VAANINET – COMPLETE PROTOTYPE & CODING DOCUMENT

Team ABS

---

1. PROJECT OVERVIEW

VaaniNet is an AI-powered multilingual conversational announcement delivery system for public safety and transport hubs. It converts any announcement into multiple languages, pushes notifications to user phones, gives exact actionable details, and supports conversational AI queries using Agora LLMs.

---

2. PROTOTYPE DESIGN (A–Z)

• Instant announcement capture (text/audio)

• Whisper STT for audio ingestion

• Translation to 100+ languages

• TTS audio output for each language

• User app with language preference

• Push notifications for announcements

• Conversational clarification (e.g., “What time is my train now?”)

• Train delay explanation + compensation rules

• Crowd-loss avoidance alerts

• Web admin dashboard

• Mobile app for end users

---

3. SYSTEM FLOW

---

1. Admin makes an announcement

2. STT converts voice → text

3. Text translated to all languages

4. TTS generates audio outputs

5. Users receive notification in their chosen language

6. User asks chatbot follow-ups

7. Agora Conversational AI answers with natural dialogue

---

4. TECH STACK

• Backend: FastAPI

• AI Models: Whisper, Googletrans, gTTS

• Conversational AI: Agora LLMs (gpt-4o)

• Database: MongoDB

• Notifications: Firebase Cloud Messaging

• Frontend: Flutter / React

• Deployment: Docker + Render/Heroku

---

5. COMPLETE CODING (A–Z)

--- BACKEND API (FASTAPI) ---

```
from fastapi import FastAPI
from pydantic import BaseModel
import googletrans
from gtts import gTTS

app = FastAPI()

class Announcement(BaseModel):
```

```
text: str
lang: str

@app.post("/translate")
def translate_text(data: Announcement):
translator = googletrans.Translator()
result = translator.translate(data.text, dest=data.lang)
return {"translated_text": result.text}

--- SPEECH TO TEXT (WHISPER) ---
import whisper
model = whisper.load_model("small")
result = model.transcribe("announcement.wav")
print(result["text"])

--- TEXT TO SPEECH ---
from gtts import gTTS
def generate_audio(text, lang):
tts = gTTS(text=text, lang=lang)
tts.save("output.mp3")

--- AGORA CONVERSATIONAL AI ---
import agora_ai
client = agora_ai.Client(api_key="YOUR_KEY")
response = client.chat.complete(
model="gpt-4o",
messages=[{"role": "user", "content": "Train 1234 delay"}]
)
print(response.text)

--- PUSH NOTIFICATIONS ---
```

```
from firebase_admin import messaging
def send_push(to, message):
note = messaging.Message(
notification=messaging.Notification(
title="Public Announcement",
body=message
),
token=to
)
messaging.send(note)

--- DATABASE (MONGODB) ---
from pymongo import MongoClient
client = MongoClient("mongodb://localhost:27017/")
db = client.vaaninet
db.announcements.insert_one({"msg": "Train delayed", "time": "2025"})
```

---

6. USER APP WORKFLOW

• Select language

• Receive alerts instantly

• Ask conversational questions like:

“When will the train actually arrive?”

“What compensation will I get?”

“Which platform has changed?”

• Listen to multilingual audio playback

---

7. ADMIN DASHBOARD

• Upload announcements

• See analytics

• Trigger emergency alerts

• Edit translation settings

---

8. DEPLOYMENT OUTLINE

---

1. Dockerize backend

2. Push to cloud (Render)

3. Connect Firebase for notifications

4. Integrate Agora AI key

5. Publish app on Play Store

END OF DOCUMENT
