🛠️ The Tech Stack (The Toolkit)

To achieve "Workspaces" (e.g., Marvel vs. Family) and "Experiments," we need to upgrade our architecture slightly.

1. Frontend (The Experience)

- **React + Vite:** (You already have this).
- **Framer Motion:** *Essential* for the "Smooth animations" and "Moments of delight" you mentioned in the PDF.
- **Recharts:** To build graphs for the "Similarity Score" or "Accuracy Tests."
- **React-Dropzone:** For dragging and dropping multiple images.
- **Zustand:** For managing state (like "Current Workspace").

2. Middleware (The Manager - Node.js)

- **Prisma:** We need a real database now to track which user owns which workspace.
- **Multer/FS:** To manage file uploads into specific folders (e.g., uploads/user_1/workspace_marvel/).

3. The Brain (Python FastAPI)

- **DeepFace:** (Existing)
- **OpenCV:** For drawing the "Green Boxes" in the "Find Me" experiment.
- **Pandas:** To manage data analysis if users want to export results.

🗓️ The Master Work Plan

We will break this into **4 Phases**. Do not try to do everything at once.

Phase 1: The "Workspace" Architecture (Backend Upgrade)

Currently, your system has **one** global database (known_faces). We need to make it dynamic so users can switch between "Game of Thrones" and "Marvel."

- **Step 1.1:** Update Prisma Schema to support Workspace and FaceCharacter.
- **Step 1.2:** Update Python API to accept a workspace_id.
    - *Old:* verify(image) -> loads global.pkl
    - *New:* verify(image, workspace_id) -> loads storage/{workspace_id}/embeddings.pkl
- **Step 1.3:** Create APIs to Create/Delete/Switch Workspaces.

Phase 2: Experiment 1 & 2 (The Analytics)

This focuses on the "Magic Mirror" and "Twin-O-Meter."

- **Step 2.1: "Magic Mirror":** Create a Python endpoint that runs DeepFace.analyze() (Age, Gender, Emotion) and returns JSON.
- **Step 2.2: Frontend UI:** Build a nice card component that flips over to reveal the analysis.
- **Step 2.3: "Twin-O-Meter":** Create a Python endpoint that compares two uploaded images and returns the distance float (0.0 to 1.0).
- **Step 2.4: Frontend UI:** Build a "Gauge" or "Progress Bar" that visualizes that number.

Phase 3: Experiment 3 (Computer Vision)

This is the "Find Me in the Crowd" feature.

- **Step 3.1: Python Logic:** Write a script that detects *all* faces in a group photo.
- **Step 3.2: The Loop:** Compare the "Target Face" against every face found in the group.
- **Step 3.3: Drawing:** Use OpenCV to draw a Green Rectangle on the match and Red Rectangles on non-matches.
- **Step 3.4: Response:** Return the *processed image* to the frontend.

Phase 4: Polish & "Moments of Delight"

This is where we implement the UI tips from your PDF.

- **Step 4.1:** Add Micro-interactions (e.g., Confetti when a face is added).
- **Step 4.2:** Add "Loading States" (e.g., "Scanning neural pathways..." instead of just "Loading").
- **Step 4.3:** Build the "History" tab (Prisma logs) so users can see past experiments.

🧠 Conceptual Change: How "Workspaces" will work

This is the hardest part to visualize, so here is the logic for your backend.

File Structure on your Server:
Instead of one folder, you will have this:

```
/storage
  /user_123
    /workspace_marvel
      /images
        iron_man.jpg
        hulk.jpg
      embeddings.pkl  <-- Only contains Marvel faces

    /workspace_family
      /images
        dad.jpg
        mom.jpg
      embeddings.pkl  <-- Only contains Family faces
```

The Python Logic:
When the React App sends a request, it **must** send the workspace_id.

```python
# Conceptual Python Code
@app.post("/verify")
def verify(file: UploadFile, workspace_id: str):
    # 1. Dynamically load the correct PKL file based on workspace_id
    pkl_path = f"storage/{user_id}/{workspace_id}/embeddings.pkl"

    local_db = pickle.load(open(pkl_path, "rb"))

    # 2. Run recognition against THAT specific database
    result = find_match(file, local_db)
    return result
```

🚀 Immediate Next Step

We cannot build the UI until the Backend supports multiple workspaces.

Do you want to start with Phase 1 (Building the Multi-Workspace Backend System) today?
I can help you design the **Prisma Schema** and the **Folder Logic** first.