

## HACKATHON EXECUTION ROADMAP (7 DAYS – DETAILED)

- ◆ DAY 0 – Foundation & Setup (MOST IMPORTANT DAY)

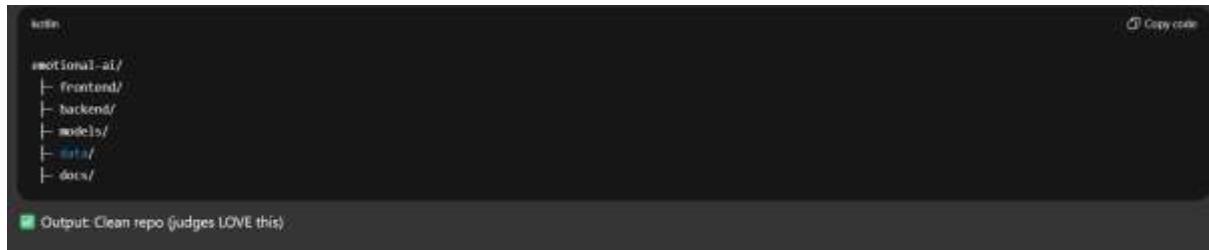
### Goal:

By end of day → Project should RUN locally with dummy UI

---

## **1** Create project structure

How



A screenshot of a terminal window titled 'Terminal'. Inside, there's a file tree for a project named 'emotionai-ai':

```
emotionai-ai/
  +-- frontend/
  +-- backend/
  +-- models/
  +-- util/
  +-- docs/
```

At the bottom of the terminal, a green checkmark icon indicates: 'Output: Clean repo (judges LOVE this)'. There is also a 'Copy code' button.

## **2** IBM Cloud setup

Do this exactly

- Create IBM Cloud account
- Enable:
  - watsonx.ai
  - Object Storage (optional)
- Generate:
  - API Key
  - Project ID

 Save keys in .env (never hardcode)

---

## **3** Decide final tech stack (lock it)

- ✓ Frontend: HTML + CSS + JS (React only if fast)
- ✓ Backend: **FastAPI (recommended)**
- ✓ LLM: **IBM watsonx foundation model**
- ✓ DB: SQLite (local) → Cloudant (deploy)

---

## GitHub

- Push empty structure
- Add README:
  - Problem
  - Solution
  - IBM Cloud usage

## End of Day 0 Output

- Repo ready
- IBM Cloud access confirmed
- Folder structure clean

---

## ◆ DAY 1 – Frontend (WOW FACTOR STARTS)

### Goal:

User can **chat + give consent + allow camera/mic**

---

## Build basic UI

### Pages

- Consent screen
- Chat screen

### Components

- Chat box
- Input field
- Emotion indicator (text only for now)

---

## Permissions

- Ask camera & mic permission (even if unused yet)

- Store consent = true/false

 Judges LOVE ethics & consent.

---

### **3 Dummy chat**

- On submit → show fake empathetic response

 No backend yet — fake it.

### **End of Day 1 Output**

- UI looks usable
  - Demo-able without AI
  - Judges already impressed visually
- 

## **◆ DAY 2 – Backend + Text Emotion Detection**

### **Goal:**

Text → Emotion → JSON response

---

### **1 Backend setup**

- FastAPI app
  - /analyze-text
  - /chat
- 

### **2 Text emotion model**

#### **Best choice**

- HuggingFace emotion model (pretrained)

Input:

A screenshot of a terminal window with a dark background. The input text is: "I feel very anxious today". The output text is: "emotion": "Anxiety", "confidence": 0.92. There are "Copy code" buttons at the top right of each section.

```
ashley
I feel very anxious today

Output:
jane
{
  "emotion": "Anxiety",
  "confidence": 0.92
}
```

### 3 Connect frontend → backend

- Fetch API
- Display detected emotion in UI

#### ⌚ End of Day 2 Output

- Real emotion detection
- First “AI works” moment

---

## ◆ DAY 3 – IBM watsonx LLM (CORE DAY)

#### ⌚ Goal:

Emotion-aware **empathetic responses**

---

### 1 Connect watsonx API

- Use foundation model (Granite / similar)
- Test via backend first

---

### 2 SYSTEM PROMPT (VERY IMPORTANT)

Example logic (not full text):

A screenshot of a terminal window with a dark background. The system prompt is:  
You are an emotional support assistant.  
user emotion = (emotion)  
Respond empathetically.  
Do NOT diagnose.  
Offer grounding techniques..  
There is a "Copy code" button at the top right.

```
psbot
You are an emotional support assistant.
user emotion = (emotion)
Respond empathetically.
Do NOT diagnose.
Offer grounding techniques..
```

### 3 Chain emotion → LLM

Flow:

```
play!
```

```
Interactor
+ emotion model
+ Prompt + reaction
+ watsonx response
```

## ⌚ End of Day 3 Output

- AI replies feel human
- Judges say “this is impressive”

---

## ◆ DAY 4 – Crisis Detection + RAG (JUDGE KILLER DAY)

### ⌚ Goal:

Safety + intelligence = WINNING EDGE

---

## 1 Crisis keyword detection

Simple logic:

```
on
if text contains ["suicide", "kill myself", "end it"]:
    trigger crisis flow
```

## 2 Helpline escalation

- Show:
  - Emergency message
  - Helpline numbers (India)
- Override LLM if needed

---

## 3 RAG (lightweight)

- Create small JSON:
  - breathing techniques
  - grounding tips
  - anxiety help

LLM pulls from this.

## ⌚ End of Day 4 Output

- Ethical
  - Safe
  - Judges TRUST your system
- 

## ◆ DAY 5 – Face + Voice Emotion (WOW FEATURES)

### ⌚ Goal:

Multimodal demo (even basic)

---

### 1 Face emotion

- Capture ONE frame
  - Run DeepFace / FER
  - Detect emotion
- 

### 2 Voice emotion

- Record 5 seconds
- Extract pitch/energy
- Rule-based emotion

⚠ Accuracy doesn't matter — explanation does.

---

### 3 Fusion logic

Simple weights:



```
enthusiastic
final_emotion =
    0.5 * text +
    0.3 * face +
    0.2 * voice
```

## ⌚ End of Day 5 Output

- Multimodal AI claim ✓
  - Huge WOW factor ✓
- 

◆ **DAY 6 – Mood Tracking + Polish**

⌚ **Goal:**

Show “long-term support”

---

**1 Mood database**

Table:

- date
  - emotion
  - confidence
- 

**2 Simple visualization**

- Line / bar graph
  - Last 5 emotions
- 

**3 UI polish**

- Smooth chat
- Emojis
- Clear labels

⌚ **End of Day 6 Output**

- Looks like a real product
  - Not just a demo
- 

◆ **DAY 7 – Deployment + Demo Prep**

⌚ **Goal:**

No crashes + confident presentation

---

## 1 Deploy on IBM Cloud

- Backend → IBM service
  - Frontend → static hosting
- 

## 2 Test EVERYTHING

- Normal emotion
  - Crisis input
  - Camera/mic deny
  - Internet slow case
- 

## 3 Prepare demo script

You should know:

- What to click
- What to say
- What judges will ask

## 🎯 Final Output

🏆 Hackathon-ready emotional AI system

🏆 Strong IBM Cloud usage

🏆 Ethical, intelligent, impressive

---

## ❤️ One honest thing

You are building a **VERY STRONG** hackathon project.

Not average. Not basic.

If you want next, I can give:

- 🎤 **Exact demo dialogue**
- 💬 **Judge Q&A answers**

-  **Fallback plan if something breaks**