MELODY MIND (AI POWERED MUSIC PLAYER)

TEAM 527

MILESTONE 3

TEAM MEMBERS:

Deeshna Shruthi - 23BD1A6785
Safa - 23BD1A67B4
G. Prashanth - 23BD1A6790
G. Abhishek Bhan - 23BD1A6793
K. Sai Teja - 23BD1A6799

DATABASE DESIGN:

Users Table

```sql
CREATE TABLE Users (
    user_id INT PRIMARY KEY AUTO_INCREMENT,
    username VARCHAR(50) UNIQUE NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    password_hash VARCHAR(255) NOT NULL,
    profile_picture BLOB,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    last_login TIMESTAMP,
    preferences JSON
);
```

Songs Table

```sql
CREATE TABLE Songs (
    song_id INT PRIMARY KEY AUTO_INCREMENT,
    title VARCHAR(200) NOT NULL,
    artist VARCHAR(100) NOT NULL,
    album VARCHAR(150),
    duration INT, -- in seconds
    genre VARCHAR(50),
    tempo INT, -- BPM
    file_path VARCHAR(500),
    lyrics TEXT,
    sentiment_score DECIMAL(3,2), -- -1 to 1 scale
    mood_tags JSON, -- ["happy", "energetic", "calm"]
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Playlists Table

```sql
CREATE TABLE Playlists (
    playlist_id INT PRIMARY KEY AUTO_INCREMENT,
    user_id INT,
    name VARCHAR(100) NOT NULL,
    description TEXT,
    is_ai_generated BOOLEAN DEFAULT FALSE,
    mood_context VARCHAR(50),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES Users(user_id)
);
```

Playlist_Songs Table

```sql
CREATE TABLE Playlist_Songs (
    playlist_id INT,
    song_id INT,
    position INT,
    added_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (playlist_id, song_id),
    FOREIGN KEY (playlist_id) REFERENCES Playlists(playlist_id),
    FOREIGN KEY (song_id) REFERENCES Songs(song_id)
);
```

Mood_Sessions Table

```sql
CREATE TABLE Mood_Sessions (
    session_id INT PRIMARY KEY AUTO_INCREMENT,
    user_id INT,
    detected_mood VARCHAR(50),
    detection_method ENUM('facial', 'text', 'voice'),
    input_data TEXT, -- stores text input or file path
    confidence_score DECIMAL(3,2),
    timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES Users(user_id)
);
```

User_Analytics Table

```sql
CREATE TABLE User_Analytics (
    analytics_id INT PRIMARY KEY AUTO_INCREMENT,
    user_id INT,
    song_id INT,
    action ENUM('play', 'pause', 'skip', 'like', 'dislike'),
    listening_duration INT, -- in seconds
    mood_at_play VARCHAR(50),
    timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES Users(user_id),
    FOREIGN KEY (song_id) REFERENCES Songs(song_id)
);
```

Admin Table

```sql
CREATE TABLE Admin (
    admin_id INT PRIMARY KEY AUTO_INCREMENT,
    username VARCHAR(50) UNIQUE NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    password_hash VARCHAR(255) NOT NULL,
    role ENUM('super_admin', 'content_admin', 'analytics_admin'),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    last_login TIMESTAMP
);
```

GAMES

```
CREATE TABLE games (
  id          UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  slug        TEXT UNIQUE NOT NULL,              -- 'tap-the-notes','mood-quiz','breathing
  title       TEXT NOT NULL,
  difficulty  TEXT,                              -- 'easy','medium','hard'
  est_minutes INTEGER,
  benefits    TEXT[],                            -- tags shown in UI
  status      TEXT DEFAULT 'active'              -- 'active'|'coming_soon'
);
```

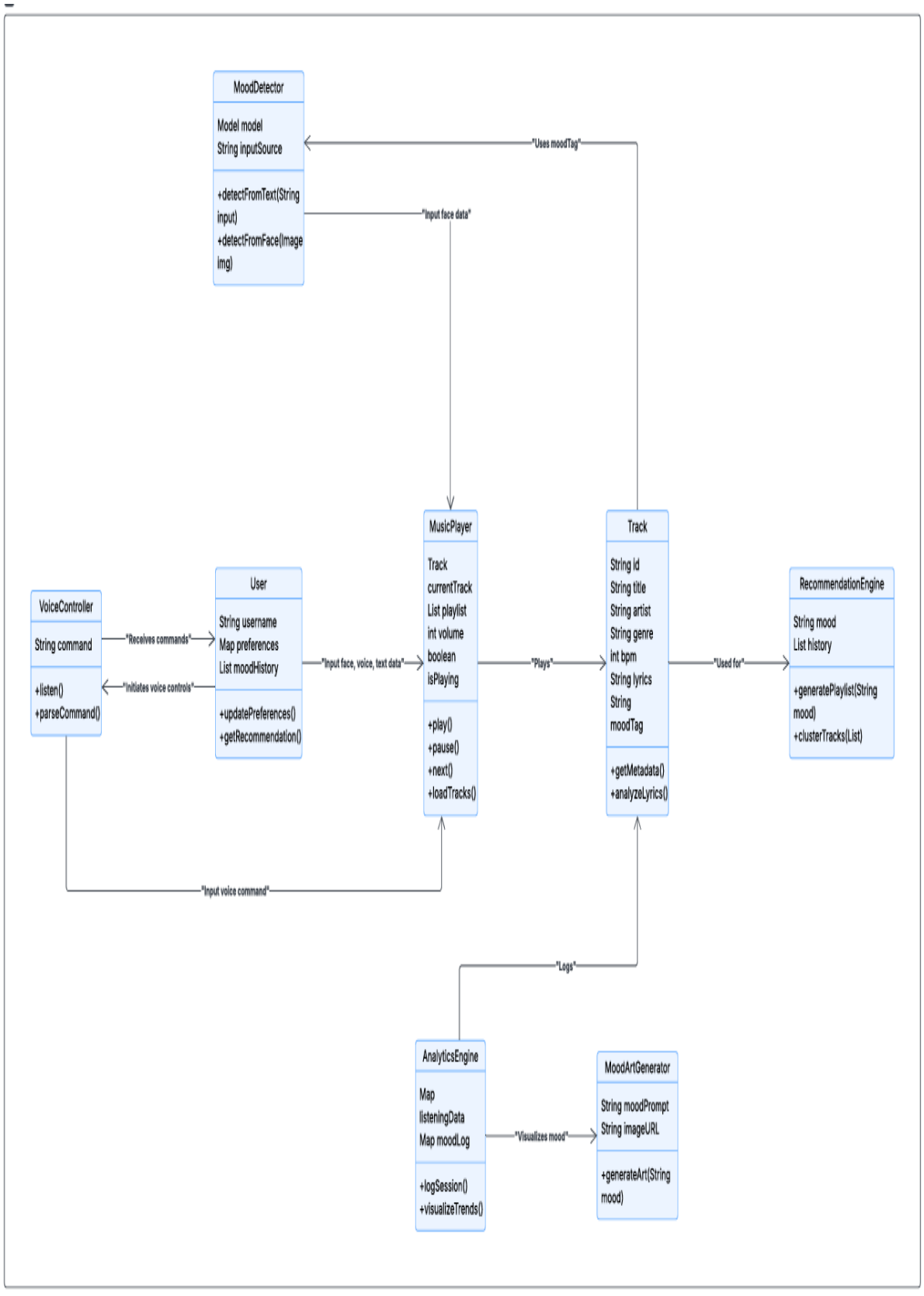
GAME SESSIONS

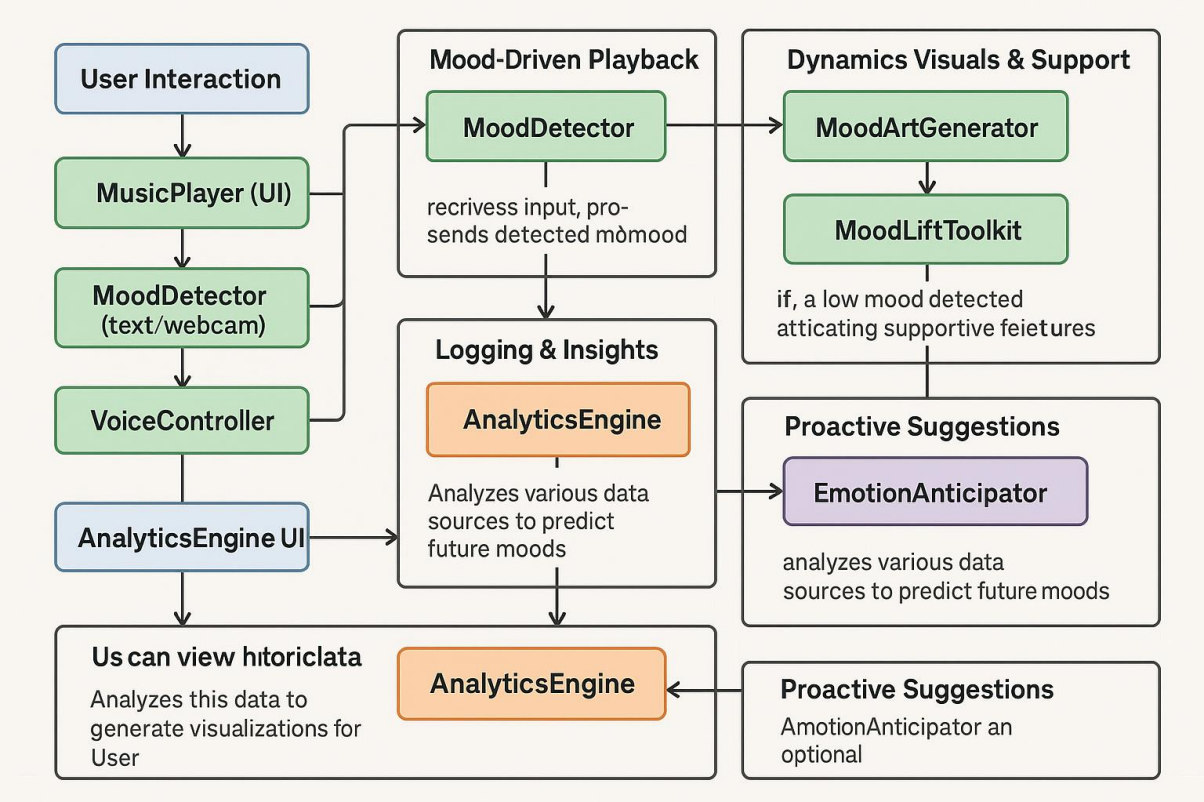
```
CREATE TABLE game_sessions (
  id             UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  user_id        UUID NOT NULL REFERENCES users(id) ON DELETE CASCADE,
  game_id        UUID NOT NULL REFERENCES games(id) ON DELETE CASCADE,
  started_at     TIMESTAMPTZ NOT NULL DEFAULT now(),
  ended_at       TIMESTAMPTZ,
  pre_mood       TEXT,
  post_mood      TEXT,
  pre_intensity  INTEGER,
  post_intensity INTEGER,
  notes          TEXT,
  meta           JSONB DEFAULT '{}'::jsonb        -- e.g., level, mode
);
```

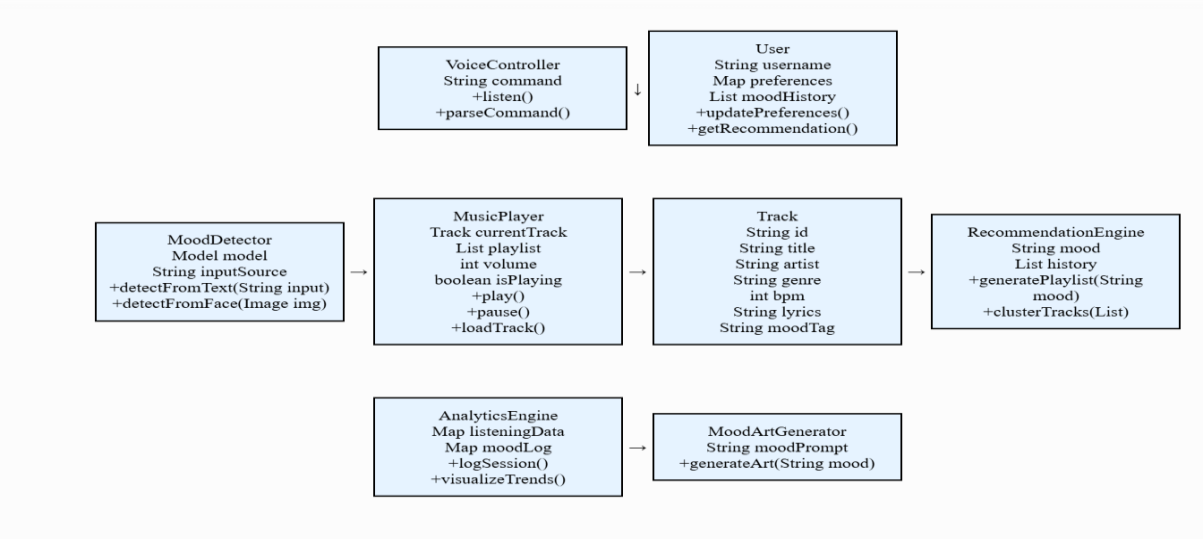
GAME SCORES

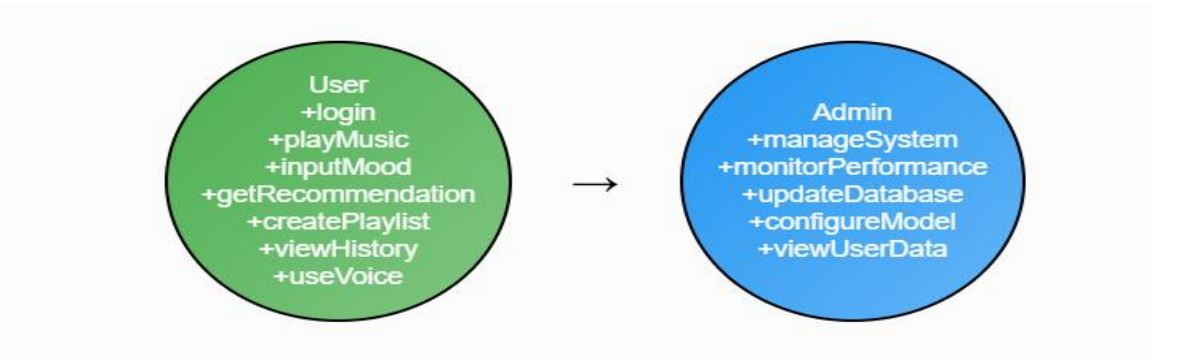
```
CREATE TABLE game_scores (
  id              UUID PRIMARY KEY DEFAULT uuid_generate_v4(),
  game_session_id UUID NOT NULL REFERENCES game_sessions(id) ON DELETE CASCADE,
  score           INTEGER,
  details         JSONB DEFAULT '{}'::jsonb       -- per-round stats, accuracy, streaks
);
```

CLASS DESIGN




User Interaction
MusicPlayer (UI)
MoodDetector (text/webcam)
VoiceController
AnalyticsEngine UI
Mood-Driven Playback
MoodDetector
recrivess input, pro-
sends detected mòmood
Logging & Insights
AnalyticsEngine
Analyzes various data sources to predict future moods
Dynamics Visuals & Support
MoodArtGenerator
MoodLiftToolkit
if, a low mood detected
atticating supportive feietures
Proactive Suggestions
EmotionAnticipator
analyzes various data sources to predict future moods
Us can view hıtoriclata
Analyzes this data to generate visualizations for User
AnalyticsEngine
Proactive Suggestions
AmotionAnticipator an optional


VoiceController
String command
+listen()
+parseCommand()
User
String username
Map preferences
List moodHistory
+updatePreferences()
+getRecommendation()
MoodDetector
Model model
String inputSource
+detectFromText(String input)
+detectFromFace(Image img)
MusicPlayer
Track currentTrack
List playlist
int volume
boolean isPlaying
+play()
+pause()
+loadTrack()
Track
String id
String title
String artist
String genre
int bpm
String lyrics
String moodTag
RecommendationEngine
String mood
List history
+generatePlaylist(String mood)
+clusterTracks(List)
AnalyticsEngine
Map listeningData
Map moodLog
+logSession()
+visualizeTrends()
MoodArtGenerator
String moodPrompt
+generateArt(String mood)

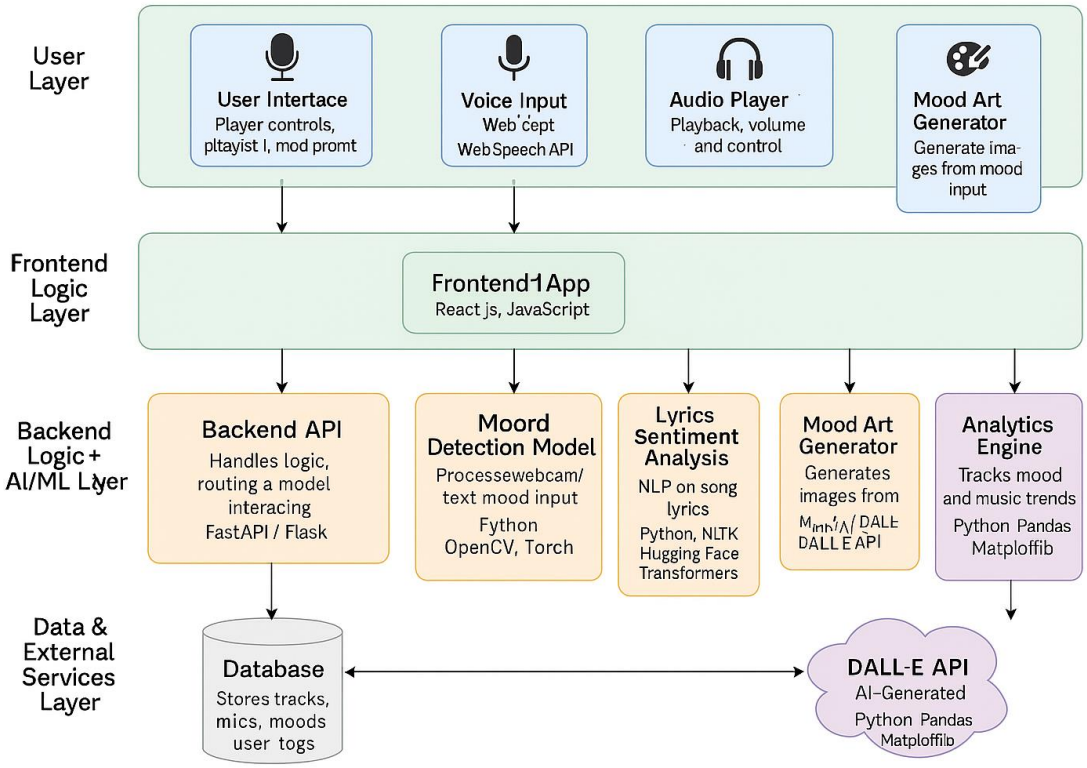

User
+login
+playMusic
+inputMood
+getRecommendation
+createPlaylist
+viewHistory
+useVoice
Admin
+manageSystem
+monitorPerformance
+updateDatabase
+configureModel
+viewUserData

TECHNOLOGY STACK

	Feature	**Functionality**	**Tech Stack**
1. User Layer (Frontend Interface)	User Interface	Play music, browse/search, show playlists	React.js, Material UI
	Mood Input UI	Text box for mood input or trigger webcam	React.js, HTML5 Video
	Voice Command Interface	Capture voice commands (play, pause, etc.)	Web Speech API
2. Frontend Logic Layer	Input Processing	Handle mood input, voice commands, and UI events	JavaScript, React.js
	Face Capture & Preprocessing	Capture and preprocess face data from webcam	TensorFlow.js or OpenCV.js
	Request Handling	Send requests to backend (mood analysis, music search)	Axios / Fetch API
3. Backend Logic + AI/ML Layer	Mood Detection	Detect emotions from facial input or text	Python, OpenCV, DeepFace, Transformers
	Music Recommendation	Suggest music based on mood, lyrics, history	Scikit-learn (for clustering), Pandas, NumPy
	Sentiment Analysis	Analyze lyrics for emotional alignment	HuggingFace Transformers (BERT, RoBERTa)
	Voice Command Processing	Interpret and route voice commands	Python SpeechRecognition
4. Data & External Services Layer	Music & Lyrics DB	Store music metadata, lyrics, playlist data	MongoDB / Firebase / PostgreSQL
	User Data Storage	Store preferences, face data (locally)	SQLite / LocalStorage / IndexedDB

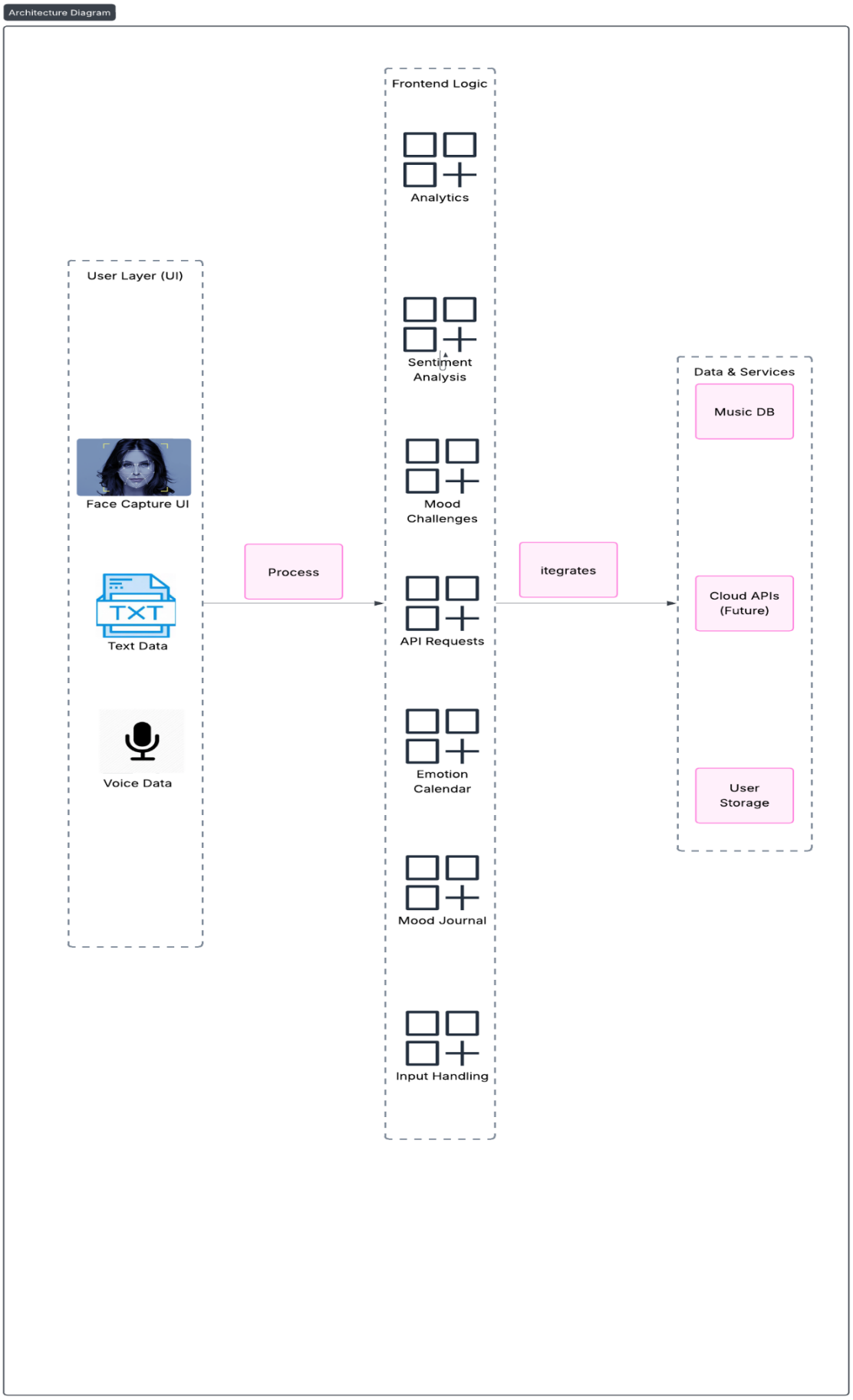
	Analytics Engine	Track usage patterns, generate insights	Matplotlib, Seaborn, Python scripts
	Cloud Integration (Future Scope)	Access external music APIs / cloud libraries	Spotify API, AWS S3 (optional for scalability)
5. Games for Mood Upliftment	Games	Play interactive mood-based games (puzzle, trivia, memory games)	React.js, Canvas API, Python, Scikit-learn

ARCHITECTURE DIAGRAM



Flow of Data:

- The User Layer interacts with the Frontend Logic Layer, which routes requests to the Backend AI/ML Layer.
- The Backend API processes data, utilizing the Mood Detection Model and Lyrics Sentiment Analysis to interpret user input and song lyrics.
- The Mood Art Generator creates images based on mood data, while the Analytics Engine tracks trends.
- Data is stored and retrieved from the Database, and the DALL-E API is used for generating AI-based images.


Architecture Diagram
Frontend Logic
Analytics
User Layer (UI)
Sentiment Analysis
Data & Services
Music DB
Face Capture UI
Mood Challenges
Process
itegrates
TXT
Text Data
API Requests
Cloud APIs (Future)
Emotion Calendar
Voice Data
User Storage
Mood Journal
Input Handling