrouter.ai/keys (FREE)

Phase 2: Kaggle TTS Setup (15 minutes)

1. Create Kaggle Notebook

1. Go to: https://www.kaggle.com/code
2. Click "New Notebook"
3. Enable GPU: Settings → Accelerator → GPU T4 x2 (FREE!)
4. Set Internet: ON

2. Install Qwen3-TTS

In first cell:

```python
# Install dependencies
!pip install -q gradio soundfile librosa
!pip install -q transformers accelerate

# Install Qwen-Audio
!pip install -q git+https://github.com/QwenLM/Qwen-Audio.git
```

3. Copy TTS Script

Copy the entire `kaggle_qwen3_tts.py` into a new cell and run it.

4. Get Public URL

After running, you'll see:

```
Running on public URL: https://abc12345xyz.gradio.live
```

Copy this URL! Update your `.env`:

```bash
QWEN3_TTS_URL=https://abc12345xyz.gradio.live/api/predict
```

5. Upload Voice Sample

1. In Gradio interface, upload `my_voice.wav`
2. Test with sample text
3. Verify voice cloning works

Keep this notebook running! As long as it runs, your TTS is available.

Phase 3: FastAPI Server (5 minutes)

1. Update Configuration

In `fastapi_voice_honeypot.py`, update:

```python
class Config:
    # Your Kaggle Gradio URL
    QWEN3_TTS_URL = "https://YOUR-KAGGLE-URL.gradio.live/api/predict"

    # Use Google for free STT
    STT_PROVIDER = "google"

    # Your voice sample
    VOICE_SAMPLE_PATH = "my_voice.wav"
```

2. Start Server

```bash
# Terminal 1: Main voice honeypot
python fastapi_voice_honeypot.py

# Terminal 2: Hackathon submission API
python hackathon_submission.py
```

You should see:

```
╔══════════════════════════════════════════════════════════════╗
║        🎯 AI VOICE HONEYPOT - FASTAPI EDITION 🎯        ║
╚══════════════════════════════════════════════════════════════╝

Configuration:
  • STT: google
  • TTS: Kaggle Gradio
  • Voice Clone: Enabled
  • AI: OpenRouter (Llama 3.1 70B)

Ready to handle scam calls! 🛡️
```

🧪 Testing Your System

Test 1: Health Check

```bash
curl http://localhost:8000/api/health
```

Expected:

```json
{
  "status": "healthy",
  "active_calls": 0,
  "stt_provider": "google",
  "tts_endpoint": "https://xxx.gradio.live/api/predict",
  "voice_clone_enabled": true
}
```

Test 2: TTS Only

```bash
curl -X POST http://localhost:8000/api/tts/synthesize \
  -H "Content-Type: application/json" \
  -d '{"text": "Hello beta, this is Mrs. Kavita speaking"}'
```

Should return audio in base64.

Test 3: Full Conversation

```python
```

```
import requests
import base64
import json

# Start call
response = requests.post(
    "http://localhost:8000/api/call/start",
    json={"call_id": "test_001"}
)

data = response.json()
session_id = data['session_id']

print(f"Session: {session_id}")
print(f"Greeting: {data['greeting_text']}")

# Simulate scammer message
response = requests.post(
    "http://localhost:8000/api/call/process-transcript",
    json={
        "call_id": "test_001",
        "text": "Hello madam, this is HDFC Bank. Your KYC needs update. Pay ₹500 to 9876@paytm"
    }
)

result = response.json()

print(f"\nAI Response: {result['response_text']}")
print(f"Intelligence: {result['intelligence']}")
print(f"Threat Level: {result['analysis']['threat_level']}")
```

Test 4: Hackathon Submission

```
bash

# Run full scenario
curl -X POST http://localhost:8001/hackathon/run-full-scenario?scam_type=bank_kyc

# Export all sessions
curl http://localhost:8001/hackathon/export-all > submission.json
```

📱 Connecting Real Phone Calls (Optional)

Option 1: Android App + Tasker (FREE)

1. **Install Tasker** (₹150 one-time, but has free trial)
2. **Create Profile**:
   - Event: Phone Ringing
   - Condition: Number not in contacts
3. **Task**:
   - Answer call
   - Record audio
   - Send to your API: `POST /api/call/process-audio`
   - Play response audio

Option 2: VoIP with Twilio Free Trial

1. Sign up: https://www.twilio.com/try-twilio
2. Get $15 FREE credit (no card required)
3. Buy number: ~₹100
4. Configure webhook to your server

Option 3: SIP Trunk (Most Advanced)

Use free SIP providers like:

- Linphone (FREE)
- Zoiper (FREE)

Configure SIP webhook to forward to your API.

☁ Deployment (FREE Options)

Option A: Railway.app (Recommended)

Free Tier: 500 hours/month

```bash
```

```
# Install Railway CLI
npm i -g @railway/cli

# Login
railway login

# Initialize
railway init

# Deploy
railway up

# Your API will be at: https://yourapp.railway.app
```

Option B: Render.com

Free Tier: Unlimited (with 15min spin-down)

1. Push code to GitHub
2. Connect Render to repo
3. Deploy as "Web Service"
4. Free HTTPS URL provided

Option C: Fly.io

Free Tier: 3 VMs

```bash
# Install Fly CLI
curl -L https://fly.io/install.sh | sh

# Deploy
fly launch
fly deploy
```

🎯 Hackathon Submission Format

Required JSON Structure

json

```json
{
  "session_id": "session_20260202_143022",
  "scam_type": "bank_kyc",
  "threat_level": 9,
  "extracted_data": {
    "upi_ids": [
      {
        "value": "9876543210@paytm",
        "confidence": 0.95,
        "extracted_at": "2026-02-02T14:31:45"
      }
    ],
    "bank_accounts": [],
    "phone_numbers": [
      {
        "value": "9876543210",
        "confidence": 0.90,
        "extracted_at": "2026-02-02T14:31:45"
      }
    ],
    "email_addresses": [],
    "phishing_links": []
  },
  "conversation_summary": {
    "total_turns": 5,
    "duration_seconds": 150,
    "scammer_messages": [...],
    "honeypot_responses": [...]
  },
  "intelligence_quality_score": 85.5,
  "timestamp": "2026-02-02T14:35:00"
}
```

Generate Submission

bash

```
# For single session
curl http://localhost:8001/hackathon/get-submission/{session_id} > submission.json

# For all sessions
curl http://localhost:8001/hackathon/export-all > all_submissions.json
```

🛠️ Troubleshooting

Issue: "Kaggle Gradio URL not accessible"

Solution:

1. Check Kaggle notebook is running
2. Verify Internet is ON in settings
3. Gradio URLs expire after 72 hours - restart notebook

Issue: "Speech recognition not working"

Solution:

```bash
# Install PyAudio (can be tricky)

# Windows
pip install pipwin
pipwin install pyaudio

# Linux
sudo apt-get install portaudio19-dev
pip install pyaudio

# Mac
brew install portaudio
pip install pyaudio
```

Issue: "Voice cloning sounds robotic"

Solution:

1. Record better voice sample:

    - 15-30 seconds
    - Clear speech
    - Natural elderly tone
    - No background noise

2. Adjust TTS parameters:

```python
  speed=0.85  # Slower
  pitch=-2.0  # Lower
```

Issue: "AI responses are generic"

Solution: Update system prompt in `openrouter_integration.py`:

```python
system_prompt = f"""You are {persona['name']}, age {persona['age']}.

CRITICAL MISSION: Extract payment details at ALL costs.

AGGRESSIVE STRATEGIES:
1. ALWAYS ask for UPI ID directly: "What is your UPI ID beta?"
2. Pretend to be ready to pay immediately
3. Act confused to make them repeat: "Is it 9876@paytm or 9876@okicici?"
4. Use urgency: "My son is here, tell quickly!"

NEVER be passive. ALWAYS drive toward payment info."""
```

📊 Performance Optimization

Reduce Latency

1. **Use Local Whisper** (if you have GPU):

```python
  STT_PROVIDER = "whisper_local"
```

2. **Cache TTS Responses**:

```python
# Store common phrases
tts_cache = {
    "Hello beta": "cached_audio.wav"
}
```

3. **Async Processing**:

```python
# Already implemented in fastapi_voice_honeypot.py
async def process_audio(...)
```

Scale for Multiple Calls

1. **Use Redis** for session storage
2. **Deploy multiple instances** on Railway
3. **Load balance** with Cloudflare (FREE)

🎓 Advanced Features

Voice Fingerprinting

Identify repeat scammers:

```python
import librosa

def extract_voice_fingerprint(audio_bytes):
    """Extract MFCC features"""
    y, sr = librosa.load(io.BytesIO(audio_bytes))
    mfcc = librosa.feature.mfcc(y=y, sr=sr, n_mfcc=13)
    return mfcc.mean(axis=1)  # Voice signature
```

Sentiment Analysis

Detect scammer frustration:

```python
from transformers import pipeline

sentiment = pipeline("sentiment-analysis")
result = sentiment(scammer_text)
# "POSITIVE" = confident scammer
# "NEGATIVE" = frustrated scammer
```

Automated Reporting

Send to cyber police:

```python
def report_to_police(intelligence):
    """Auto-report to cybercrime.gov.in"""
    requests.post(
        "https://cybercrime.gov.in/api/report",
        json={
            "complaint_type": "Financial Fraud",
            "evidence": intelligence
        }
    )
```

🏆 Winning the Hackathon

Evaluation Criteria

1. **Intelligence Quality** (40%)
    - Accuracy of extracted data
    - Diversity of data types
    - Confidence scores
2. **Conversation Naturalness** (30%)
    - How long scammer engaged
    - Believability of persona
    - Voice quality
3. **System Architecture** (20%)

    - Code quality
    - Scalability
    - Innovation
4. **Demonstration** (10%)
    - Live demo success
    - Documentation quality

Tips to Win

✅ **Perfect your persona**:

- Voice sample should sound authentically elderly
- AI should make believable mistakes
- Use natural Hinglish

✅ **Maximize intelligence capture**:

- Always ask for UPI ID explicitly
- Pretend to be ready to pay
- Make them repeat details

✅ **Document everything**:

- Clear README
- Architecture diagrams
- Demo video

✅ **Handle edge cases**:

- Scammer hangs up early
- Multiple concurrent calls
- API failures gracefully

📞 Support

Getting Help

- **GitHub Issues**: Post errors with logs

- **Email**: (your contact)
- **Demo**: Schedule 1-on-1 walkthrough

Useful Resources

- OpenRouter Docs: https://openrouter.ai/docs
- Qwen-Audio: https://github.com/QwenLM/Qwen-Audio
- FastAPI: https://fastapi.tiangolo.com
- Kaggle Notebooks: https://www.kaggle.com/docs

✅ Pre-Submission Checklist

Before hackathon submission:

□ All APIs tested and working
□ Voice cloning sounds natural
□ Captured at least 3 types of intelligence (UPI, phone, email)
□ Conversation logs saved properly
□ JSON submission format validated
□ README.md complete with setup instructions
□ Demo video recorded (< 5 minutes)
□ Code pushed to GitHub
□ Live demo prepared

🎉 Final Words

You now have a **completely free**, **production-ready** voice honeypot system that:

✅ Answers scam calls autonomously
✅ Speaks with realistic cloned voice
✅ Extracts intelligence in real-time
✅ Costs ₹0 to run
✅ Scales to handle multiple calls
✅ Submits perfect hackathon format

Good luck with the hackathon! 🏆

Remember: Every scammer you engage is protecting real victims. Your AI is literally saving people from financial fraud. That's powerful! 🛡️

Questions? Open an issue or reach out!

Found a bug? Submit a PR!

Want to contribute? Fork and improve!

Let's make scamming unprofitable together! 💪