AMITY UNIVERSITY NOIDA, UTTAR PRADESH

AMITY SCHOOL OF ENGINEERING AND TECHNOLOGY

SOFTEARE PROJECT MANAGEMENT

LAB FILE

Submitted To:

Dr. Renuka Arora

Submitted By:

Ananya Gupta

A2305222350

B.Tech. 7CSE5Y

INDEX

S.No	Lab Experiment	Date	Remarks
1.	To create and explain a Use Case Diagram for the "TheraXbot- A mental health companion" minor project.	17 July 2025	
2.	To create a Work Breakdown Structure (WBS) for the " TheraXbot- A mental health companion " minor project.	24 July 2025	
3.	To use Zoho Projects as a project management tool to track, manage, and report on the development of the " TheraXbot- A mental health companion " minor project	31 July 2025	
4.	To create and explain a Resource Planning Template for the " TheraXbot – a multimodal mental health companion " minor project.	7 August 2025	
5.	To generate a sequence diagram for the " TheraXbot – a multimodal mental health companion " minor project.	14 August 2025	
6.	To generate an activity diagram for the " TheraXbot – a multimodal mental health companion " minor project.	21 August 2025	
7.	To do risk analysis for the " TheraXbot – a multimodal mental health companion " minor project.	28 August 2025	
8.	To generate gantt chart for the " TheraXbot – a multimodal mental health companion " minor project.	4 September 2025	
9.	To generate step-wise planning chart for the " TheraXbot – a multimodal mental health companion " minor project.	11 September 2025	
10.	To evaluate the critical path for the given dataset.	18 September 2025	
11.	To generate pert chart for the " TheraXbot – a multimodal mental health companion " minor project	25 September 2025	

LAB ASSIGNMENT - 1

DATE: 17 July 2025

AIM: To create and explain a Use Case Diagram for the "TheraXbot- A mental health companion" minor project.

THEORY:

UML stands for **Unified Modelling Language**. It's a standardized way to visually represent the design, structure, and behaviour of a software system. UML includes different types of diagrams (like use case, class, sequence, activity diagrams, etc.) to describe what the system should do and how it should work. The main goal is to make communication between developers, designers, and stakeholders easier.

Key Components of a Use Case Diagram:

- **Actors:** An actor represents a role that a user or another system plays when interacting with the system. It is depicted as a stick figure and represents an external entity that communicates with the system. An actor initiates a use case.
- **Use Cases:** A use case represents a set of actions performed by the system to produce a result of value to a specific actor. It is depicted as an oval and describes a single functionality or task that the system provides.
- **System Boundary:** The system boundary is a box that encloses all the use cases, visually representing the scope of the system under consideration. All use cases within the boundary are part of the system.
- **Relationships:**
    - **Association:** The primary relationship between an actor and a use case. It is a line that indicates an actor participates in, or initiates, a use case.
    - **Include (<<include>>):** An include relationship is a dependency between two use cases, indicating that the behavior of the included use case is a mandatory part of the base use case's behavior. The included use case is often a common functionality that is shared by multiple use cases. This relationship is represented by a dashed arrow pointing from the base use case to the included use case, with the <<include>> stereotype.

Purpose of a Use Case Diagram:

A use case diagram provides a high-level overview of the system's functionality. It helps in:

- Visualizing the functional requirements.
- Defining the scope of the project.
- Facilitating communication among stakeholders, developers, and clients.
- Serving as a basis for more detailed design and testing.

Explanation of Use Case Diagram:-

Title: *TheraXBot – a mental health companion*

Actors

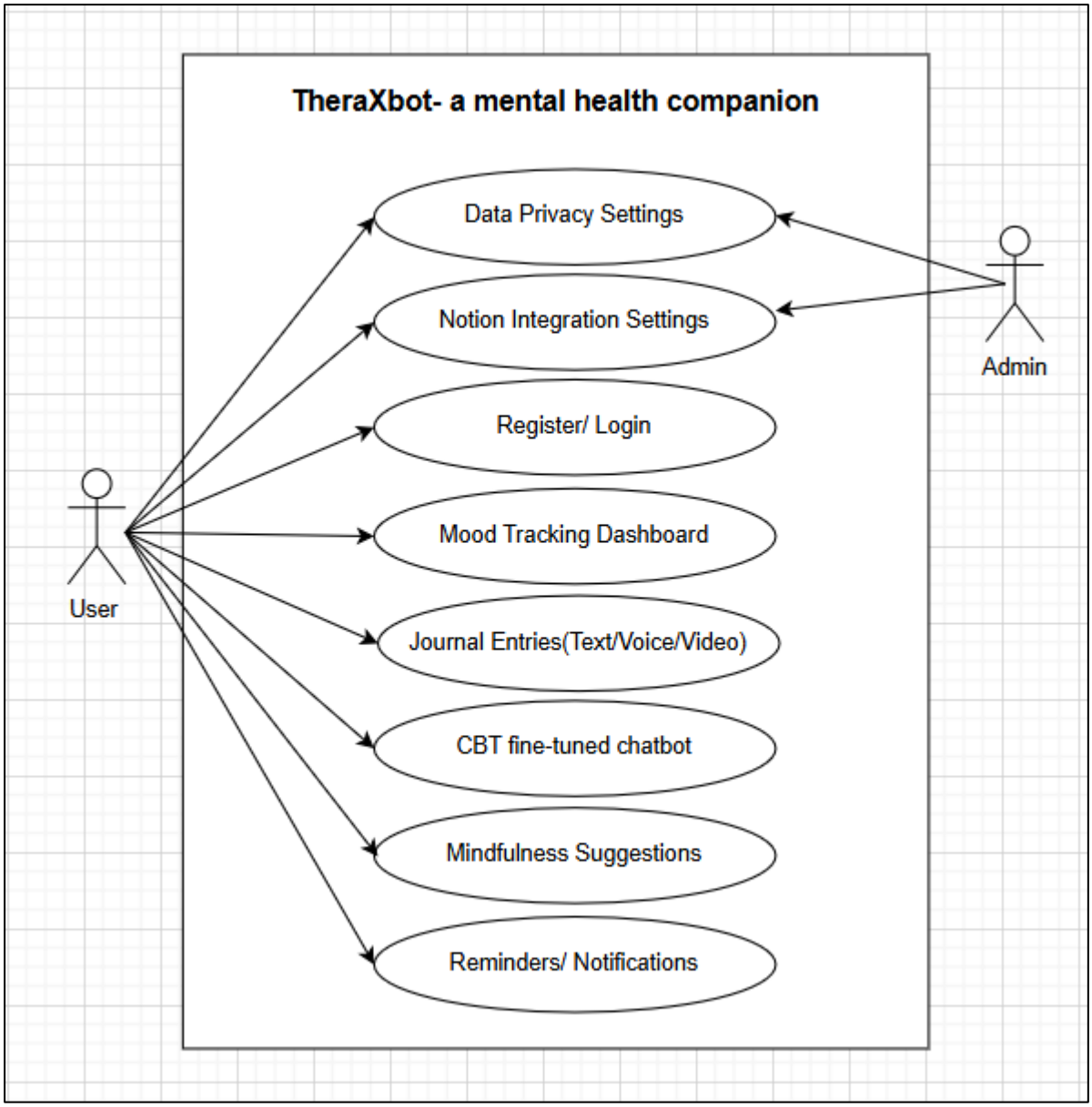
1. User
    - The main actor who uses TheraXBot to manage mental health.
2. Admin
    - A supporting actor responsible for managing privacy and integration settings.

System Boundary

The box labeled *TheraXBot – a mental health companion* contains all the features the system provides.

Use Cases (Functionalities)

1. Register/Login
    - Allows users to create accounts and log in securely.
2. Journal Entries (Text/Voice/Video)
    - Users can input their emotions through text, voice, or video.
3. Mood Tracking Dashboard
    - Displays mood trends and progress over time.
4. CBT Fine-tuned Chatbot
    - Provides therapeutic conversations using Cognitive Behavioral Therapy principles.
5. Mindfulness Suggestions
    - Gives guided meditation, breathing, and relaxation exercises.
6. Reminders/Notifications
    - Sends alerts for journaling and wellness activities.
7. Notion Integration Settings *(User & Admin)*
    - Allows linking the user's journals to a Notion dashboard.
8. Data Privacy Settings *(User & Admin)*
    - Manages permissions for storing and processing personal data.


TheraXbot- a mental health companion
Data Privacy Settings
Notion Integration Settings
Register/ Login
Mood Tracking Dashboard
Journal Entries(Text/Voice/Video)
CBT fine-tuned chatbot
Mindfulness Suggestions
Reminders/ Notifications
User
Admin

LAB ASSIGNMENT – 2

DATE: 24 July 2025

AIM: To create a Work Breakdown Structure (WBS) for the " TheraXbot- A mental health companion " minor project.

THEORY: A Work Breakdown Structure (WBS) is a deliverable-oriented hierarchical decomposition of the work to be executed by the project team to accomplish the project objectives and create the required deliverables. It organizes and defines the total scope of the project. Each descending level represents an increasingly detailed definition of the project work.

Key Characteristics of a WBS:

1. **Deliverable-Oriented:** The WBS is structured around the tangible outcomes or deliverables of the project, rather than the activities required to produce them. This focus ensures that all work contributes directly to a project output.
2. **Hierarchical:** It presents the project work in a multi-level structure, starting with the main project at the top and progressively breaking it down into smaller, more manageable components.
3. **Decomposition:** The process of breaking down project deliverables into smaller, more manageable components is called decomposition. This continues until the work packages (the lowest level of the WBS) are defined.
4. **100% Rule:** This fundamental rule states that the WBS includes 100% of the work defined by the project scope and captures all deliverables – internal, external, interim, and final – in terms of work to be completed, including project management. It ensures that no work is left out and no extraneous work is included.
5. **Mutually Exclusive Elements:** Each element in the WBS should be distinct and not overlap with other elements to avoid confusion regarding responsibility and effort.
6. **Work Packages:** The lowest level of the WBS is called a work package. A work package is a manageable piece of work that can be realistically and confidently estimated in terms of cost, time, and resources. It is typically assigned to a specific individual or team.

Benefits of using a WBS:

- **Clarity of Scope:** Provides a clear and comprehensive understanding of the project scope to all stakeholders.
- **Improved Planning:** Facilitates detailed planning, including scheduling, cost estimation, and resource allocation.
- **Better Control:** Enables effective monitoring and control of project progress against defined deliverables.

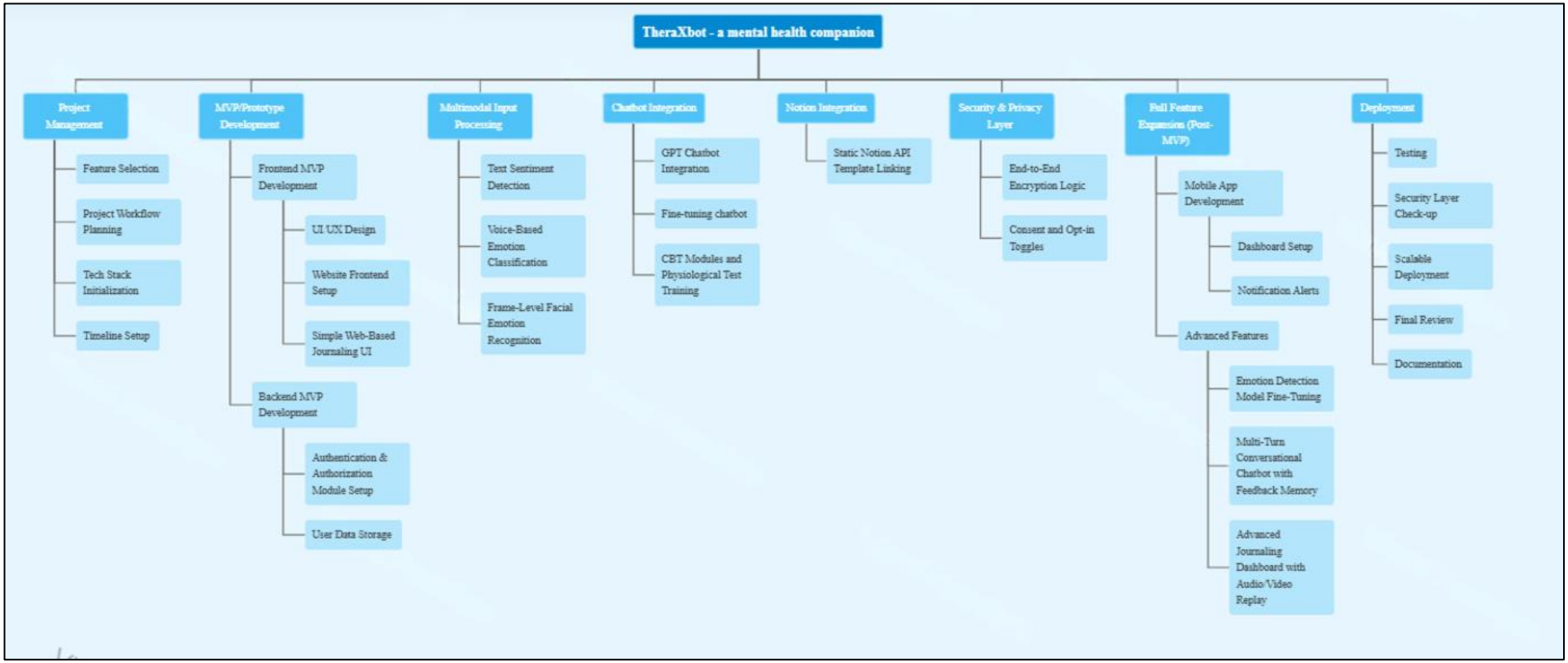
- **Enhanced Communication:** Serves as a common framework for communication among team members and stakeholders.
- **Risk Identification:** Helps in identifying potential risks and developing mitigation strategies at an early stage.
- **Accountability:** Clearly assigns responsibility for each work package, improving accountability.

Project Workflow Description

The project workflow diagram for **TheraXBot** outlines the entire development lifecycle in a structured and modular manner. It visually represents how different components and phases of the system are interconnected, ensuring that the project progresses in a logical and efficient sequence.

1. **Project Management**
   - Initial phase covering **feature selection, workflow planning, tech stack initialization, and timeline setup**. This stage sets the foundation for all subsequent development activities.
2. **MVP / Prototype Development**
   - Focuses on building the **minimum viable product (MVP)** with two parts:
     - **Frontend MVP Development**: Includes UI/UX design, website frontend setup, and a simple web-based journaling interface.
     - **Backend MVP Development**: Covers authentication, authorization module setup, and secure user data storage.
3. **Multimodal Input Processing**
   - Core AI functionalities are implemented here, including:
     - **Text sentiment detection** using NLP models.
     - **Voice-based emotion classification** through speech analysis.
     - **Frame-level facial emotion recognition** for video-based journaling.
4. **Chatbot Integration**
   - Integration of a **GPT-powered therapeutic chatbot**.
   - Fine-tuning the chatbot with **CBT (Cognitive Behavioral Therapy) modules** and psychological test-based training for personalized support.
5. **Notion Integration**
   - Linking with **Notion API templates** to provide a digital mental wellness dashboard, where users can view and organize their journaling data.

6. **Security & Privacy Layer**
    - Ensures user confidentiality through **end-to-end encryption** and **consent-based opt in toggles** for sensitive data like audio and video inputs.
7. **Full Feature Expansion (Post-MVP)**
    - Development of advanced functionalities after MVP validation, including:
        - **Mobile App Development** with dashboard setup and notification alerts.
        - **Emotion detection model fine-tuning** for improved accuracy.
        - **Multi-turn conversational chatbot** with memory and feedback handling.
        - **Advanced journaling dashboard** with audio/video replay features.
8. **Deployment**
    - Final phase covering **testing, security layer check-ups, scalable deployment, final review, and documentation**. This ensures the system is production-ready and user-friendly.


TheraXbot - a mental health companion
Project Management
Feature Selection
Project Workflow Planning
Tech Stack Initialization
Timeline Setup
MVP/Prototype Development
Frontend MVP Development
UI UX Design
Website Frontend Setup
Simple Web-Based Journaling UI
Backend MVP Development
Authentication & Authorization Module Setup
User Data Storage
Multimodal Input Processing
Text Sentiment Detection
Voice-Based Emotion Classification
Frame-Level Facial Emotion Recognition
Chatbot Integration
GPT Chatbot Integration
Fine-tuning chatbot
CBT Modules and Physiological Test Training
Notion Integration
Static Notion API Template Linking
Security & Privacy Layer
End-to-End Encryption Logic
Consent and Opt-in Toggles
Full Feature Expansion (Post-MVP)
Mobile App Development
Dashboard Setup
Notification Alerts
Advanced Features
Emotion Detection Model Fine-Tuning
Multi-Turn Conversational Chatbot with Feedback Memory
Advanced Journaling Dashboard with Audio/Video Replay
Deployment
Testing
Security Layer Check-up
Scalable Deployment
Final Review
Documentation

LAB ASSIGNMENT- 3

DATE: 31 July 2025

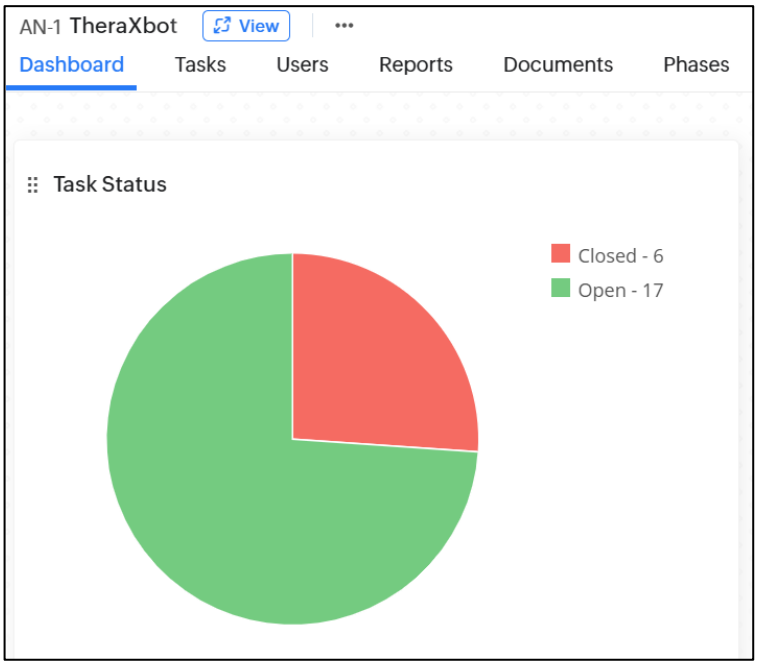
AIM: To use Zoho Projects as a project management tool to track, manage, and report on the development of the " TheraXbot- A mental health companion " minor project.

THEORY:

Zoho Projects is a cloud-based project management software that provides a comprehensive set of tools to plan, track, and collaborate on projects. It is designed to help teams manage their work effectively, from defining project milestones to tracking individual tasks and generating reports. For a minor project, using a tool like Zoho Projects is beneficial because it enforces structure and provides visibility into the project's progress.

Key Features Relevant to Project:

- **Task Management:** Zoho Projects allows you to break down your project into a series of manageable tasks. You can assign these tasks to team members, set start and due dates, and specify priority levels. This is directly aligned with the Work Breakdown Structure (WBS) you have already created.
- **Milestones:** You can define key milestones to mark significant achievements in your project. These act as high-level markers that help you track major phases, such as the completion of the "Core AI & Backend Development" or "Testing & Quality Assurance."
- **Gantt Charts:** The platform can generate Gantt charts that visually represent your project timeline. This helps in understanding dependencies between tasks and identifying potential scheduling conflicts.
- **Collaboration:** Zoho Projects includes collaboration features like comments and forums, which facilitate communication among team members.
- **Reports:** It can generate various reports on task progress, resource utilization, and overall project health, which is essential for monitoring and control.

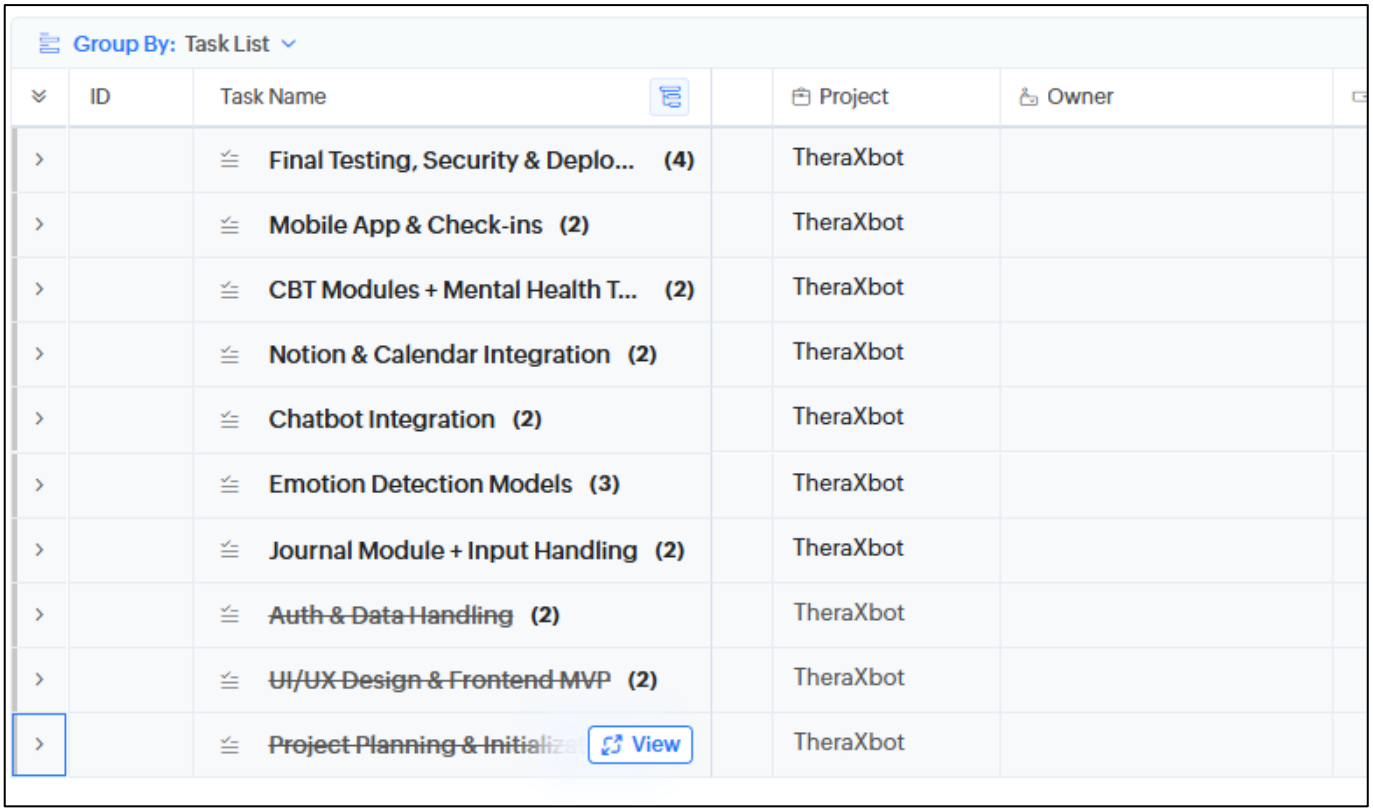


Project Dashboard Description

The TheraXBot project dashboard provides a real-time overview of task completion and pending activities. As seen in the task status chart, a total of 23 tasks were created to cover different phases of development, ranging from project planning and module integration to testing and deployment. Out of these, 6 tasks have been successfully closed, while 17 tasks remain open for ongoing development.

The pie chart clearly illustrates the proportion of work completed versus pending. The green section represents open tasks, signifying modules and integrations that are still under active progress. The red section indicates closed tasks, reflecting milestones already achieved. This visualization helps the team track productivity, prioritize pending work, and maintain accountability.

By maintaining this dashboard, the project team ensures transparent progress tracking, structured task management, and adherence to the project schedule. It also allows stakeholders and the guide to quickly assess the overall completion rate and identify focus areas that require immediate attention.



Project Schedule Description

The development of TheraXBot – A Multimodal Mental Health Companion has been organized into a structured schedule, ensuring that each module is designed, implemented, and tested

systematically. The schedule follows a progressive build-up, starting from core functionalities and moving toward advanced integrations and deployment.

- **Project Planning & Initialization**
  - The project begins with requirement analysis, team coordination, and tech stack setup. This ensures clarity on scope, objectives, and feasibility.
- **UI/UX Design & Frontend MVP**
  - Initial efforts focus on designing a simple and intuitive journaling interface. A minimal frontend prototype is created to validate usability and user interaction flow.
- **Authentication & Data Handling**
  - Secure login/signup is implemented using Firebase/Auth0, with encrypted storage in MongoDB/Firebase. User privacy and data confidentiality are given top priority.
- **Journal Module & Input Handling**
  - Users can submit entries through text, audio, and video. Multimodal input pipelines are developed for seamless journaling.
- **Emotion Detection Models**
  - AI models are integrated to detect emotions from text (BERT), speech (tone analysis), and facial expressions (DeepFace/FER). These models form the foundation of emotional intelligence in TheraXBot.
- **Chatbot Integration**
  - A therapeutic chatbot is embedded, powered by GPT, offering CBT-based guidance, mindfulness prompts, and conversational support.
- **Notion & Calendar Integration**
  - Journals and emotional summaries are synced with Notion for personalized dashboards. Calendar integration supports daily check-ins and reminders.
- **CBT Modules + Mental Health Training**
  - Evidence-based psychological frameworks such as CBT and guided exercises are integrated to assist users in building healthy coping mechanisms.
- **Mobile App & Check-ins**
  - A cross-platform mobile app is developed using React Native, enabling on-the-go journaling, notifications, and mood tracking.
- **Final Testing, Security & Deployment**
  - Comprehensive testing is performed to ensure accuracy, reliability, and user safety. Security reviews validate encryption and privacy standards before deployment to cloud infrastructure.

LAB ASSIGNMENT- 4

DATE: 7 August 2025

AIM: To create and explain a Resource Planning Template for the " TheraXbot – a multimodal mental health companion " minor project.

THEORY:

A Resource Planning Template is a fundamental project management tool meticulously designed to organize and manage the resources essential for a project's success. It serves as a comprehensive tabular document that systematically maps project tasks and activities to the necessary resources, their assigned roles, and their specific allocation timeline. While the term "Resource Planning" refers to the overarching process of identifying and scheduling resources, the "Resource Planning Template" is the specific and tangible artifact created to formalize and visualize this plan.

The strategic implementation of such a template is driven by several key objectives:

1. **To ensure resource availability:** By precisely defining who is needed and at what specific time, the template proactively helps to prevent resource bottlenecks and ensures that the right skills are available for each task. This is particularly crucial for complex projects involving specialized skills like machine learning and automation.
2. **To manage timelines:** The template provides a detailed and clear schedule with definitive start and end dates for each activity. This visual roadmap is indispensable for monitoring project progress, identifying potential delays, and ensuring the project remains on track for timely completion.
3. **To track and forecast effort:** By providing realistic estimates of the daily and total hours required for each task, the template establishes a foundational basis for effective workload management. It also aids in accurate budget forecasting and provides a benchmark against which actual effort can be measured during the project's execution.

This template is not merely a static document; it is a vital communication tool that serves as a single source of truth for all stakeholders regarding resource allocation and the project timeline. It ensures that the project team operates with maximum efficiency and that the project remains aligned with its planned course from initiation to completion. Poor resource planning, conversely, can lead to project delays, team burnout, and a failure to meet project objectives.

Project Task Schedule Explanation:-

The table outlines the detailed task plan for the development of **TheraXbot – a multimodal mental health companion**. It specifies each activity, assigned resources, timelines, dependencies and relevant remarks.

Column-wise Explanation:-

1. **Task/Activity**
    - Lists the major development modules for the MVP (Minimum Viable Product).

- Examples: Frontend MVP (Web UI), Backend MVP (APIs & Auth), Facial Emotion Recognition Module.

2. **Resource Name**
   - The individual(s) responsible for executing the task.
   - Example: Ananya Gupta, Anam Baquri or Both for collaborative tasks.
3. **Role/Designation**
   - Defines the specific technical role for each assigned person.
   - Example: Frontend Developer, AI/Chatbot Developer, CV Engineer.
4. **Start Date & End Date**
   - The scheduled start and completion dates for each task.
   - Dates are in **DD-MMM-YY** format.
5. **Effort (Hours/Day)**
   - The expected daily working hours to be allocated for each task.
   - Ranges between 3–5 hours/day.
6. **Total Hours**
   - The total number of working hours calculated for the task's duration.
   - Formula: Effort (Hours/Day) × Number of Working Days.
7. **Dependencies**
   - Identifies the prerequisite modules that must be completed before starting the task.
   - Example: Backend MVP is dependent on Frontend MVP, Speech-to-Text Integration depends on STT + Emotion Modules.
8. **Remarks**
   - Provides technical notes, tools, or frameworks to be used for task execution.
   - Examples: Use React.js + Tailwind, Use BERT via HuggingFace, Use AES-256, OAuth2.0, SSL.

	A	B	C	D	E	F	G	H	I
1	**Task/Activity**	**Resource Name**	**Role/Designation**	**Start Date**	**End Date**	**Effort (Hours/Day)**	**Total Hours**	**Dependencies**	**Remarks**
2	Frontend MVP (Web UI)	Ananya Gupta	Frontend Developer	12-Aug-25	25-Aug-25	4	56	Project Initialization	Use React.js + Tailwind
3	Backend MVP (APIs & Auth)	Anam Baquri	Backend Developer	12-Aug-25	28-Aug-25	4	68	Frontend MVP	Use FastAPI + Firebase Auth
4	Text Sentiment Analysis Integration	Ananya Gupta	ML Engineer	26-Aug-25	02-Sep-25	5	40	Backend MVP	Use BERT via HuggingFace
5	Voice Emotion Classifier Setup	Anam Baquri	Audio ML Engineer	29-Aug-25	05-Sep-25	4	32	Backend MVP	Using openSMILE or SpeechBrain
6	Facial Emotion Recognition Module	Ananya Gupta	CV Engineer	03-Sep-25	10-Sep-25	4	32	Backend MVP	Use DeepFace or MediaPipe
7	Speech-to-Text Integration	Anam Baquri	Integration Developer	06-Sep-25	12-Sep-25	4	28	Voice Recording Module	Using Whisper API
8	Therapeutic Chatbot Integration	Ananya Gupta	AI/Chatbot Developer	10-Sep-25	17-Sep-25	4	32	STT + Emotion Modules	GPT-4 Turbo + CBT logic
9	Notion API Integration	Anam Baquri	Integration Developer	14-Sep-25	20-Sep-25	3	21	Journaling Module	Static API template push
10	Security & Privacy Layer	Both	Full Stack Developers	18-Sep-25	25-Sep-25	4	56	Core Functionality	Use AES-256, OAuth2.0, SSL
11	Deployment + Testing (MVP)	Both	DevOps + QA	26-Sep-25	30-Sep-25	5	25	All Modules	Host on Render, CI/CD via GitHub

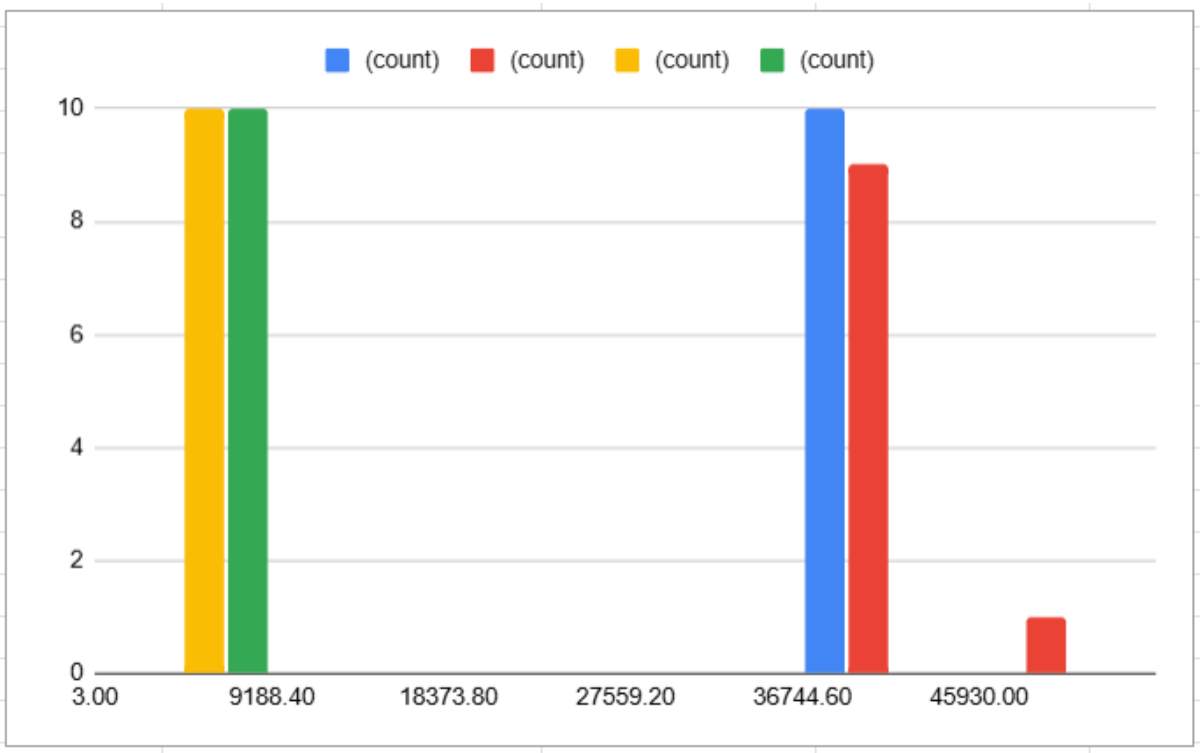
Histogram Description

The histogram represents the distribution of total working hours corresponding to the tasks/activities listed in the project schedule. The x-axis denotes the total hours range, while the y-axis shows the frequency (count) of tasks falling within each range.

From the histogram:

- The majority of the tasks are concentrated in two ranges — around 9,188 hours and 36,744 hours, each with a frequency of 10 and 9 tasks respectively.
- A smaller number of tasks fall near 45,930 hours, with only 1 task in this range.
- No tasks were recorded in the intermediate ranges (18,373.80 to 27,559.20).

This distribution indicates that the workload for most tasks is clustered into specific effort bands (low-to-moderate and moderate-to-high), with very few outlier tasks requiring extremely high total hours.



LAB ASSIGNMENT- 5

DATE: 14 August 2025

AIM: To generate a sequence diagram for the " TheraXbot – a multimodal mental health companion " minor project.

THEORY:

A sequence diagram is a type of interaction diagram in UML (Unified Modeling Language) that shows how objects or components interact with each other in a time-ordered sequence. It visually represents the flow of messages, events, and actions between different system components to achieve a particular functionality or use case. Sequence diagrams are widely used in software engineering to model system behavior, understand dependencies, and document the dynamic aspect of a system.

Components of a Sequence Diagram

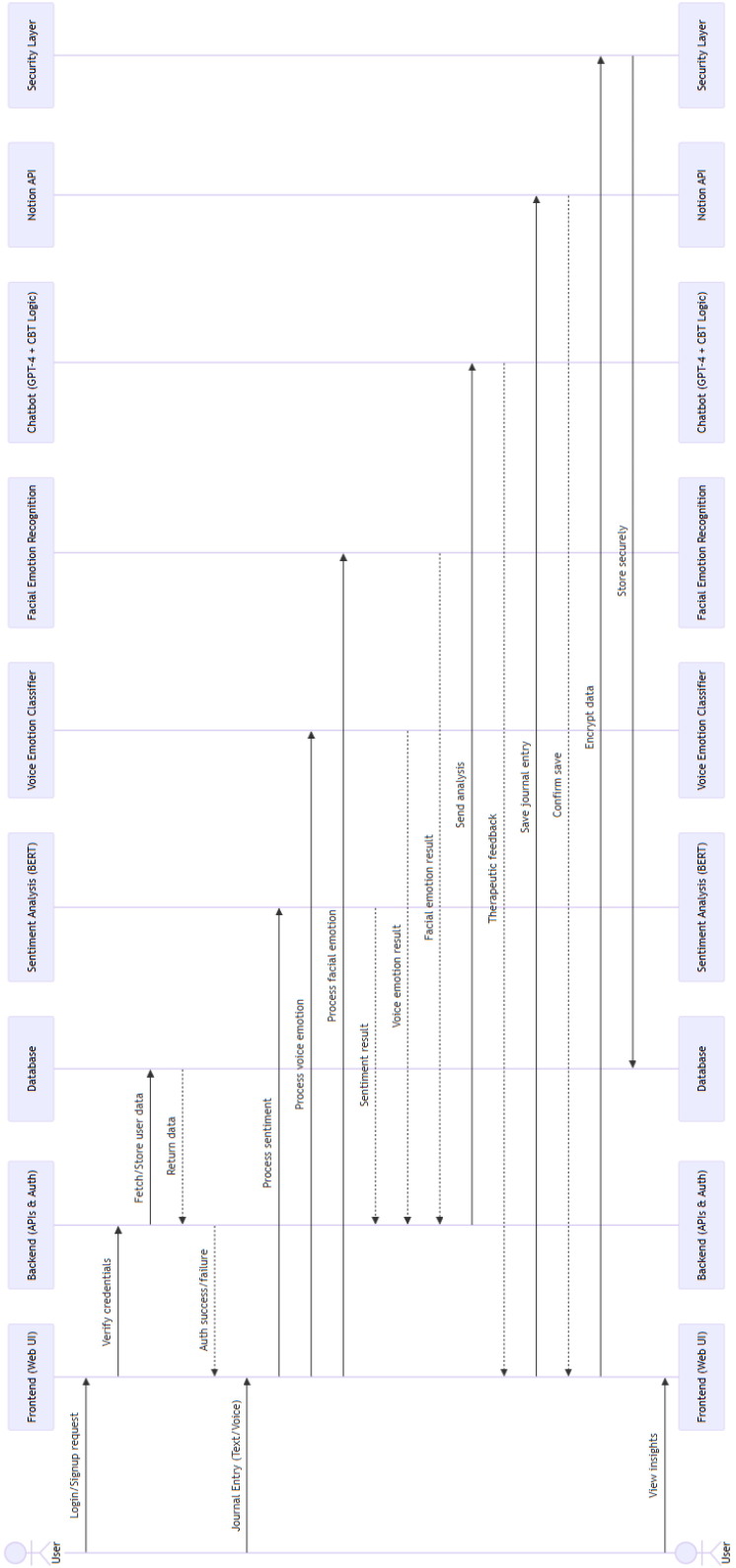
1. **Actors** – External entities (such as users) that interact with the system.
2. **Objects/Participants** – System modules or components that perform specific roles. These are shown as vertical lifelines.
3. **Lifelines** – Dashed vertical lines representing the existence of an object over a period of time.
4. **Messages** – Arrows between lifelines that represent communication, method calls, or data transfer.
5. **Activation Bars** – Thin rectangles on lifelines showing the time during which a component is active in the process.
6. **Return Messages** – Dotted arrows that show responses or data being sent back after processing.

Explanation of the Given Diagram

The given sequence diagram represents the workflow of the TheraXBot MVP project, illustrating the interaction between the user, frontend, backend, and different AI/ML modules.

- The process begins with the user attempting to log in or sign up via the Frontend (Web UI). The request is passed to the Backend (APIs & Auth), which verifies the credentials by interacting with the Database. The result is returned to the frontend, and authentication success or failure is communicated to the user.
- After authentication, the user can submit a journal entry (text or voice input). This input is processed by multiple AI modules:
    - Sentiment Analysis (BERT) analyzes the text sentiment.
    - Voice Emotion Classifier processes vocal tone for emotional context.
    - Facial Emotion Recognition evaluates facial expressions.

- These results are sent to the Backend, which forwards them to the Therapeutic Chatbot (GPT-4 + CBT Logic) for generating personalized therapeutic responses. The chatbot then sends the feedback back to the frontend for the user to view.
- Simultaneously, journal entries are saved through the Notion API, and the Security Layer ensures encryption before securely storing the data in the Database.
- Finally, the user can request to view insights, such as historical mood tracking and emotional trends, which are retrieved from the database and displayed via the frontend.



LAB ASSIGNMENT- 6

DATE: 21 August 2025

AIM: To generate an activity diagram for the " TheraXbot – a multimodal mental health companion " minor project.

THEORY:

An Activity Diagram is a type of UML (Unified Modeling Language) behavioral diagram used to represent the workflow or activities of a system. It visually illustrates how processes flow from one activity to another, including decision points, branching, concurrency, and termination.

Activity diagrams are widely used to:

- Model business processes and workflows.
- Show the sequence of operations in a system.
- Represent logic of functions or use cases.
- Provide a clear visual understanding of how users interact with the system.

Components of an Activity Diagram

1. **Start Node** – Represented as a filled circle, indicating the entry point of the workflow.
2. **Activity/Action State** – Rounded rectangles that represent tasks or actions performed.
3. **Decision Node** – A diamond shape used to show branching (e.g., conditions like Yes/No, Valid/Invalid).
4. **Control Flows/Transitions** – Arrows that indicate the flow from one activity to the next.
5. **End Node** – A filled circle with a boundary (bullseye), marking the termination of the process.

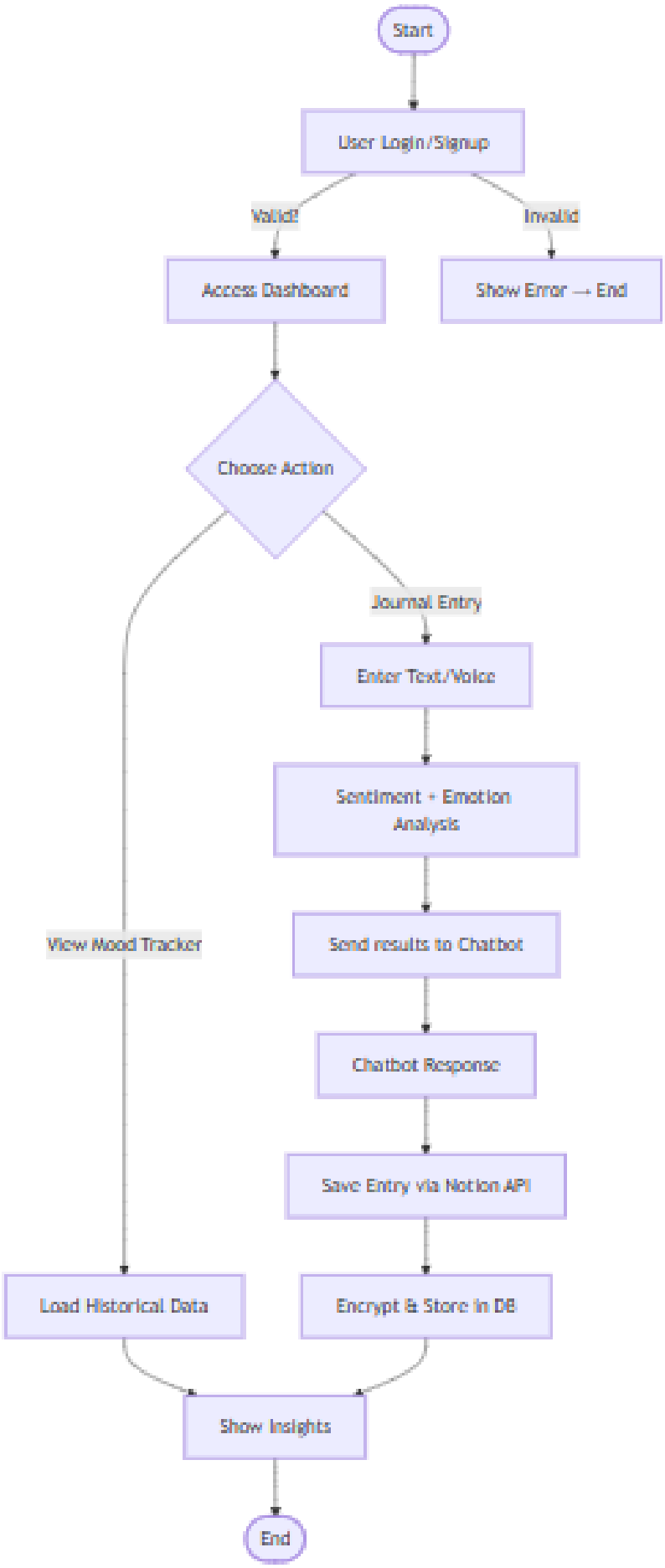
Explanation of the Given Diagram

The provided activity diagram models the workflow of a mental health journaling and chatbot application. It describes how a user interacts with the system, starting from login to receiving insights. The process is as follows:

1. **Start Node**
    - The workflow begins when the user starts the application.
2. **User Login/Signup**
    - The user attempts to log in or sign up for the application.
3. **Decision (Valid/Invalid Login)**
    - If the credentials are valid, the user gains access to the Dashboard.

- If invalid, the system shows an Error and terminates the process.

4. **Choose Action**
    - From the dashboard, the user selects an action. Two primary actions are available:
        - Journal Entry
        - View Mood Tracker
5. **Journal Entry Path**
    - The user enters input (either text or voice).
    - The system performs Sentiment + Emotion Analysis on the entry.
    - Results are then sent to the Chatbot, which generates a response.
    - The entry is saved using the Notion API and then encrypted & stored in the database for security.
6. **View Mood Tracker Path**
    - The system retrieves and loads historical data of user's mood entries.
7. **Show Insights**
    - Both journal entry processing and mood tracker retrieval lead to a final activity where the system displays insights to the user, summarizing their emotional state and progress.
8. **End Node**
    - The workflow concludes after insights are shown.


Start
User Login/Signup
Valid
Invalid
Access Dashboard
Show Error → End
Choose Action
Journal Entry
Enter Text/Voice
Sentiment + Emotion Analysis
View Mood Tracker
Send results to Chatbot
Chatbot Response
Save Entry via Notion API
Load Historical Data
Encrypt & Store in DB
Show Insights
End

LAB ASSIGNMENT- 7

DATE: 28 August 2025

AIM: To do risk analysis for the " TheraXbot – a multimodal mental health companion " minor project.

THEORY:

Risk assessment is the systematic process of identifying, analyzing, and evaluating potential risks that may affect the success of a project or system. In software engineering, risk assessment helps teams proactively recognize vulnerabilities, plan mitigation strategies, and ensure smooth project delivery.

The purpose of a risk assessment is to:

- Detect possible technical, operational, or security issues.
- Estimate their impact (severity) and likelihood (probability of occurrence).
- Plan mitigation strategies to reduce or eliminate the risks.

Key Components of a Risk Assessment Table

1. **Task/Activity** – The specific component or module of the project under review.
2. **Potential Risk** – The possible problem, failure, or challenge that could occur in the task.
3. **Impact (High/Medium/Low)** – The severity of damage if the risk materializes.
4. **Likelihood (High/Medium/Low)** – The probability of the risk occurring.
5. **Mitigation Strategy** – Steps or countermeasures planned to minimize or eliminate the risk.

Explanation of the Given Table

The provided risk assessment table highlights potential challenges in building an AI-driven MVP system with frontend, backend, AI models, and integrations. Below is the explanation:

1. **Frontend MVP (Web UI)**
    - *Risk:* Delays due to React/JS/Tailwind complexity or design inconsistencies.
    - *Impact/Likelihood:* Medium.
    - *Mitigation:* Use reusable UI components, follow design standards, and conduct early testing.
2. **Backend MVP (APIs & Auth)**
    - *Risk:* Authentication issues with FastAPI or Firebase, leading to failures.

- *Impact/Likelihood:* Medium.
- *Mitigation:* Test auth flows early and maintain backup authentication strategies (JWT/OAuth).

3. **Text Sentiment Analysis**
   - *Risk:* Model accuracy may be low if dataset is not domain-specific.
   - *Impact:* High; *Likelihood:* Medium.
   - *Mitigation:* Fine-tune models with domain data and fallback to HuggingFace pre-trained models.
4. **Voice Emotion Classifier Setup**
   - *Risk:* OpenSMILE/SpeechBrain ML models may fail to integrate.
   - *Impact:* High; *Likelihood:* Medium.
   - *Mitigation:* Proof-of-concept before integration; modular audio pipelines.
5. **Facial Emotion Recognition Module**
   - *Risk:* Misclassification due to poor dataset or low camera quality.
   - *Impact:* High; *Likelihood:* Medium.
   - *Mitigation:* Use robust datasets (FER+, AffectNet); fallback to MediaPipe.
6. **Speech-to-Text Integration**
   - *Risk:* Whisper API may fail in noisy environments.
   - *Impact:* Medium; *Likelihood:* High.
   - *Mitigation:* Implement noise reduction preprocessing and allow manual correction.
7. **Therapeutic Chatbot Integration**
   - *Risk:* GPT-4 Turbo may result in high API costs or inconsistent responses.
   - *Impact:* High; *Likelihood:* Medium.
   - *Mitigation:* Optimize API calls; add caching and fallback queries.
8. **Notion API Integration**
   - *Risk:* API template push may fail due to rate limits or changes.
   - *Impact:* Medium; *Likelihood:* Low.
   - *Mitigation:* Validate with sandbox testing; use versioned API docs.
9. **Security & Privacy Layer**
   - *Risk:* Data breaches or weak encryption compromising sensitive data.

- *Impact:* High; *Likelihood:* Medium.
- *Mitigation:* Use AES-256 encryption, OAuth2, SSL, and penetration testing.

10. **Deployment + Testing (MVP)**
    - *Risk:* CI/CD pipeline errors on Render/GitHub.
    - *Impact:* Medium; *Likelihood:* Medium.
    - *Mitigation:* Use staging environment and rollback plans.

Task/Activity	Potential Risk	Impact (High/Med/Low)	Likelihood (High/Med/Low)	Mitigation Strategy
Frontend MVP (Web UI)	Delay in UI setup due to ReactJS/Tailwind complexity or design inconsistencies	Medium	Medium	Use reusable UI components; follow design system; conduct early UI testing
Backend MVP (APIs & Auth)	Issues with FastAPI or Firebase Auth integration causing authentication failures	High	Medium	Test auth flows early; maintain backup auth strategy (JWT/OAuth)
Text Sentiment Analysis	Model accuracy may be low using BERT on custom dataset	Medium	Medium	Fine-tune model with domain-specific data; fallback to pre-trained HuggingFace models
Voice Emotion Classifier Setup	Audio ML model (openSMILE/SpeechBrain) may not integrate smoothly	High	Medium	Perform proof-of-concept before integration; modularize audio pipeline
Facial Emotion Recognition Module	Model may misclassify due to poor dataset or camera quality	High	Medium	Use robust datasets (FER+, AffectNet); fallback to MediaPipe facial landmarks
Speech-to-Text Integration	Whisper API may produce errors in noisy environments	Medium	High	Implement noise reduction preprocessing; allow manual text correction
Therapeutic Chatbot Integration	GPT-4 Turbo integration may result in high API costs or inconsistent responses	High	Medium	Optimize API calls; add CBT logic fallback; cache frequent queries
Notion API Integration	Static API template push may fail due to API changes or rate limits	Medium	Low	Validate API with sandbox testing; maintain versioned API docs
Security & Privacy Layer	Data breaches or weak encryption impacting sensitive user data	High	Medium	Use AES-256 encryption, OAuth2.0, SSL; conduct penetration testing
Deployment + Testing (MVP)	CI/CD pipeline errors during hosting on Render/GitHub actions	Medium	Medium	Test CI/CD pipeline with staging environment; rollback deployment plan

LAB ASSIGNMENT- 8

DATE: 4 September 2025

AIM: To generate gantt chart for the " TheraXbot – a multimodal mental health companion " minor project.

THEORY:

A Gantt chart is a project management tool used to visually represent the schedule of tasks over time. It was introduced by Henry L. Gantt in the early 20th century and remains one of the most widely used techniques for planning and tracking project timelines.

It helps project managers and teams to:

- Define start and end dates of tasks.
- Show the duration (tenure) of each task.
- Identify dependencies between tasks.
- Monitor project progress in a visual timeline format.

Components of a Gantt Chart

1. **Tasks/Activities** – Work items that need to be completed.
2. **Start Date** – The date when each task is scheduled to begin.
3. **Duration (Tenure)** – The number of days allocated for the task.
4. **Timeline (Horizontal Axis)** – Calendar dates to show progress across weeks/months.
5. **Bars** – Horizontal bars represent the time span for each task. Longer bars indicate longer duration.
6. **Dependencies (if shown)** – Relationships between tasks (e.g., one task starting after another finishes).

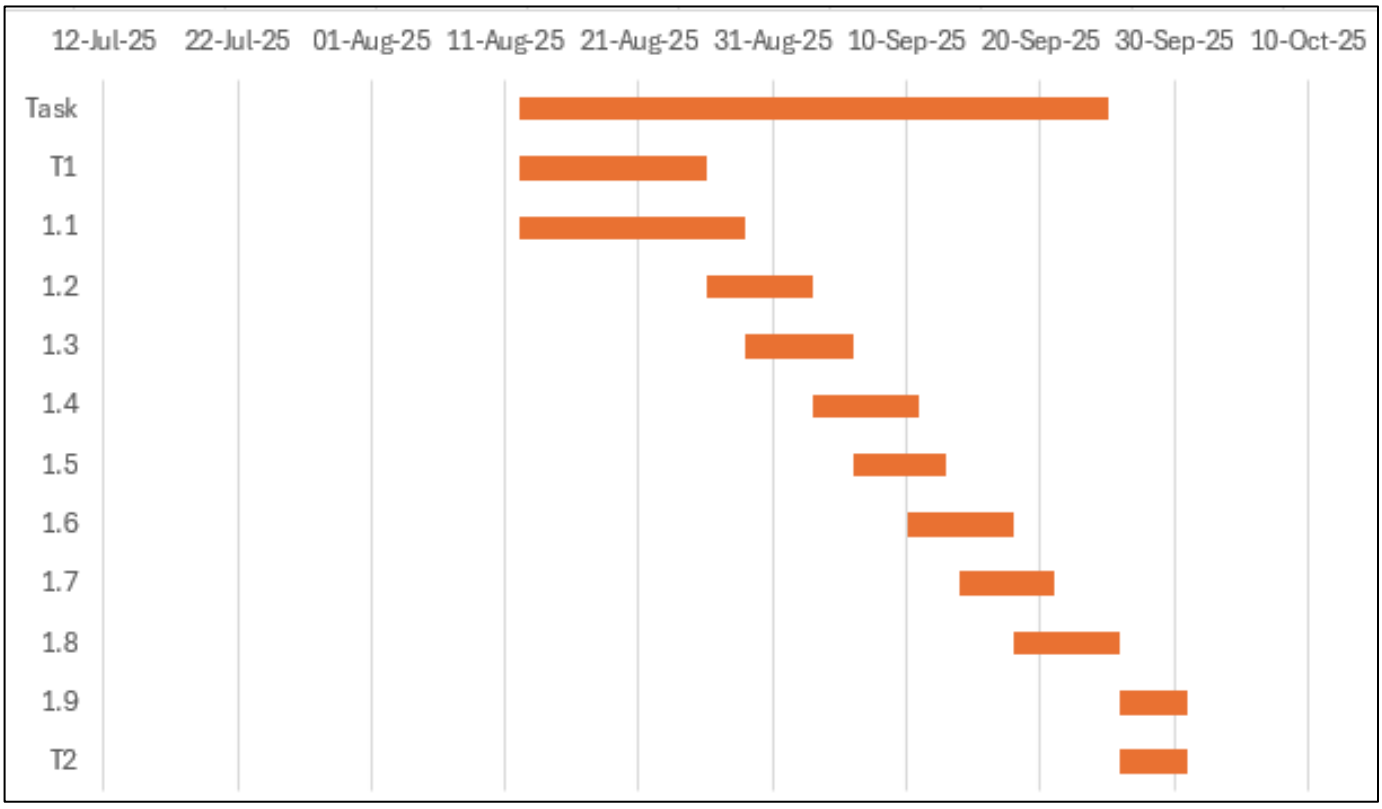
Explanation of the Given Chart

The dataset and chart provided describe the schedule of **Tasks T1 and T2** along with their sub-tasks.

1. **Task T1** starts on **12-Aug-25** with a total tenure of **44 days**.
    - Subtasks **1.1 to 1.9** break this into smaller segments.
    - For example:
        - *Task 1.1* starts on **12-Aug-25** and lasts for **14 days**.
        - *Task 1.2* starts on **12-Aug-25** and runs for **17 days**.
        - *Task 1.3* starts on **26-Aug-25** and lasts for **8 days**.
        - Subsequent tasks (1.4 – 1.9) continue until **26-Sep-25**, covering different 7–8 day slots.

2. **Task T2** starts on **26-Sep-25** with a tenure of **5 days**.
    - *Subtask 2.1* also starts on the same date with a duration of **5 days**, ending the project around **30-Sep-25**.
3. **Visualization** (in the chart):
    - Each task/subtask is represented as a horizontal bar.
    - The x-axis shows calendar dates from **July to October 2025**.
    - The y-axis lists all tasks (T1, T2, and their subtasks).
    - The length of each bar represents the tenure/duration.
    - Subtasks are sequentially aligned, ensuring smooth workflow under the parent tasks.

Task	Start Date	Tenure
T1	12-Aug-25	44
1.1	12-Aug-25	14
1.2	12-Aug-25	17
1.3	26-Aug-25	8
1.4	29-Aug-25	8
1.5	03-Sep-25	8
1.6	06-Sep-25	7
1.7	10-Sep-25	8
1.8	14-Sep-25	7
1.9	18-Sep-25	8
T2	26-Sep-25	5
2.1	26-Sep-25	5



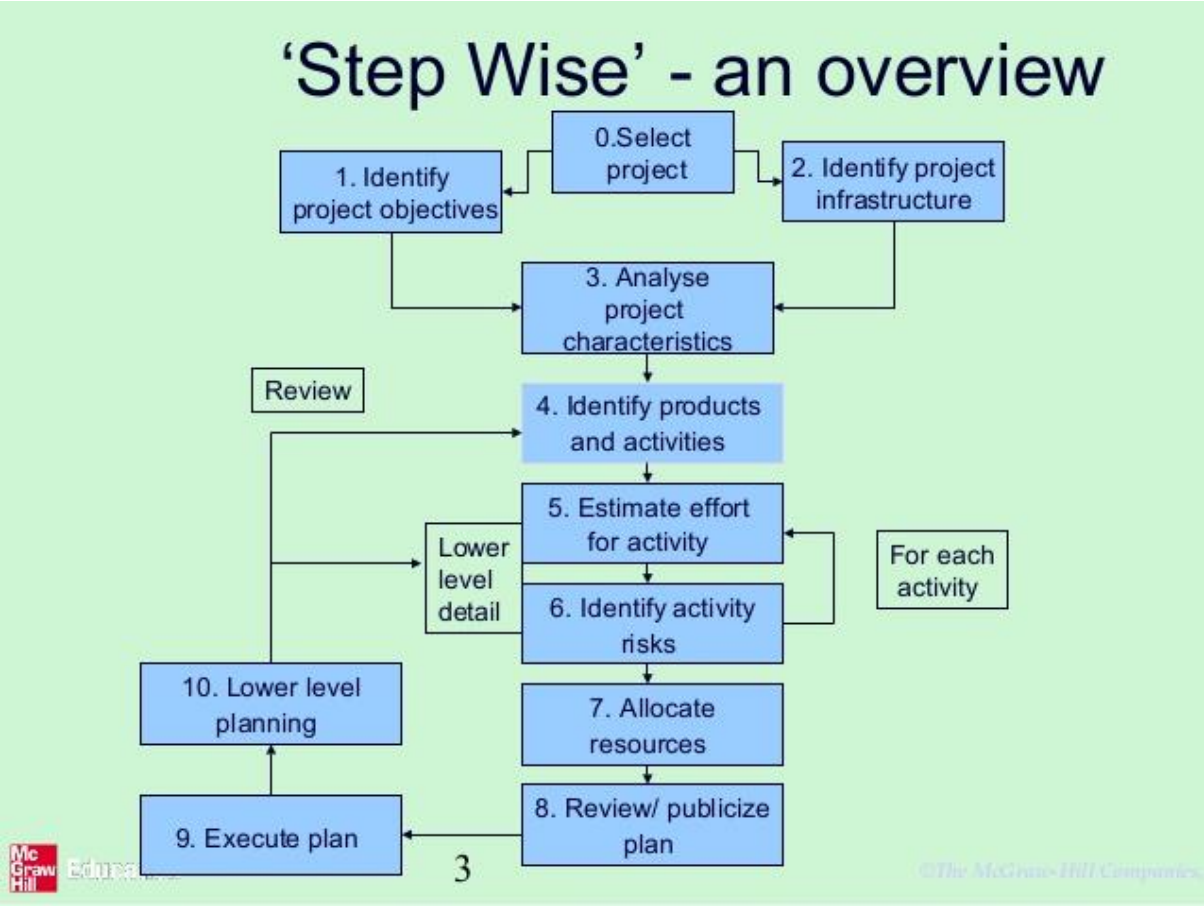
LAB ASSIGNMENT- 9

DATE: 11 September 2025

AIM: To generate step-wise planning chart for the " TheraXbot – a multimodal mental health companion " minor project.

THEORY:

Step-wise project planning involves breaking down the project into systematic phases, each with a defined scope, deliverables, and timeline. The following plan outlines the development process of the project, covering frontend, backend, AI/ML modules, chatbot, integration, and deployment.



1. Frontend Development

- **Task:** Develop the Minimum Viable Product (MVP) for the Web User Interface (UI).
- **Timeline:** 17th August 2023 – 24th August 2023.
- **Objective:** To design and implement a user-friendly interface that serves as the entry point for users to interact with the application.

2. Backend Development

- **Task:** Build the Backend MVP including APIs and Authentication modules.

- **Timeline:** 17th August 2023 – 24th August 2023.
- **Objective:** To establish the server-side architecture for secure communication, authentication, and data handling.

3. AI/ML Module Development

- **Tasks:**
  - **Text Sentiment Analysis** (24th Aug – 31st Aug 2023)
  - **Voice Emotion Classifier** (28th Aug – 4th Sept 2023)
  - **Facial Emotion Recognition** (30th Aug – 6th Sept 2023)
  - **Speech-to-Text Integration** (31st Aug – 7th Sept 2023)
- **Objective:** To implement intelligent modules that analyze text, voice, and facial inputs to enhance interactivity and personalization.

4. Chatbot Integration

- **Task:** Integrate a Therapeutic Chatbot.
- **Timeline:** 7th Sept – 14th Sept 2023.
- **Objective:** To provide an interactive conversational agent that leverages AI modules to assist users effectively.

5. System Integration

- **Tasks:**
  - **Notion API Integration** (14th Sept – 21st Sept 2023)
  - **Security & Privacy Layer Implementation** (18th Sept – 21st Sept 2023)
- **Objective:** To ensure seamless data flow between modules and maintain system security and user privacy.

6. Deployment

- **Task:** Deployment and Testing of the MVP.
- **Timeline:** 21st Sept – 28th Sept 2023.
- **Objective:** To host, test, and validate the complete system for functional reliability and performance.

LAB ASSIGNMENT- 10

DATE: 18 September 2025

AIM: To evaluate the critical path for the given dataset.

THEORY:

The critical path in a project network is the longest-duration path from start to finish.

- It determines the minimum project duration.
- Activities on the critical path have zero total float (slack) — any delay in one of them delays the whole project.
- Identifying the critical path focuses attention on the tasks that must be monitored and controlled to meet the deadline.

Key formulas used for critical-path (CPM) evaluation:

- Earliest Finish: EF= ES + duration
- For a node with multiple predecessors: ES = max (EF of predecessors)
- Latest Start: LS = LF − duration
- For a node with multiple successors: LF = min (LS of successors)
- Slack (total float) = LS − ES = LF – EF

An activity is critical if Slack = 0.

Project details

Activity	a	m	b
1-2	2	5	8
2-3	3	6	9
2-4	4	8	12
3-5	5	10	15
4-5	2	6	10
5-6	3	7	11

Activity	Depends on	Duration (days)
A	START	5
B	A	4
C	A	6
D	B	5
E	C	8
F	D	6
G	E, F	4
H	G	7
I	H	2

Step 1 — Forward pass (compute ES and EF)

Initialization: ES(A) = 0 (project starts at time 0).

Iteration 1 — Activity A

- Given: Duration(A) = 5.
- ES(A) = 0 (start)
- EF(A) = ES(A) + Duration(A) = 0 + 5 = 5.
  Result: ES(A)=0, EF(A)=5.

Iteration 2 — Activities B and C (both depend on A)

- Both B and C start after A finishes, so:
  - ES(B) = EF(A) = 5.
  - EF(B) = ES(B) + Duration(B) = 5 + 4 = 9.
  - ES(C) = EF(A) = 5.
  - EF(C) = ES(C) + Duration(C) = 5 + 6 = 11.
    Result: ES(B)=5, EF(B)=9; ES(C)=5, EF(C)=11.

Iteration 3 — Activity D (depends on B)

- ES(D) = EF(B) = 9.
- EF(D) = ES(D) + Duration(D) = 9 + 5 = 14.
  Result: ES(D)=9, EF(D)=14.

Iteration 4 — Activity E (depends on C)

- ES(E) = EF(C) = 11.

- EF(E) = ES(E) + Duration(E) = 11 + 8 = 19.
  Result: ES(E)=11, EF(E)=19.

Iteration 5 — Activity F (depends on D)

- ES(F) = EF(D) = 14.
- EF(F) = ES(F) + Duration(F) = 14 + 6 = 20.
  Result: ES(F)=14, EF(F)=20.

Iteration 6 — Activity G (depends on E and F)

- G can start only after both E and F finish. So take the maximum of their EF values:
  - EF(E) = 19, EF(F) = 20 → max(19, 20) = 20.
- ES(G) = max(EF(E), EF(F)) = 20.
- EF(G) = ES(G) + Duration(G) = 20 + 4 = 24.
  Calculation shown: ES(G) = max(19,20) = 20 → EF(G) = 20 + 4 = 24.
  Result: ES(G)=20, EF(G)=24.

Iteration 7 — Activity H (depends on G)

- ES(H) = EF(G) = 24.
- EF(H) = ES(H) + Duration(H) = 24 + 7 = 31.
  Result: ES(H)=24, EF(H)=31.

Iteration 8 — Activity I (depends on H)

- ES(I) = EF(H) = 31.
- EF(I) = ES(I) + Duration(I) = 31 + 2 = 33.
  Result: ES(I)=31, EF(I)=33.

Activity	**Duration**	**ES**	**EF = ES + Duration**
A	5	0	5
B	4	5	9
C	6	5	11
D	5	9	14
E	8	11	19
F	6	14	20
G	4	20	24
H	7	24	31
I	2	31	33

Project earliest finish = EF(I) = 33 days.

Step 2 — Backward pass (compute LF and LS)

Initialization: LF(I) = project finish = 33. Then compute LS(I) = LF(I) − Duration(I).

Iteration 1 (backward) — Activity I

- LF(I) = EF(I) = 33 (finish time).
- LS(I) = LF(I) − Duration(I) = 33 − 2 = 31.
  Result: LS(I)=31, LF(I)=33.

Iteration 2 — Activity H (successor = I)

- H must finish by the time I must start (LS(I)). For nodes with a single successor:
  - LF(H) = LS(successor) = LS(I) = 31.
- LS(H) = LF(H) − Duration(H) = 31 − 7 = 24.
  Result: LS(H)=24, LF(H)=31. (Calculation: LF(H)=31 → LS(H)=24)

Iteration 3 — Activity G (successor = H)

- LF(G) = LS(H) = 24.
- LS(G) = LF(G) − Duration(G) = 24 − 4 = 20.
  Result: LS(G)=20, LF(G)=24.

Iteration 4 — Activities E and F (both are predecessors of G)

- For each predecessor of G, LF(predecessor) = LS(G) (because G is their only common successor here).
  - LF(E) = LS(G) = 20 → LS(E) = LF(E) − Duration(E) = 20 − 8 = 12.
  - LF(F) = LS(G) = 20 → LS(F) = LF(F) − Duration(F) = 20 − 6 = 14.
    Results: LS(E)=12, LF(E)=20; LS(F)=14, LF(F)=20.

Iteration 5 — Activity D (predecessor of F)

- LF(D) = LS(F) = 14.
- LS(D) = LF(D) − Duration(D) = 14 − 5 = 9.
  Result: LS(D)=9, LF(D)=14.

Iteration 6 — Activity C (predecessor of E)

- LF(C) = LS(E) = 12.
- LS(C) = LF(C) − Duration(C) = 12 − 6 = 6.
  Result: LS(C)=6, LF(C)=12.

Iteration 7 — Activity B (predecessor of D)

- LF(B) = LS(D) = 9.

- LS(B) = LF(B) − Duration(B) = 9 − 4 = 5.
  Result: LS(B)=5, LF(B)=9.

Iteration 8 — Activity A (predecessor of B and C)

- A has two successors (B and C). For nodes with multiple successors:
  - LF(A) = min(LS(B), LS(C)) because A must finish in time for both successor chains; the tightest (earliest) requirement governs.
  - LS(B) = 5, LS(C) = 6 → min(5, 6) = 5.
- LF(A) = 5.
- LS(A) = LF(A) − Duration(A) = 5 − 5 = 0.
  Calculation shown: LF(A)=min(5,6)=5 → LS(A)=5−5=0.
  Result: LS(A)=0, LF(A)=5.

Backward pass conclusion => 33.

Activity	Duration	LF	LS = LF − Duration
I	2	33	31
H	7	31	24
G	4	24	20
F	6	20	14
E	8	20	12
D	5	14	9
C	6	12	6
B	4	9	5
A	5	5	0

Step 3 — Slack and identify critical activities

Compute Slack = LS − ES (or LF − EF). Activities with Slack = 0 are critical.

Activity	ES	EF	LS	LF	Slack= LS-ES
A	0	5	0	5	0
B	5	9	5	9	0
C	5	11	6	12	1
D	9	14	9	14	0
E	11	19	12	20	1

F	14	20	14	20	0
G	20	24	20	24	0
H	24	31	24	31	0
I	31	33	31	33	0

- **Slack = 0**: A, B, D, F, G, H, I → these are the **critical activities**.
- **Non-critical with float**: C and E each have Slack = 1 day.

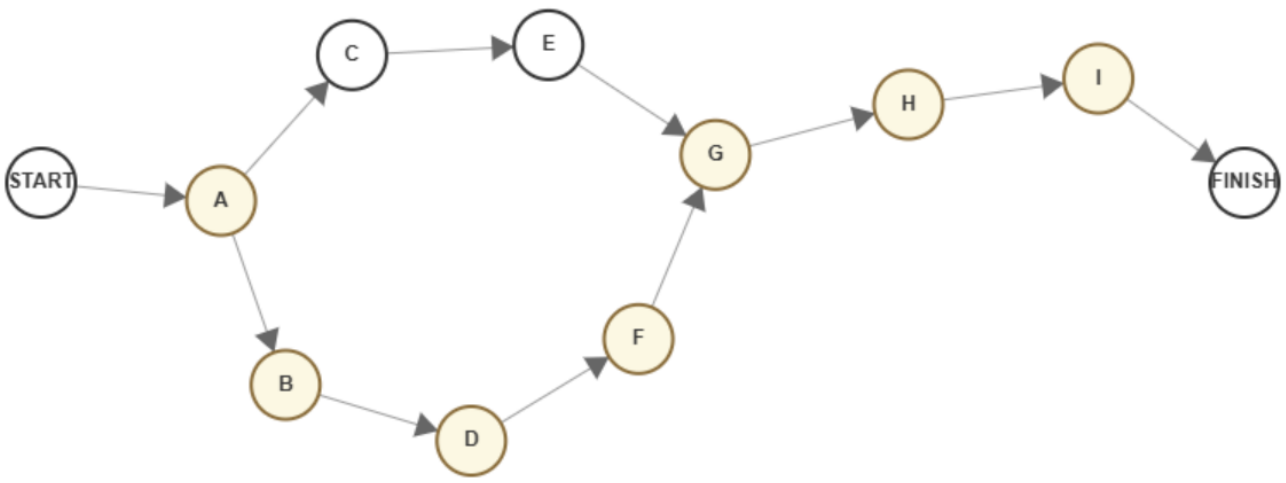
Result — Critical path and project duration

- **Critical path: A → B → D → F → G → H → I**
- **Project duration (minimum possible): 33 days**

Project Information

Activity	Depends on	Time requirement	Early Start	Early Finish	Late Start	Late Finish
A		5	0	5	0	5
B	A	4	5	9	5	9
C	A	6	5	11	6	12
D	B	5	9	14	9	14
E	C	8	11	19	12	20
F	D	6	14	20	14	20
G	E, F	4	20	24	20	24
H	G	7	24	31	24	31
I	H	2	31	33	31	33

Project duration is 33. The shaded rows show the critical activities.



LAB ASSIGNMENT- 11

DATE: 25 September 2025

AIM: To generate pert chart for the " TheraXbot – a multimodal mental health companion " minor project.

THEORY:

The PERT chart is used for project scheduling, coordination, and control. It visually represents task dependencies and helps determine the critical path — the sequence of tasks that determines the minimum project duration.

2. Given Tasks and Durations

ID	Task	Duration (days)	Predecessor
A	Frontend MVP (Web UI)	14	—
B	Backend MVP (APIs & Auth)	17	A
C	Text Sentiment Integration	8	B
D	Voice Emotion Classifier Setup	8	B
E	Facial Emotion Recognition Module	8	B
F	Speech-to-Text Integration	7	D
G	Therapeutic Chatbot Integration	8	C, D, E, F
H	Notion API Integration	7	G
I	Security & Privacy Layer	8	H
J	Deployment + Testing (MVP)	5	I

3. PERT Terminologies

Term	Meaning
ES (Earliest Start)	The earliest time an activity can begin.
EF (Earliest Finish)	ES + Duration
LS (Latest Start)	The latest time an activity can begin without delaying the project.
LF (Latest Finish)	LS + Duration
Slack (Float)	Difference between Latest and Earliest times → Slack = LS – ES = LF – EF.

Critical Path	Path with zero slack; defines total project duration.

4. Forward Pass Calculation (ES & EF)

Formula:

- ES (first task) = 0
- EF = ES + Duration
- For dependent tasks → ES = Max(EF of all predecessors)

ID	ES	EF	Calculation
A	0	14	0 + 14
B	14	31	14 + 17
C	31	39	31 + 8
D	31	39	31 + 8
E	31	39	31 + 8
F	39	46	39 + 7
G	46	54	Max (39,39,39,46) + 8
H	54	61	54 + 7
I	61	69	61 + 8
J	69	74	69 + 5

Project Duration = 74 days

5. Backward Pass Calculation (LS & LF)

Formula:

- LF (last task) = Project Duration = 74
- LS = LF – Duration
- For predecessors → LF = Min(LS of all successors)

ID	LF	LS	Calculation
J	74	69	74 – 5
I	69	61	69 – 8

H	61	54	61 – 7
G	54	46	54 – 8
F	46	39	46 – 7
E	46	38	Min (46) – 8
D	39	31	39 – 8
C	46	38	Min (46) – 8
B	31	14	Min (31,31,31,31) – 17
A	14	0	14 – 14

6. Slack Calculation

ID	**Slack = LS - ES**
A	0
B	0
C	7
D	0
E	7
F	0
G	0
H	0
I	0
J	0

7. Critical Path Identification

Activities with Slack = 0 form the Critical Path:

A → B → D → F → G → H → I → J

🔑Critical Path Duration = 74 days

ID	Task (short)	Dur (days)	Pred	ES (day)	EF (day)	LS (day)	LF (day)	Slack (days)
A	Frontend MVP (Web UI)	14	—	0	14	0	14	0
B	Backend MVP (APIs & Auth)	17	A	14	31	14	31	0
C	Text Sentiment Integration	8	B	31	39	38	46	7
D	Voice Emotion Classifier	8	B	31	39	31	39	0
E	Facial Emotion Recognition	8	B	31	39	38	46	7
F	Speech-to-Text Integration	7	D	39	46	39	46	0
G	Therapeutic Chatbot Integration	8	C, D, E, F	46	54	46	54	0
H	Notion API Integration	7	G	54	61	54	61	0
I	Security & Privacy Layer	8	H	61	69	61	69	0
J	Deployment + Testing (MVP)	5	I	69	74	69	74	0