Bilkent University
Department of Computer Engineering

Senior Design Project

Team T2507

SCUBAMIND - MENTAL HEALTH

Analysis and Requirement Report

22103231 Merve Güleç merve.gulec@ug.bilkent.edu.tr

22103775 Metin Çalışkan metin.caliskan@ug.bilkent.edu.tr

22101998 Muhammed Fatih Başal fatih.basal@ug.bilkent.edu.tr

22103416 Murathan Işık murathan.isik@ug.bilkent.edu.tr

22103406 Yiğit Koşum yigit.kosum@ug.bilkent.edu.tr

Hamdi Dibeklioğlu

İlker Burak Kurt and Mert Bıçakçı

Contents

1. Introduction

Depression has evolved into a pervasive global health crisis, significantly impacting the quality of life and productivity of millions of individuals. According to the World Health Organization (WHO), depression is a leading cause of disability worldwide, yet the diagnostic and therapeutic landscape remains fraught with challenges [1]. Traditional clinical psychiatry relies heavily on episodic consultations and self-reported questionnaires. While valuable, these methods suffer from inherent limitations: they depend on the patient's memory (recall bias), are subject to social desirability bias, and only capture a "snapshot" of the patient's mental state, failing to reflect the fluctuating nature of mood disorders in daily life [2].

ScubaMind is designed to bridge this gap by introducing a mobile-based, privacy-preserving, and continuous mental health monitoring system. Unlike traditional methods that require active user effort, ScubaMind leverages the concept of "Digital Phenotyping." By utilizing the sensors embedded in modern smartphones and wearable devices, the application collects behavioral and physiological data in the background without disrupting the user's routine.

By integrating advanced technologies, including Native Android Services for robust data collection, Speech Signal Processing for vocal biomarker analysis, and an Embedded AI Core for on-device inference, ScubaMind analyzes complex behavioral patterns to detect early signs of depression risks. Crucially, the system operates on a "Local-First" architecture, ensuring user privacy by processing sensitive data entirely on the device.

This report will begin with a section titled "Current System," which summarizes existing digital approaches used for monitoring depression and their limitations. Following this, the "Proposed System" section will present the overall structure of ScubaMind, Functional and Nonfunctional Requirements, Pseudo Requirements, and various System Models (scenarios, use cases, class and dynamic models, and UI mock-ups). The "Other Analysis Elements" section will address design constraints, standards, risks, project plan, teamwork, ethical responsibilities, and learning strategies. The report will conclude with a Glossary and References section.

2. Current System

There are currently two main "existing system" approaches in the problem area targeted by ScubaMind: (A) traditional clinical monitoring methods, (B) digital mental health applications.

2.1 Traditional Clinical Workflow

In current clinical practice, depression monitoring is mostly carried out through periodic interviews and self-report scales. In this approach, “measurement” is usually based on the user's/person's current statement and struggles to represent the continuity of daily life. Therefore, increased risk, behavioral changes, or disruptions to biological rhythms such as sleep/activity may be detected with a delay unless the user actively notices and expresses them. Furthermore, effects such as recall bias and social desirability bias can lead to underreporting or misremembering of symptoms [2]. As emphasized by the WHO, given the prevalence of depression, there is an increasing need for less strenuous and continuous monitoring approaches [1].

2.2 Digital Mental Health Applications (Digital Ecosystem)

The common digital solutions on the market are mostly divided into three categories:

2.2.1 Manual mood tracking/journalizing applications

For example, Daylio expects the user to manually select a mood each day and mark their day with short tags [3]. The strength of this approach is its simplicity; however, since the system is entirely dependent on active input, no data is generated when the user is not using the application. Therefore, "continuity" and "passive capture" are limited.

2.2.2 Mindfulness/meditation content platforms

Applications such as Headspace and Calm provide support through content such as meditation, breathing exercises, and sleep stories [4, 7]. These systems can be useful for stress management and relaxation; however, they often do not offer a monitoring & risk assessment layer that measures the individual's behavioral/biometric changes. In other words, there is content, but there is usually no personalized data-based early warning mechanism.

2.2.3 Chatbot-based support applications

Products like Wysa and Youper aim to support the user with AI-driven chat and guidance [5, 6]. This approach is valuable in that it opens up a space for expression for the user; however, evaluation often remains dependent on “active use” as there is no passive sensing (sleep, movement, usage pattern) or wearable integration. In addition, clinical-level interpretation of chat outputs requires strong safety frameworks and clear boundaries.

2.3 Key shortcomings of the current system (Gap analysis)

The common limitation of these three approaches is that it is often impossible to capture changes in the user's real-life routine continuously and with multiple sources (multimodal). That is, when the user is not open, the system's capacity to understand them "in the background" decreases. In addition, due to privacy concerns, many solutions are more inclined to move "raw data" to the cloud; this can undermine user trust in a sensitive area like mental health.

The literature has shown that significant relationships can be established between smartphone mobility traces and indicators of depressive states; for example, changes in mobility patterns and routine disruptions can correlate with depressive symptoms [8]. This is an important indicator explaining why "current system" solutions are insufficient in practice: simply providing content or simply receiving manual mood input cannot capture these kinds of behavioral signals.

2.4 Differences between ScubaMind and the existing system

In this context, ScubaMind aims to combine the elements offered in a fragmented way by existing systems under a single umbrella, with a Local-First and privacy-preserving approach:

- Ensuring continuity by collecting passive behavioral data (usage pattern, activity, etc.) without relying on manual input,
- Strengthening the context by adding rhythm signals such as sleep/HRV with wearable integration,
- Positioning the output as a risk insight for user awareness, rather than a "diagnosis,"
- Reducing privacy risks by processing the data on the device as much as possible.

Therefore, while solutions remain either content-focused (meditation platform), manual tracking-focused (mood journal), or chat-focused (chatbot); ScubaMind's goal is to transform them into a data-driven and multimodal monitoring-recommendation cycle. [3 – 8]

3. Proposed System

This section describes the general structure and expected behavior of the proposed system for the ScubaMind project. The proposed solution is designed to overcome the limitations observed in existing systems, employing a mobile-based, privacy-preserving, and on-device intelligence-focused approach. The system aims to passively analyze daily life data according to Digital Phenotyping principles, while requiring minimal active user interaction.

ScubaMind is positioned not as a clinical diagnostic tool for depression risk assessment, but as a decision-support system that raises user awareness, monitors behavioral changes over the long term, and encourages seeking professional support when needed. Accordingly, the system architecture is designed based on privacy, energy efficiency, and modularity.

3.1 Overview

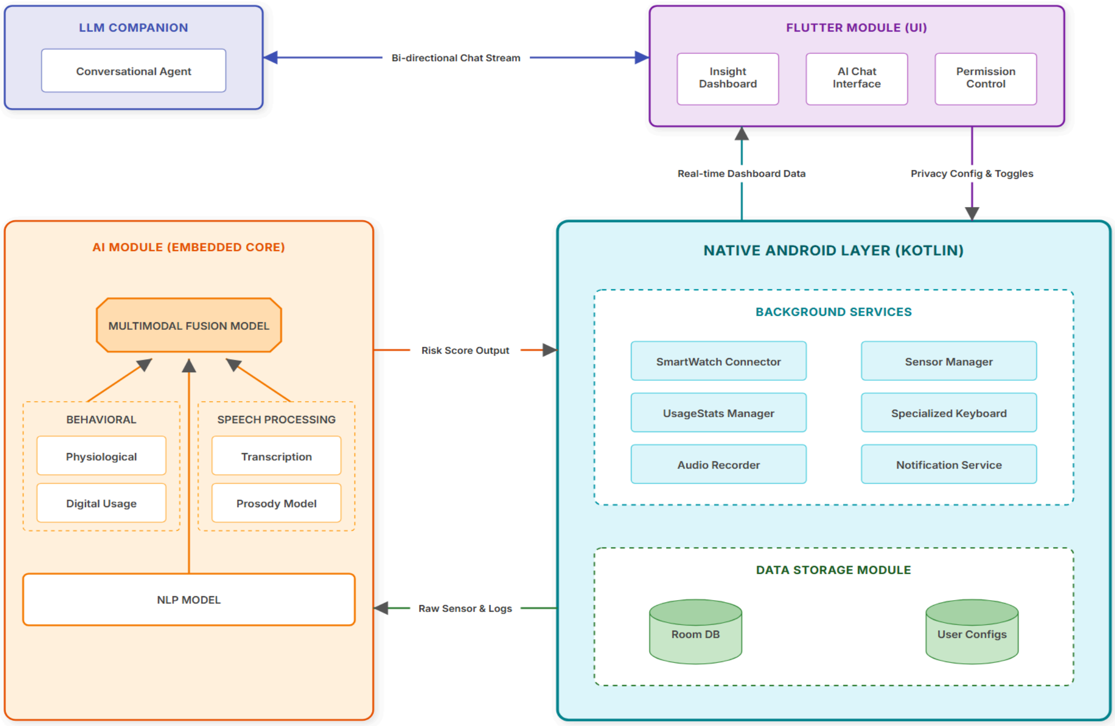


Figure 1. High-level system architecture of the proposed ScubaMind system.

The high-level architecture of the proposed system is shown in Figure 1. ScubaMind adopts a layered architecture approach and has a structure consisting of four main components:

- Flutter Module (UI Layer)
- Native Android Layer (Data & Storage)
- AI Module (Embedded Core)
- LLM Companion

Data flow and responsibility sharing between these components ensure that the system adheres to the on-device processing principle while providing the user with a transparent and controllable experience.

The Flutter Module is the layer where the user directly interacts with the system. This layer includes the Insight Dashboard, AI Chat Interface, and Permission Control components. It allows the user to be presented with the generated analysis outputs through understandable graphs and summary indicators. The user can also manage data collection permissions and perform data sovereignty operations, such as deleting local data, through this layer.

The Native Android Layer forms the operational backbone of the system. Sensor data, usage statistics, and physiological measurements from wearable devices are collected through background services provided by the Android operating system. All data is stored encrypted on the device and transmitted to the AI Module for analysis. This layer guarantees that data collection processes are carried out in accordance with user settings and privacy preferences.

The AI Module is ScubaMind's core analysis component. This module combines behavioral (digital usage, mobility), physiological (heart rate, HRV, sleep), and optionally text/voice-based features to generate a unique Risk Score using a multimodal fusion approach. All inferences are performed via on-device inference; raw data is not transferred outside the device.

The LLM Companion is an optional support component of the system. If the user allows it, this Large Language Model-based component provides a supportive and secure chat experience with the user. The LLM Companion does not replace the analysis module; it only aims to provide guidance and awareness through high-level state information.

The interaction and data flow between these components are designed with a user privacy-centric approach and prioritize local processing, as shown in Figure 1.

3.2 Functional Requirements

This section defines the core functions that the ScubaMind system must perform, from user input to output, and describes the behaviors and fundamental processes the system should exhibit. These requirements are defined in a user-centric manner and are consistent with the system's on-device, privacy-preserving architecture.

3.2.1. User Management

- A user-specific profile screen should be available, and users should be able to edit their personal information.
- After downloading the app, users should go through a brief onboarding process and be able to select their personal preferences (notification preferences, data sharing permission, wearable device connection, etc.).

3.2.2. Passive Behavioral & Sensor Data Collection

- The app should support the collection of daily user behavior without active participation, only with the necessary permissions.
- The system should automatically collect phone behavior data (screen on/off, screen time, app usage frequency).
- Keyboard usage characteristics (typing speed, interaction intensity, etc.) should only be analyzed in an anonymized form.
- Phone sensor data (accelerometer, gyroscope, step count, movement intensity) should be recorded regularly.
- The app should be able to integrate heart rate, sleep patterns, and activity data from wearable devices.
- All data collection processes should be conducted on-device and encrypted, in accordance with Privacy by Design principles.
- Users should be able to manage their data collection permissions.

3.2.3. On-Device Data Analysis & Mood Prediction

- The system should analyze passive data collected using machine learning models running on the device to provide meaningful feedback to the user.
- On-device ML models should analyze the user's daily behavioral patterns at regular intervals.
- The user should be presented with a mood prediction (categories such as "Energetic," "Low," "Stressed," and "Balanced").
- The system should be able to assess depression risk and inform the user of low, medium, or high risk levels.
- All analyses should be performed solely on the device.
- Analyses should be optimized to maintain energy efficiency.

3.2.4. Personalized Recommendations and Support System

- The app should provide personalized support and behavioral recommendations to the user based on analysis results.
- Daily recommendations (such as reminders for short walks, breathing exercises, suggestions for taking breaks, etc.) should be created specifically for the user.
- Recommendations should be dynamically updated based on the user's behavior and mood changes throughout the day.
- In certain situations, and with user permission, the system can send the user "supportive notifications" (e.g., "You haven't been very active today. A 30-minute walk would be beneficial").
- All recommendations should be based on scientific sources and verified content.

- The LLM agent integrated should guide the user correctly in order to not cause any discomfort or incompatibility with app’s aims.

3.2.5. Reporting and Visualization

- The app should present its data with clear and effective graphs to help the user better understand themselves.
- The user's daily, weekly, and monthly behavioral and mood trends should be displayed graphically.
- User activities, sleep patterns, and mood changes should be displayed in a timeline format.
- The app should generate a summary PDF report that the user can share with their doctor upon request.
- Reports should only be stored on the user's device or shared with the user's express consent.

3.2.6. Notification Management

- The system should send supportive notifications to the user at the right time.
- The system should send intelligent notifications based on recommendations, activity goals, and user behavior.
- Notifications should be optimized to avoid unnecessary alerts (not create stress).

3.2.7. Security and Privacy Requirements

- The app must protect users' personal and behavioral data in accordance with the highest security standards.
- All data must be stored on the device.
- User data must never be shared with third parties.

3.3 Nonfunctional Requirements

This section defines the quality requirements for the ScubaMind system, not just what it does, but how it does it. Non-functional requirements encompass aspects such as user experience, performance, reliability, ease of maintenance, scalability, and safety. These requirements are established to ensure the system remains sustainable, safe, and usable in the long term.

3.3.1 Usability

- The application should offer a simple and calming UI design that does not create stress or cognitive load on the user.

- The onboarding process should be intuitive, guiding the user step-by-step.
- Basic user interactions should be achievable with a maximum of 4–5 taps.
- Graphics, dashboard components, and reports should use a clear and easy-to-understand visualization language.
- The application's color palette should be consistent with soft-tone themes recommended for mental health applications.

3.3.2 Reliability

- The system should be able to process the collected data accurately and consistently.
- Data loss should not occur while background services are running.
- The application should be able to maintain its basic functions even without a network connection (offline mode).
- System errors should be managed in a way that does not impair the user experience.

3.3.3 Performance

- On-device machine learning models should operate with low energy consumption and should not negatively impact battery performance.
- Response latency in user interface interactions should be a maximum of 300 ms. Graphics and animations should render smoothly even on low-end devices.
- Data processing and analysis processes should be carried out using an optimized batch processing approach.
- The system should be able to operate without requiring a constant internet connection.

3.3.4 Supportability & Maintainability

- The application should be designed to run smoothly on different Android versions.
- The code structure should adhere to modular architecture principles to facilitate the addition of new features.
- Third-party dependencies should be regularly monitored and updated.
- System components should be decomposition-based to facilitate debugging and maintenance processes

.

3.3.5 Scalability

- The system should be able to operate without experiencing performance degradation as the number of users increases.
- Analysis and data processing components should be extensible to support new data types that will be added in the future.
- Data access and analysis performance should be maintained as the user's long-term behavioral history increases.
- Storage and analysis structures should be designed with longitudinal data growth in mind.

3.3.6 Security & Privacy

- All user data must be stored on on-device encrypted storage.
- User data must not be shared with third parties under any circumstances.
- The system must operate in accordance with the principles of privacy by design and data minimization.
- Data access policies and permissions must be presented to the user in a clear and transparent manner.

3.4 Pseudo Requirements

This section contains design assumptions and guiding constraints for the ScubaMind system, which are not finalized functional or non-functional requirements but are expected to be clarified in later development phases. These pseudo-requirements will be reviewed and updated as the system evolves.

1. The system should assume that there may be significant differences in the daily routines and behavioral patterns of different users and should be able to adapt its analysis processes according to the concept of a personal baseline.

2. It should be accepted that users may not allow access to all data sources (e.g., wearable data, keyboard features, audio journaling); the system should continue to produce meaningful outputs even in cases of partial data availability.

3. It should be assumed that depression risk analysis outputs may be mistakenly interpreted as clinical diagnoses by the user, and the interface design should guide to mitigate this risk.

4. The system should be designed to be expandable in the long term with different sensor types and new data modalities (e.g., new wearable metrics); it is assumed that the current architecture supports this expansion.

5. It should be accepted that on-device machine learning models may perform differently from device to device; It should be assumed that graceful degradation will be provided on low-end devices.

6. It should be considered that users may use the application irregularly or not open it for a long time; it should be assumed that the system can maintain its basic functions with background operation.
7. It should be accepted that the LLM Companion component may not always be active or may not be used at all by some users; it should be assumed that the core functions of the system are not dependent on this component.
8. It should be assumed that users may have a high level of sensitivity regarding data privacy; therefore, it should be accepted that privacy transparency and permission control mechanisms directly affect user trust.
9. It should be assumed that user behavior may naturally change with long-term use (e.g., lifestyle changes); the system should have the flexibility to treat these changes as behavioral shifts rather than anomalies.

3.5 System Models

3.5.1 Scenarios

3.5.1.1 Initial Setup and Privacy Configuration

- **Use Case Name:** Initial Setup and Privacy Configuration
- **Actor:** User
- **Entry Condition:** User installs and launches the application for the first time.
- **Exit Condition:** User profile is created, permissions are granted, and the system begins establishing a personal baseline.
- **Flow of Events:**
    - The user starts the onboarding process, which provides educational content on Digital Phenotyping and the "Local-First" architecture.
    - User creates a local-only profile and configures personal preferences via the Flutter Module.
    - The Permission Control module requests access to background sensors (GPS, accelerometer), usage stats, and Health Connect APIs.
    - User toggles individual data streams (e.g., enabling keyboard tracking but disabling GPS).
    - The system confirms informed consent and initializes the encrypted Room Database for local storage.
- **Alternative Flows:**
    - **Permission Refusal:** If critical permissions are denied, the system informs the user that specific risk markers (e.g., psychomotor retardation via steps) cannot be calculated, but the app continues with remaining sensors.

3.5.1.2 Passive Digital Phenotyping

- **Use Case Name:** Passive Digital Phenotyping
- **Actor:** Native Android Layer, Hardware Sensors, Wearable Devices
- **Entry Condition:** Background services are active following user consent.
- **Exit Condition:** Raw behavioral signals are securely persisted in the local Room Database.
- **Flow of Events:**
    1. **Sub-case: Mobility Tracking:** The Sensor Manager polls GPS variance and step counts to calculate the "Routine Index" and "Radius of Gyration".
    2. **Sub-case: Digital Interaction:** The UsageStats Manager logs app launch frequency and screen time to detect social withdrawal patterns.
    3. **Sub-case: Physiological Sync:** The SmartWatch Connector retrieves HRV, heart rate, and sleep cycles via IEEE 11073 standard protocols.
    4. **Sub-case: Keyboard Dynamics:** The Specialized Keyboard captures typing speed and interaction intensity in volatile memory without logging raw text.
    5. All data is batched, encrypted, and stored in the Room Database.

3.5.1.3 Context-Aware Daily Check-in

- **Use Case Name:** Context-Aware Daily Check-in
- **Actor:** User, LLM Companion, AI Module
- **Entry Condition:** The system triggers an intelligent notification based on user activity patterns.
- **Exit Condition:** A conversational session is completed, providing prosodic and semantic data for the AI analysis core.
- **Flow of Events:**
    1. **Sub-case: Intelligent Trigger:** The Notification Service sends a reminder during a low-activity period to minimize cognitive load.
    2. **Sub-case: Context Feeding:** The AI Module summarizes the day’s passive markers (e.g., "poor sleep," "low social activity") and feeds this into the LLM prompt to personalize the opening.
    3. **Sub-case: Adaptive Dialogue:** The LLM opens with a context-aware prompt: "I noticed your sleep was a bit restless last night. How are you holding up?".
    4. The user speaks into the AI Chat Interface; the Audio Recorder captures samples for the Speech Processing Module.
    5. The LLM provides empathetic responses while the system extracts vocal biomarkers and sentiment embeddings.

3.5.1.4 Multimodal Fusion and Dashboard Reporting

- **Use Case Name:** Multimodal Fusion and Dashboard Reporting
- **Actor:** AI Module, User
- **Entry Condition:** New batches of passive sensor data and active conversational data are available.
- **Exit Condition:** The Insight Dashboard is updated with a PHQ-9 aligned Risk Score.
- **Flow of Events:**
    1. The NLP Module evaluates user transcripts, while the Prosody Model analyzes acoustic properties like tone and speech rate.
    2. The Multimodal Fusion Model synthesizes these features with digital usage habits and physiological metrics (HRV/Sleep).
    3. The system calculates a continuous Risk Score, comparing current data against the user's historical personal baseline.
    4. The Risk Score is stored in the Room DB and pushed to the Flutter UI.
    5. The Insight Dashboard renders interactive charts and gauge indicators for the user.

3.5.1.5 Behavioral Intervention and Emergency Escalation

- **Use Case Name:** Behavioral Intervention and Emergency Escalation
- **Actor:** User, ScubaMind System
- **Entry Condition:** A negative behavioral trend or high risk score is detected.
- **Exit Condition:** User receives guidance or is directed to professional support resources.
- **Flow of Events:**
    1. **Sub-case: Personalized Recommendation:** The app suggests a mindfulness exercise or physical activity based on the detected anomaly (e.g., sedentary behavior).
    2. **Sub-case: Doctor's Report:** User selects "Generate Report"; the system compiles longitudinal trends into a secure PDF summary for medical sharing.
    3. **Sub-case: Emergency Escalation:** If suicidal ideation or critical risk is detected in speech/text, the app suspends the LLM and displays emergency helpline resources immediately.

3.5.1.6 Data Sovereignty and Privacy Management

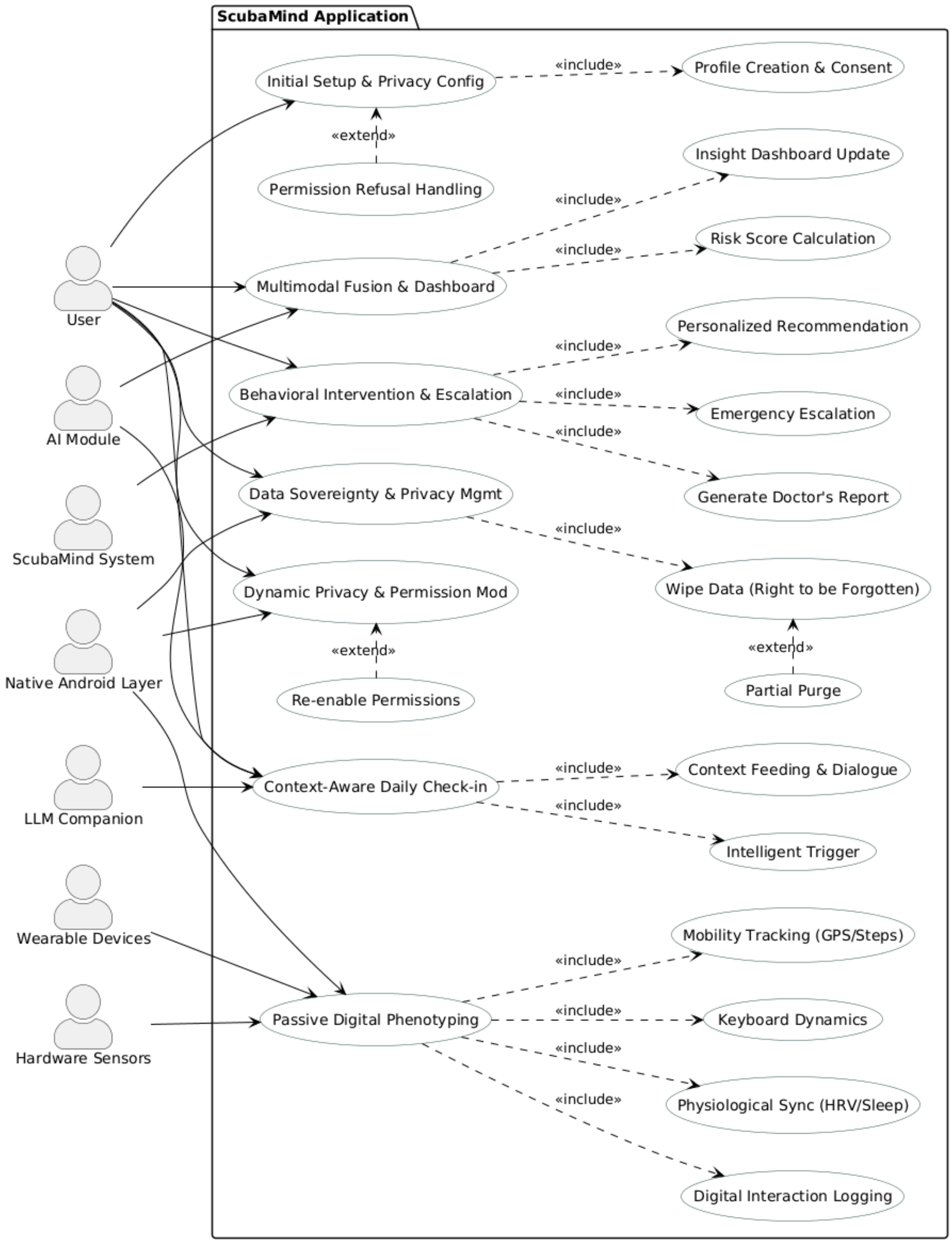
- **Use Case Name**: Data Sovereignty and Privacy Management
- **Actor**: User, Native Android Layer
- **Entry Condition**: User accesses the "Permission Control" or "Settings" panel.
- **Exit Condition**: Sensitive data is purged or data collection settings are updated locally.

- **Flow of Events**:
    - User opens the Permission Control module in the Flutter UI to review active data streams.
    - User selects the "Right to be Forgotten" option to purge all locally stored information.
    - The Flutter Module sends a wipe command to the Native Android Layer.
    - The Native Layer executes a permanent deletion of the Room Database and local user configurations.
    - The system confirms the purge and resets the application to the initial onboarding state.
- **Alternative Flows**:
    - **Partial Purge**: User selects specific date ranges or categories (e.g., "Delete only Keyboard history") to be deleted while keeping other data intact.

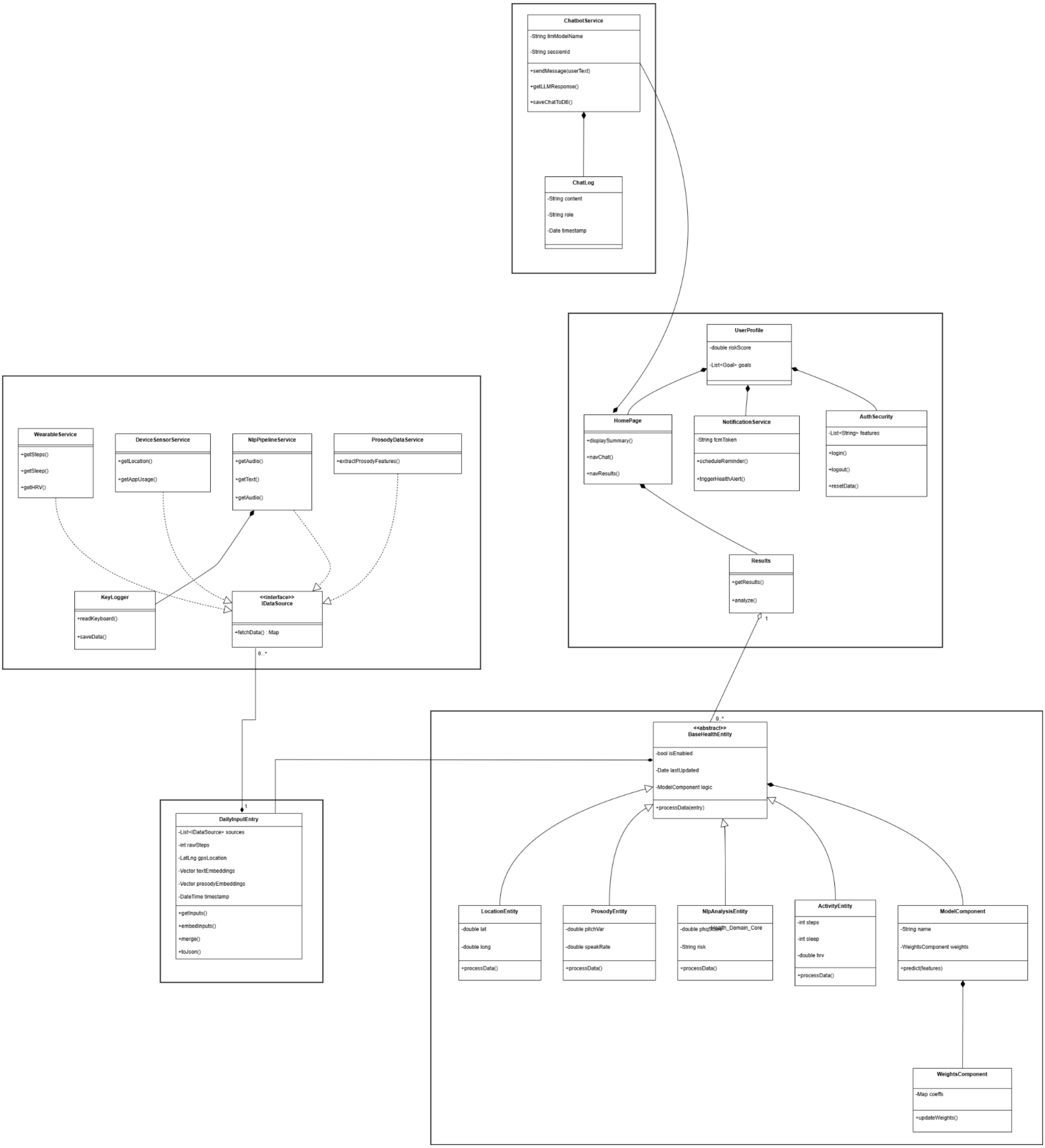
3.5.1.7 Dynamic Privacy and Permission Modification

- **Use Case Name**: Dynamic Privacy and Permission Modification
- **Actor**: User, Native Android Layer
- **Entry Condition**: User navigates to the "Permission Control" settings in the app.
- **Exit Condition**: Specific data collection streams are modified, and background services are updated accordingly.
- **Flow of Events**:
    - User opens the Permission Control settings panel via the Flutter Module.
    - User reviews the list of active sensors and data streams (e.g., Keyboard tracking, GPS variance, Audio journaling).
    - User toggles a specific sensor "Off" to stop its data collection (e.g., disabling the Specialized Keyboard).
    - The Flutter Module sends an explicit command to the Native Android Layer to update service states.
    - The Sensor Manager or the specific Background Service immediately halts the data polling for the disabled sensor.
    - The system confirms the setting change and informs the user about the impact on future Risk Score accuracy.
- **Alternative Flows**:
    - **Re-enabling Permissions**: User toggles a previously disabled sensor "On"; the app requests the necessary Android system-level permissions if they were revoked, and resumes collection upon approval.

3.5.2 Use Case Model



3.5.3 Object and Class Model



3.5.4 Dynamic Models

In the ScubaMind project, dynamic behaviors occur in processes such as background data collection, periodic on-device inference, LLM Companion interaction, and emergency escalation. However, accurately modeling these processes requires a clear definition of the exact responsibilities, interfaces, and operation frequencies of the system components.

At this stage, system scenarios and interaction flows have been defined conceptually within the scope of the project. These definitions form the basis for being transformed into more detailed and time-dependent UML diagrams in the later design phase. Therefore, Dynamic Models will be presented in the Detailed Design Report when the system design is more mature and the interactions and timing between components are clarified.

3.5.5 User Interface - Navigational Paths and Screen Mock-ups

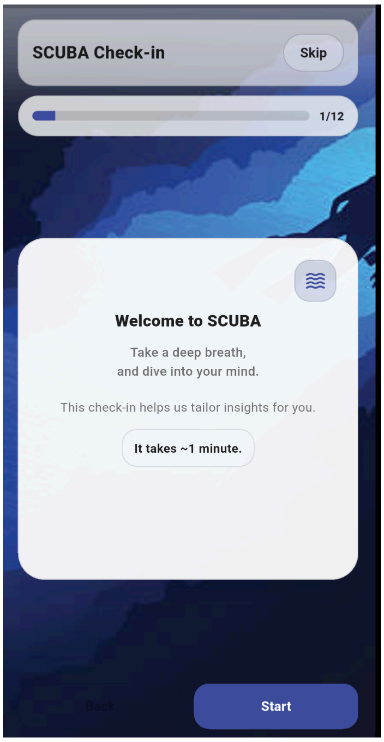


Fig.1 Onboarding page 1

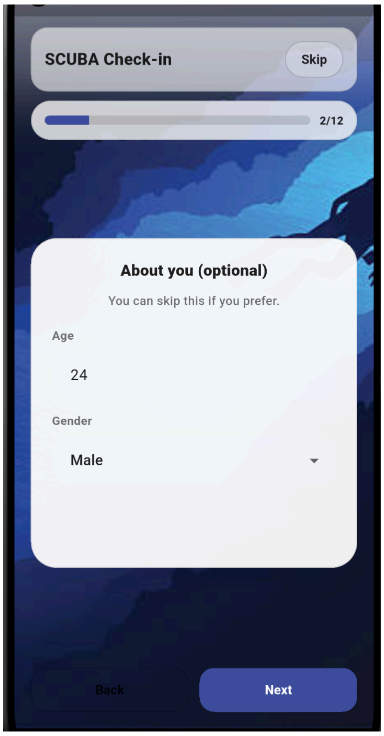


Fig.2 Onboarding page 2

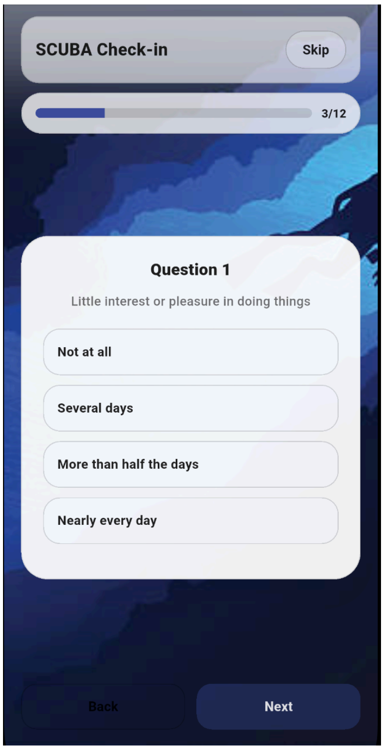


Fig.3 Onboarding page 3

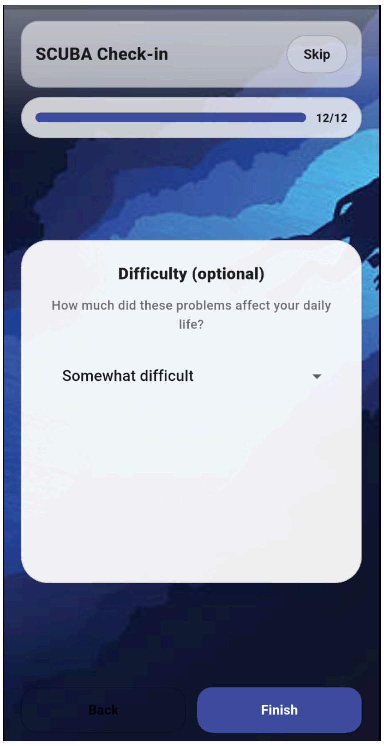


Fig.4 Onboarding page 4

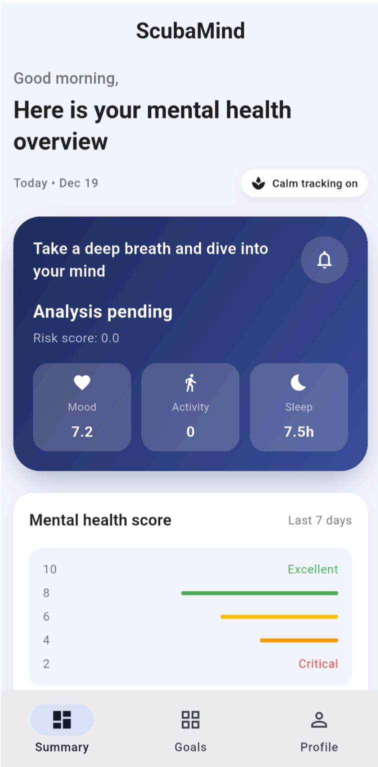


Fig.5 Homepage 1

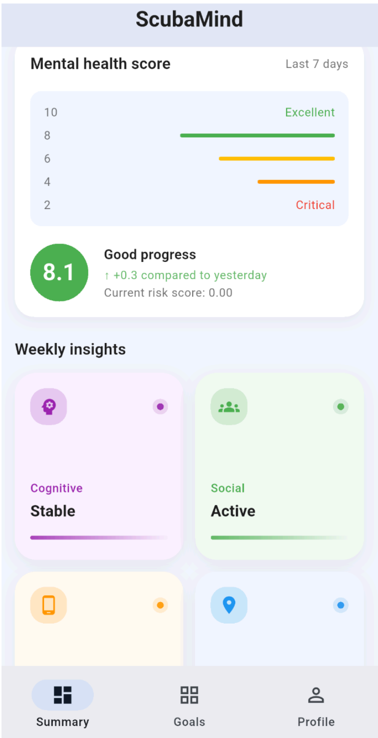


Fig.6 Homepage 2

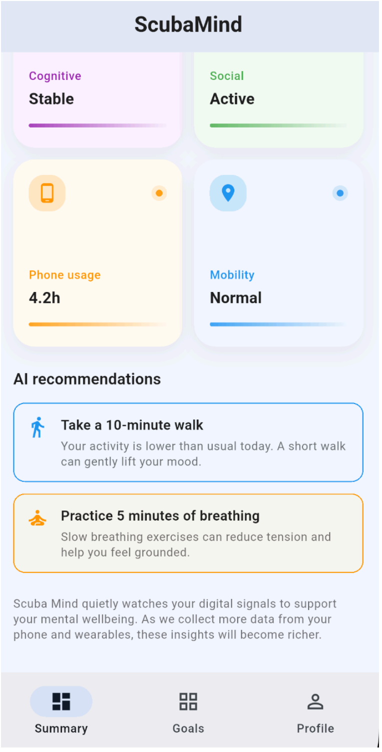


Fig.7 Homepage 3

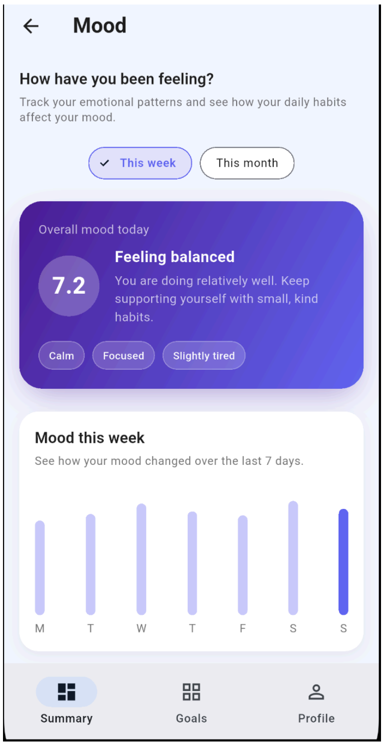


Fig.8 Mood Page 1

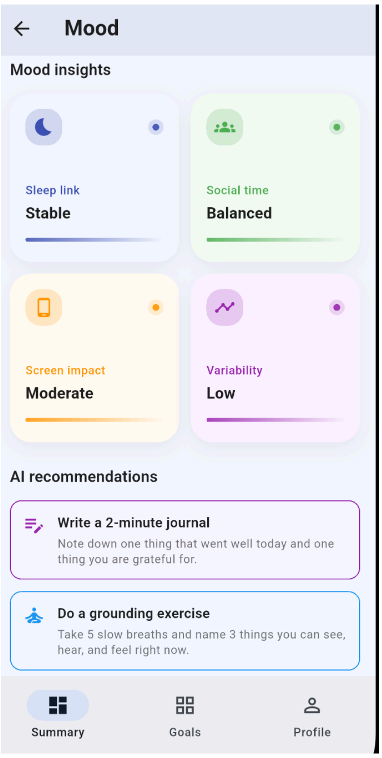


Fig.9 Mood Page 2

Fig.10 Activity Page 1

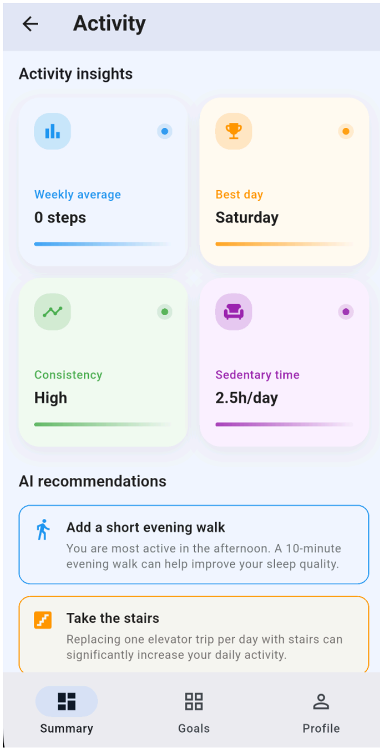


Fig.11 Activity Page 2

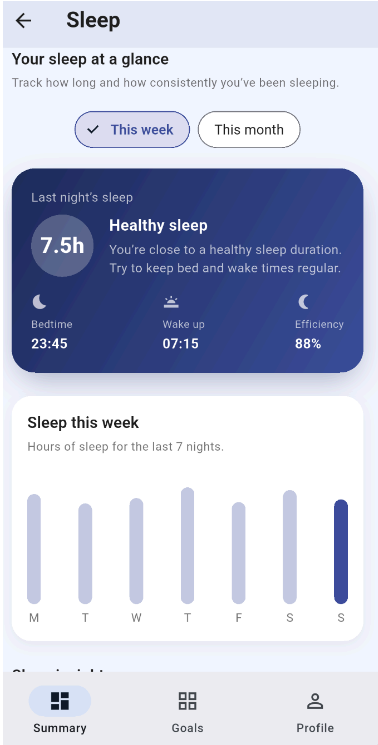


Fig.12 Sleep Page 1

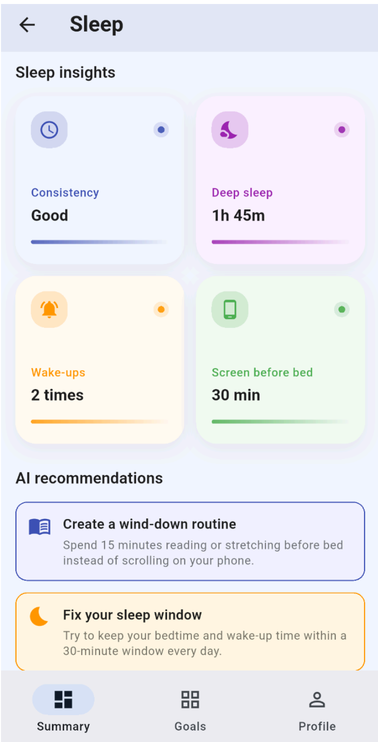


Fig.13 Sleep Page 2

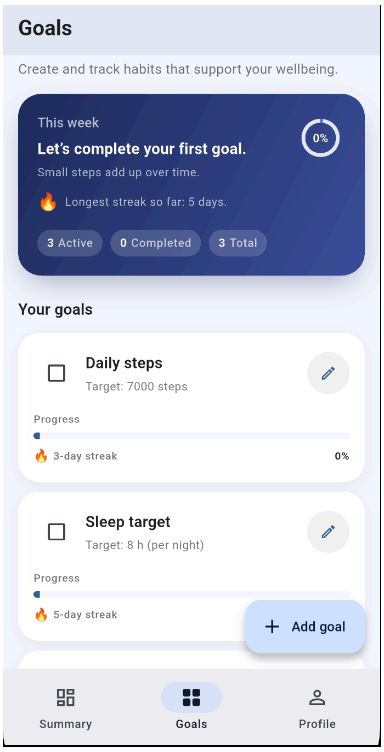


Fig.14 Goals Page 1

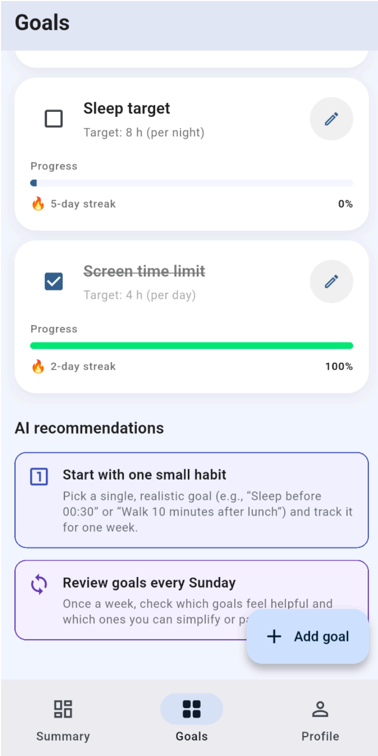


Fig.15 Goals Page 2

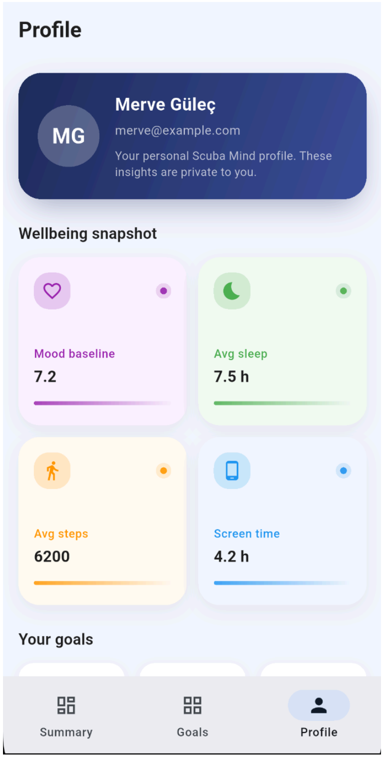


Fig.16 Profile Page 1

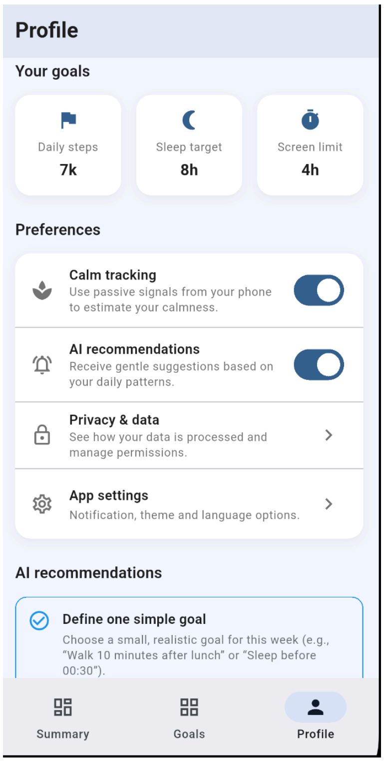


Fig. 17 Profile Page 2

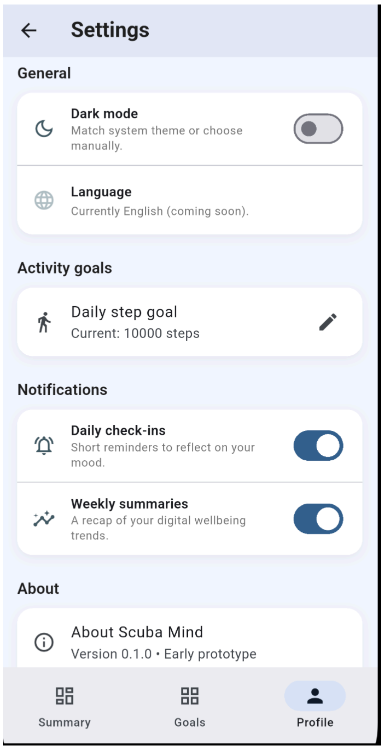
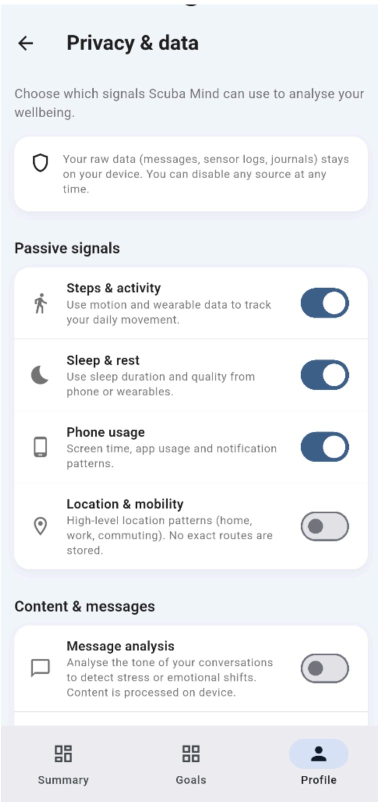


Fig.18 Settings Page



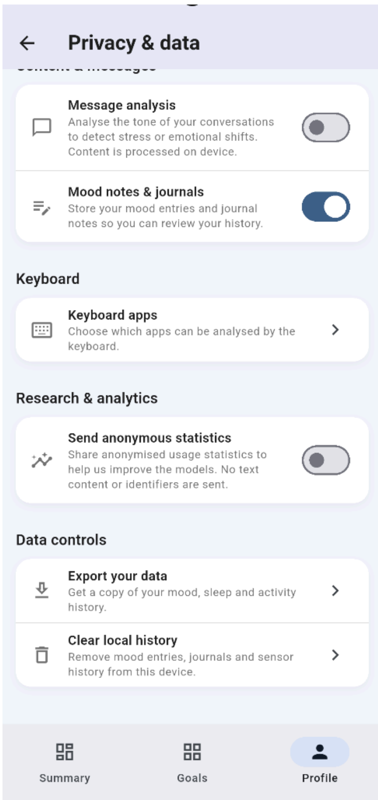
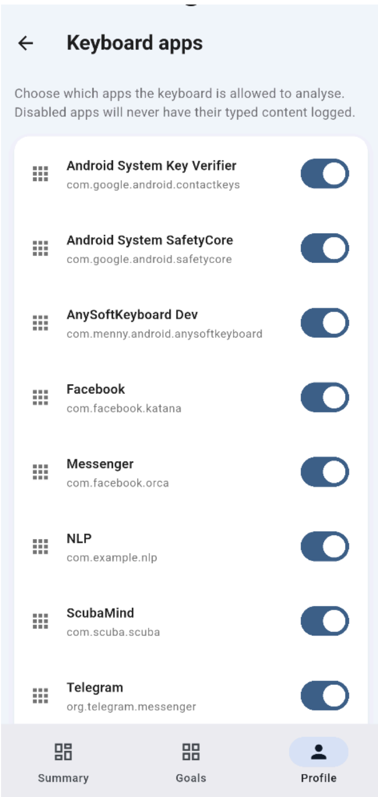


Fig. 19 Privacy & Data Page Fig. 20 Privacy & Data Page 2



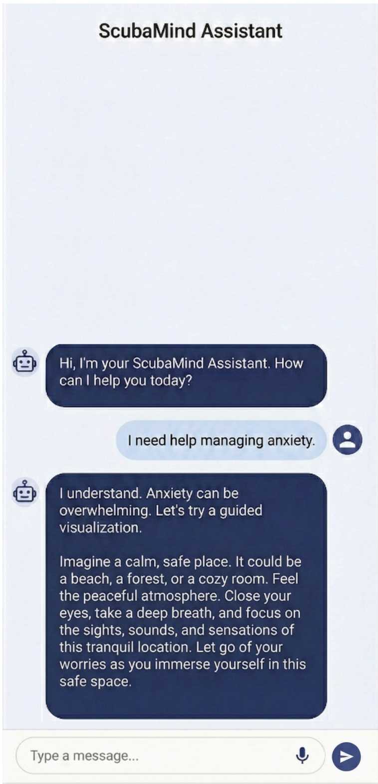


Fig. 21 Keyboard Apps Page Fig. 22 Chatbot Page

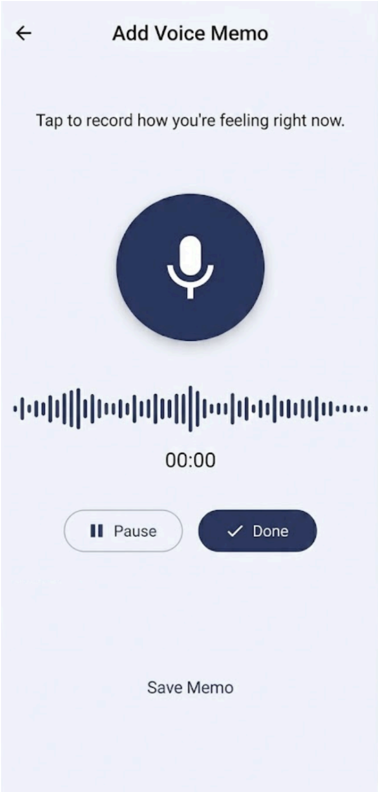


Fig. 23 Voice Memo Page

4. Other Analysis Elements

4.1. Consideration of Various Factors in Engineering Design

4.1.1 Constraints

The design and development of ScubaMind are shaped by rigorous constraints derived from its core philosophy of being a privacy-preserving, on-device mental health application. These constraints ensure that the system remains secure, efficient, and ethically sound while operating within the limitations of mobile hardware.

4.1.1.1. Implementation and Constructability Constraints

- **Local-First Architecture:** The most significant constraint is the requirement to process sensitive data (raw audio, keystrokes, sensor logs) exclusively on the user's device. This prohibits the offloading of heavy computational tasks to cloud servers, necessitating highly optimized on-device inference engines.
- **Mobile Hardware Limitations:** The system relies on AI models to perform complex multimodal analysis. These models must operate within the restricted RAM, CPU, and NPU capacities of standard smartphones without causing system lag, battery drain or overheating.

- **Platform Dependency:** Due to the need for deep integration with background services, hardware sensors, and health APIs, the Data Retrieval Module must be implemented using Native Android (Kotlin). This creates a constraint where the core data collection features are currently exclusive to the Android ecosystem.
- **Network Independence:** The application's fundamental features, including depression risk analysis and mood tracking, must remain functional even without an active internet connection. Cloud dependency is strictly limited to the optional LLM Companion features.

4.1.1.2. Economic Constraints

- **LLM API Costs:** The "LLM Companion" module relies on external APIs to provide empathetic conversational support. Since these services charge per token, the implementation is constrained by the need to optimize prompt engineering and limit context windows to maintain financial sustainability for the project.
- **Development Budget:** As a senior design project, the development budget is minimal. This constrains the team to utilize open-source libraries (e.g., Whisper.cpp, ONNX Runtime), free-tier cloud services, and pre-trained models available on platforms like Hugging Face, thereby avoiding the need for expensive proprietary software.
- **Device Accessibility:** The requirement for on-device AI processing imposes a hardware entry barrier. The application requires smartphones with moderate-to-high processing power, potentially excluding users with older or low-budget devices.

4.1.1.3. Ethical and Legal Constraints

- **Privacy & Data Sovereignty:** Adhering to the "Privacy by Design" principle, the system is constrained to collect only the minimum amount of data necessary for analysis. The architecture must support the user's "Right to be Forgotten," enabling the immediate and permanent deletion of local data from the Room Database upon request.
- **Non-Diagnostic Nature:** Legally and ethically, ScubaMind cannot function as a medical diagnostic tool. The system is constrained to frame all outputs as "risk assessments" or "behavioral insights" rather than clinical diagnoses. The user interface must include mandatory disclaimers and avoid definitive medical terminology.
- **Crisis Management Protocols:** If the system detects critical risk indicators, it is ethically constrained to suspend standard analysis and immediately direct the user to professional emergency resources.
- **Regulatory Compliance:** The system must comply with **KVKK** (Law on Protection of Personal Data) in Turkey and **GDPR** globally. This requires

explicit, informed consent mechanisms for processing sensitive health data and strict encryption standards for local storage.

4.1.1.4. Sustainability and Environmental Constraints

- **Battery Efficiency:** Continuous background sensing poses a significant risk to battery life. The implementation is designed to utilize energy-efficient Android APIs and batch processing to prevent the application from draining the user's device.
- **Computational Efficiency:** To minimize the carbon footprint associated with mobile computing and device heating, AI models must be optimized using techniques such as quantization (INT8). Unnecessary inference cycles must be avoided when the user is inactive or the battery is low.

4.1.1.5. Usability and Accessibility Constraints

- **Cognitive Load Consideration:** Given that the target audience includes individuals potentially suffering from depression or anxiety, the UI/UX design is constrained to be simple, calming, and non-intrusive. Complex navigation or overwhelming data visualizations must be avoided to prevent inducing further stress.
- **Ease of Interaction:** Key features must be accessible within 4-5 taps to ensure that users with low motivation (a symptom of depression) can still utilize the application effectively.
- **Visual Design:** The color palette and design language must adhere to "soft-tone" themes suitable for mental health contexts, avoiding aggressive colors like bright reds that might trigger anxiety.

4.1.1.6. Maintainability and Supportability Constraints

- **Modular Architecture:** The system must be built with a modular design (separating UI, Data Layer, and AI Core) to allow for the independent update of AI models without requiring to update the full app.
- **Third-Party Dependencies:** The reliance on third-party SDKs (e.g., PyTorch Mobile) imposes a constraint where the system must be regularly updated to maintain compatibility with new Android versions and API deprecations.

Table 1: Effects of various factors

	Effect level	Effect
Public health	9	The system directly addresses a global health crisis by enabling continuous, passive monitoring for early detection of depression,

		potentially reducing the diagnostic gap.
Public safety	7	The design includes critical safety protocols; if high risk is detected, the system must intervene by directing users to emergency resources rather than acting as a doctor.
Public welfare	8	By democratizing access to mental health insights via smartphones, the project enhances general welfare and self-awareness without requiring expensive clinical visits initially.
Global factors	5	While depression is a global issue, the current NLP and Speech models are primarily optimized for English, requiring future adaptation for global applicability.
Cultural factors	3	Expressions of depression vary by culture; the AI models must be carefully selected and tested to avoid bias against specific dialects or cultural behavioral norms
Social factors	8	Privacy is the central social factor. The "Local-First" architecture is a direct response to the social stigma and fear of surveillance associated with mental health data.
Environmental factors	5	The system must be optimized for "Green Computing"; AI models are quantized and background services are scheduled to minimize battery drain and energy consumption.
Economic factors	4	The design is constrained by the cost of external LLM APIs and the need for open-source alternatives to keep the solution affordable and sustainable.

“

4.1.2 Standards

The development and documentation of ScubaMind strictly adhere to internationally recognized engineering standards to ensure interoperability, security, and architectural robustness.

4.1.2.1. ISO/IEC/IEEE 29148

This “Systems and Software Engineering - Requirements Engineering” standard suppresses the olded IEE 830 and provides the guidelines for elicitation, analysis and and specification of software requirements. In this report, the Functional and Non-Functional Requirements (Section 3.2 and 3.3) have been structured according to ISO/IEC/IEEE 29148 principles to ensure clarity, consistency, and verifiability of the system specifications.

4.1.2.2. UML 2.5.1 – Unified Modeling Language

UML 2.5.1 is utilized as the standard visual modeling language to architect the system’s complex components. The System Models section (Section 3.5), including Use Case, Class, and Object diagrams, follows UML 2.5.1 specifications to visually communicate the system's dynamic and static behaviors to developers and stakeholders effectively.

4.1.2.3. HL7 FHIR

HL7 FHIR (Fast Healthcare Intreoperability Resources) is the industry standard for the secure and reliable exchange of electronic health data. ScubaMind adopts the FHIR data structure resource definitions for storing the calculated Depression Risk Scores and longitudinal analysis logs within the local Room Database. This ensures that the data generated by the application is future-proof and can be easily integrated with other medical systems if the user chooses to share their reports.

4.1.2.4. IEEE 11073 – Health Informatics

This standard defines the protocols for data exchange between personal health devices and external computer systems. In ScubaMind, IEEE 11073 guidelines are utilized within the Data Retrieval Module to standardize the collection and normalization of physiological metrics such as Heart Rate Variability (HRV) and sleep patterns from various wearable devices.

4.1.2.5. ISO/IEC 27001 – Information Security Management

Due to the processing of highly sensitive mental health data (Special Category Data), ScubaMind adheres to the information security controls defined in ISO/IEC 27001. This standard dictates the "Privacy by Design" architecture, enforcing requirements such as the encryption of data at rest (local database encryption) and strict access controls, ensuring that user data confidentiality and integrity are maintained throughout the software life cycle.

4.2. Risks and Alternatives

The development of ScubaMind involves navigating significant technical challenges, particularly regarding the trade-off between AI model accuracy and mobile resource constraints, as well as critical safety concerns associated with mental health monitoring.

4.2.1 Performance Degradation due to Model Quantization and Compression

To enable Large Language Models (LLMs) and complex NLP architectures (like RoBERTa or BERT variants) to run directly on mobile devices, aggressive optimization techniques such as Quantization (e.g., FP32 to INT8) and Model Pruning are required. There is a significant risk that these compression methods may severely degrade the model's accuracy, causing it to lose the ability to detect subtle linguistic nuances associated with depression (e.g., implicit sadness or passive suicidal ideation). Furthermore, even optimized models might overwhelm the mobile CPU/NPU, leading to unacceptable application latency or excessive battery drain, rendering the app unusable for daily monitoring.

Plan B: If on-device mobile inference proves unfeasible due to accuracy loss or hardware limitations, we will implement a "Local Network Offloading" (PC Companion) architecture. In this scenario, the user installs a companion application on their personal computer (which possesses significantly higher processing power). When the mobile device and the PC are connected to the same Wi-Fi network, the heavy AI processing tasks will be securely offloaded to the PC via the Local Area Network (LAN). The mobile app will send the encrypted feature vectors to the PC, where the full-scale (non-quantized) model will process the data and return the results. This approach ensures that sensitive data never leaves the user's local network or touches the internet/cloud, preserving the "Local-First" privacy promise while utilizing superior hardware performance.

4.2.2 Training Limitations

The dataset used for training the depression detection models contains highly sensitive, confidential private health information. Due to strict data privacy agreements and ethical constraints, this dataset cannot be uploaded to cloud-based training environments (such as Google Colab Pro, AWS, or Azure). This forces the team to rely solely on local machines for training. There is a risk that our local hardware resources (GPU VRAM and Compute capability) may be insufficient to fine-tune large-scale transformer models effectively, leading to memory overflows (OOM errors) or prohibitively long training times that hinder the iterative development process.

Plan B: To mitigate the limitations of local hardware without violating data privacy, we will adopt two strategies. First, we will implement memory-efficient training techniques such as Gradient Accumulation and Low-Rank Adaptation (LoRA). These methods allow for fine-tuning large models on consumer-grade GPUs by freezing most weights and only training a small subset of parameters, drastically reducing VRAM usage. Second, if local resources are still insufficient, we will utilize Secure On-Premise High-Performance Computing (HPC) resources provided by the university laboratories or our project partners, which allow for training on powerful hardware within a controlled, offline environment that meets the data confidentiality requirements.

4.2.3 LLM "Hallucination" and Safety Risks

The integration of an LLM Companion brings the risk of "hallucinations," where the model might generate factually incorrect medical advice or fail to recognize a crisis situation (e.g., subtle expressions of self-harm) due to prompt injection or context limitations.

Plan B: We will implement a strict, deterministic "Rule-Based Safety Layer" that sits between the user and the LLM. Before any user input is sent to the LLM, it will be scanned for specific keywords and semantic patterns related to crisis scenarios using a local dictionary or a tiny classification model. If a threat is detected, the LLM generation will be bypassed entirely, and a pre-written, hard-coded "Crisis Resource Card" (containing helpline numbers) will be displayed. This ensures that safety mechanisms do not rely on the probabilistic nature of the LLM.

4.2.4 User Permission Denial (Data Sparsity)

The Multimodal Fusion model relies on having access to all data streams: Audio, Usage Stats, and Sensors. There is a risk that users may deny permissions for specific sensitive sensors (e.g., Microphone or App Usage Access), leading to missing modalities and crashing the fusion model.

Plan B: The system will be designed with "Graceful Degradation" (Modality Dropout) capabilities. The AI model will be trained to handle missing vectors by using zero-padding or dedicated "missing" tokens. If a user denies microphone access, the system will not crash but will switch to a "Behavioral-Only" mode, calculating risk scores based solely on digital phenotyping (steps, sleep, screen time) and explicitly informing the user that the analysis accuracy may be reduced.

Table 2: Factors that can affect analysis and design.

Risk	Likelihood	Plan B (Alternative)
Mobile Model Inefficiency	High	Local Network Offloading: Process data on the user's PC via LAN, keeping data private within the home network.
Training Hardware Constraints	Medium	Use LoRA / Gradient Accumulation for efficiency and utilize Secure On-Premise (Lab) hardware.
LLM Safety Failure	Low	Implement a deterministic "Rule-Based Safety Layer" that bypasses the LLM during crisis detection.
Permission Denial	High	Implement "Graceful Degradation" (Modality Dropout) to allow the app to function with partial data.

4.3. Project Plan

Below you can see various tables that you will make use of.

The project plan can be reported by list of work packages and their content.

For better readability, a Gant chart based on work packages can also be added.

Table 3: List of work packages

WP#	Work package title	Leader	Members involved
WP1	Project Specification Document	Merve Güleç	All Members
WP2	Analysis and Requirement Report	Murathan Işık	All Members

WP3	Detailed Design Report	Fatih Başal	All Members
WP4	Final Report	Metin Çalışkan	All Members
WP5	Presentation and Beta Demo	Yiğit Koşum	All Members
WP6	UI Desing	Merve Güleç	Murathan Işık, Yiğit Koşum
WP7	Flutter Development	Murathan Işık	Yiğit Koşum, Merve Güleç
WP8	Kotlin Development	Yiğit Koşum	Murathan Işık, Merve Güleç
WP9	NLP Models	Metin Çalışkan	Fatih Başal
WP10	Activity Models	Fatih Başal	Metin Çalışkan, Murathan Işık
WP11	Project Management	Yiğit Koşum	All Members
WP12	Final Presentation and Demo	Fatih Başal	All Members

<table>
<tr><td colspan="4">WP 1: Project Specification Document</td></tr>
<tr><td colspan="4">Start date: Nov 24, 2025 End date: Nov 28, 2025</td></tr>
<tr><td>Leader:</td><td>Merve Güleç</td><td>Members involved:</td><td>All Members</td></tr>
<tr><td colspan="4">Objectives: The objective is to define the problem scope, identify the core features of the ScubaMind solution, and establish the initial constraints and success criteria.</td></tr>
<tr><td colspan="4">Tasks:
Task 1.1 Introduction
Task 1.2 High Level System Architecture
Task 1.3 Constraints & Professional Issue
Tasl 1.4 Standards
Task 1.5 Design Requirements
Task 1.6 Feasibility Discussions
Task 1.7 Glossary & References</td></tr>
<tr><td colspan="4">Deliverables: Project Specification Report</td></tr>
</table>

Table 4: Work Package 1

<table>
<tr><td colspan="4">WP 2: Analysis and Requirement Report</td></tr>
<tr><td colspan="4">Start date: Dec 12, 2025 End date: Dec 19, 2025</td></tr>
<tr><td>Leader:</td><td>Murathan Işık</td><td>Members involved:</td><td>All Members</td></tr>
<tr><td colspan="4">Objectives: To perform a comprehensive analysis of the project's requirements, model the system architecture using UML diagrams, and address engineering constraints.</td></tr>
<tr><td colspan="4">Tasks:
Task 2.1 Introduction: Define the purpose of the analysis report and the scope of the ScubaMind system.
Task 2.2 Current System: Analyze the existing solutions in the market to identify gaps and opportunities for improvement.</td></tr>
</table>

Task 2.3 Proposed System: Define the detailed functional, non-functional, and pseudo requirements, and create comprehensive system models, including Use Case Models, Object/Class Models, and User Interface mockups. **Task 2.4 Other Analysis Elements:** Document the engineering constraints, standards, risks (with Plan B), project plan, teamwork strategies, ethical responsibilities, and learning strategies. **Task 2.5 Glossary** **Task 2.6 References**
Deliverables ***D2.1:*** *Analysis and Requirements Report*

Table 5: Work Package 2

WP 3: Detailed Design Report			
Start date: Second Semester **End date:** *Second Semester*			
Leader:	*Fatih Başal*	**Members involved:**	*All Members*
Objectives: *To translate the requirements into a concrete technical architecture. This involves defining the system's subsystems, data management strategies, security protocols, and designing comprehensive test cases to ensure quality assurance.*			
Tasks: ***Task 3.1 Introduction & Design Goals:*** *Define the primary design goals (e.g., Privacy, Low Latency, Energy Efficiency) and their prioritization.* ***Task 3.2 Software Architecture:*** *Detail the subsystem decomposition (AI Core, Android Services, UI Layer) and the hardware/software mapping (On-Device Inference architecture).* ***Task 3.3 Data & Security Design:*** *Specify the Room Database schema for persistent data management and design the access control policies (Authentication, Encryption).* ***Task 3.4 Subsystem Services:*** *Define the APIs and interfaces for internal communication (e.g., Method Channels between Flutter and Kotlin).* ***Task 3.5 Test Case Design:*** *Write 50+ detailed integration test cases (functional and non-functional) including Test ID, steps, expected outcomes, and severity, without executing them yet.* ***Task 3.6 Engineering Factors:*** *Re-evaluate and document how public health, safety, security, and other factors influenced the detailed design decisions (scored 0-10).*			

Task 3.7 Teamwork Documentation: *Document specific examples of how each member contributed, fostered collaboration, and shared leadership roles.*
Deliverables **D3.1:** Detailed Design Report

Table 6: Work Package 3

WP 4: Final Report			
Start date: Second Semester **End date:** *Second Semester*			
Leader :	*Metin Çalışkan*	**Members involved:**	*All Members*
Objectives: *To produce a comprehensive, self-contained document representing the culmination of the project. This package aims to document the final status of the system, including detailed implementation specifics, execution results of test cases, maintenance plans, and a reflective analysis of teamwork and ethics.*			
Tasks: ***Task 4.1 Introduction & Requirements Details*** ***Task 4.2 Final Architecture and Design Details:*** *Document the final system architecture, differentiating between the initial design and the implemented solution.* ***Task 4.3 Development/Implementation Details:*** *Provide detailed technical explanations of the code structure, specific algorithms, and libraries used.* ***Task 4.4 Test Cases and Results:*** *Execute the test plan defined in WP3 and document the actual results (pass/fail), including integration and user acceptance tests.* ***Task 4.5 Maintenance Plan and Details:*** *Create a guide for future maintenance, including how to update AI models or handle API changes.* ***Task 4.6 Other Project Elements:*** *Document "Consideration of Various Factors," "Ethics and Professional Responsibilities," "Teamwork Details" (contributions, shared leadership, meeting objectives), and "New Knowledge Acquired."* ***Task 4.7 User's Manual Preparation:*** *Create a guide with installation instructions and usage scenarios for the final software.* ***Task 4.8 Conclusion and Future Work:*** *Summarize the project achievements and outline potential future improvements.*			

Deliverables **D4.1:** Final Project Report **D4.2:** User's Manual & Installation Guide **D4.3:** Final Software Source Code

Table 7: Work Package 4

<table>
<tr><td colspan="4">WP 5: Presentation and Beta Demo</td></tr>
<tr><td colspan="4">Start date: Dec 12, 2025 End date: Dec 23, 2025</td></tr>
<tr><td>Leader:</td><td>Yiğit Koşum</td><td>Members involved:</td><td>All Members</td></tr>
<tr><td colspan="4">Objectives: To prepare and deliver a professional presentation and a functional Beta demonstration to the course instructors and supervisors.</td></tr>
<tr><td colspan="4">Tasks:
Task 5.1 Presentation Content Preparation: Design slides covering the project overview, problem definition, architectural diagrams, and current project status.
Task 5.2 Beta Prototype Integration: Implement a functional demo version that showcases installed technologies and core resolutions, even if features are not fully complete.
Task 5.3 Rehearsal & Coordination: Conduct team practice sessions to ensure smooth transitions between speakers, verify timing, and prepare for potential Q&A scenarios regarding technical challenges.
Task 5.4 Demo Environment Setup</td></tr>
<tr><td colspan="4">Deliverables
D5.1: Project Presentation Slides
D5.2: Functional Beta Prototype
D5.3: Live Demo Session</td></tr>
</table>

Table 8: Work Package 5

WP 6: UI Design			
Start date: Oct 24, 2025 **End date:** *Second Semester*			
Leader :	*Merve Güleç*	**Members involved:**	Murathan Işık, Yiğit Koşum
Objectives: *To design a calming, intuitive, and accessible user interface tailored for individuals potentially experiencing depression or anxiety.*			
Tasks: ***Task 6.1 Design System Creation:*** *Define the typography, iconography, and color palette (e.g., soothing blues/greens) to ensure consistency across the application.* ***Task 6.2 Low-Fidelity Wireframing:*** *Sketch the structural layout of core screens to validate user flow and navigation logic.* ***Task 6.3 High-Fidelity Prototyping:*** *Create detailed, pixel-perfect screen designs in Figma, including the "Insight Dashboard" with gauge indicators and trend charts .* ***Task 6.4 Usability Review: E****valuate the designs against accessibility standards to ensure ease of use for the target demographic.*			
Deliverables **D6.1:** UI Design System & Style Guide **D6.2:** High-Fidelity Screen Mock-ups **D6.3:** Interactive UI Prototype			

Table 9: Work Package 6

WP 7: Flutter Development			
Start date: Nov 15, 2025 **End date:** *Second Semester*			
Leader :	*Murathan Işık*	**Members involved:**	Merve Güleç, Yiğit Koşum
Objectives: *To implement the cross-platform user interface using the Flutter framework.*			
Tasks: ***Task 7.1 Project Setup & State Management: I****nitialize the Flutter project structure and implement a robust state management solution (e.g., BLoC , Cubit) to handle real-time data updates.*			

Task 7.2 Dashboard Implementation: *Develop the "Insight Dashboard" to visualize longitudinal trend charts for mood and behavioral metrics .* ***Task 7.3 Chat Interface Development:*** *Build the secure "AI Chat Interface" for the LLM Companion, ensuring smooth text streaming and a user-friendly conversational experience.* ***Task 7.4 Method Channel Integration:*** *Implement the asynchronous bridge (Method Channels) to trigger Native Android services and retrieve sensor data from the Kotlin layer.* ***Task 7.5 Permission & Settings UI:*** *Create the "Permission Control" screens to allow users to toggle specific sensors and manage data retention policies.*
Deliverables **D7.1:** Functional Mobile Application UI **D7.2:** Flutter Source Code Repository

Table 10 Work Package 7

WP 8: Kotlin Development			
Start date: Nov 25, 2025 **End date:** *Second Semester*			
Leader :	*Yiğit Koşum*	**Members involved:**	Murathan Işık, Merve Güleç
Objectives: *To develop the operational backbone of the system using Native Android (Kotlin).*			
Tasks: ***Task 8.1 Background Service Architecture:*** *Implement Android `WorkManager` and `JobScheduler` to orchestrate periodic data collection tasks efficiently.* ***Task 8.2 Sensor & Data Managers:*** *Develop specific modules to interface with hardware sensors and system APIs to retrieve behavioral data .* ***Task 8.3 Audio Recorder Implementation:*** *Create the secure audio capture module that records raw audio.* ***Task 8.4 Local Database Implementation:*** *Set up the Room Database with encryption to securely persist sensor logs, journals, and calculated risk scores locally .* ***Task 8.5 AI Inference Bridge:*** *Implement the logic to feed collected data into the embedded AI core and retrieve inference results for the UI .*			

Deliverables
D8.1: Native Android Data Layer
D8.2: Kotlin Source Code Repository

Table 11: Work Package 8

WP 9: NLP Models

Start date: Nov 15, 2025 **End date:** *Second Semester*

Leader:	*Metin Çalışkan*	**Members involved:**	Fatih Başal

Objectives: *To design, train, and optimize the core AI model responsible for detecting depression risk from text and speech transcripts.*

Tasks:
Task 9.1 Data Preparation & Preprocessing: *Clean, format and transcript the E-DAIC dataset audios, implementing "User-Level" splits to prevent data leakage and ensure robust validation .*
Task 9.2 Teacher Model Training: *Fine-tune large-scale Transformer models on specific auxiliary tasks: Emotion Recognition, Sentiment Analysis, and Personality Detection to act as "Teachers".*
Task 9.3 Student Model Distillation: *Implement the distillation training loop where the lightweight Student model learns to mimic the soft labels of the Teacher models alongside the ground truth depression scores.*
Task 9.2 Speech Feature Extraction (Prosody): *Develop the pipeline to extract acoustic features (pitch, energy, speaking rate) using libraries like OpenSmile or pre-trained audio models.*
Task 9.4 Hyperparameter Tuning: *Utilize optimization frameworks to find the optimal balance for learning rates, distillation temperature, and loss weights to maximize the Concordance Correlation Coefficient (CCC).*
Task 9.5 Model Quantization & Export: *Convert the trained PyTorch model to the ONNX format and apply Dynamic Quantization to reduce model size and inference latency for mobile integration.*

Deliverables
D9.1: Preprocessed Training & Validation Datasets
D9.2: Trained & Validated Student Model (.onnx)
D9.3: Model Performance Report (CCC Metrics & Error Analysis)

Table 12: Work Package 9

<table>
<tr><td colspan="4">WP 10: Activity Models</td></tr>
<tr><td colspan="4">Start date: Dec 15, 2025 End date: Second Semester</td></tr>
<tr><td>Leader :</td><td>Fatih Başal</td><td>Members involved:</td><td>Metin Çalışkan, Murathan Işık</td></tr>
<tr><td colspan="4">Objectives: To develop the algorithmic models responsible for analyzing non-verbal, passive behavioral data.</td></tr>
<tr><td colspan="4">Tasks:
Task 10.1 Mobility Feature Extraction: Implement algorithms to calculate key mobility metrics from location data to detect reduced movement associated with depression.
Task 10.2 Digital Phenotyping Logic: Develop the logic to analyze aggregated application usage patterns to infer personality traits and social withdrawal.
Task 10.3 Behavior Continuity Modeling: Implement the "behavior continuity" approach to establish a user-specific baseline, allowing the system to detect deviations from the user's own normal patterns rather than generic population averages

Task 10.4 Multimodal Fusion Strategy: Design the final fully connected neural network that synthesizes the diverse feature vectors into a single, continuous Risk Score.</td></tr>
<tr><td colspan="4">Deliverables
D10.1: Behavioral Feature Extraction Algorithms (Python/Kotlin)
D10.2: Fusion Model Architecture & Weights
D10.3: Behavioral Analysis Validation Report</td></tr>
</table>

Table 13: Work Package 10

<table>
<tr><td colspan="4">WP 11: Project Management</td></tr>
<tr><td colspan="4">Start date: Oct 1, 2025 End date: Second Semester</td></tr>
<tr><td>Leader :</td><td>Yiğit Koşum</td><td>Members involved:</td><td>All Members</td></tr>
<tr><td colspan="4">Objectives: To ensure the project stays on schedule, team communication remains effective, and risks are actively managed.</td></tr>
</table>

Tasks: ***Task 11.1 Meeting Coordination:*** *Schedule and conduct weekly internal team synchronization meetings and bi-weekly supervisor progress meetings.* ***Task 11.2 Repository & Version Control:*** *Manage the GitHub organization, enforce branching strategies (feature/main), and conduct code reviews to maintain code quality.* ***Task 11.3 Risk Monitoring:*** *Continuously monitor the "Risks" identified in the Analysis Report (e.g., model accuracy, battery drain) and trigger "Plan B" contingencies if necessary.* ***Task 11.4 Documentation Management:*** *Organize shared resources to ensure all documentation is collaborative, up-to-date, and accessible to all members.*
Deliverables **D11.1:** Weekly Meeting Minutes **D11.2:** Bi-Weekly Progress Reports (for Supervisor) **D11.3:** Updated Project Schedule/Gantt Chart

Table 14: Work Package 11

<table>
<tr><td colspan="4">WP 12: Final Presentation and Demo</td></tr>
<tr><td colspan="4">Start date: Second Semester End date: Second Semester</td></tr>
<tr><td>Leader:</td><td>Fatih Başal</td><td>Members involved:</td><td>All Members</td></tr>
<tr><td colspan="4">Objectives: To deliver a professional, polished presentation and live demonstration of the completed ScubaMind project.</td></tr>
<tr><td colspan="4">Tasks:

Task 12.1 Presentation Design & Structure: Prepare a focused slide deck that covers the project

Task 12.2 Demo Flow Planning: Design a specific demo scenario that strictly avoids focuses on unique value propositions.</td></tr>
</table>

Task 12.3 Environment Setup ***Task 12.4 Reflection & Technical Narrative*** ***Task 12.5 Rehearsal***
Deliverables **D12.1:** Final Presentation Slides **D12.2:** Live System Demonstration **D12.3:** Project Demo Video

Table 15: Work Package 12

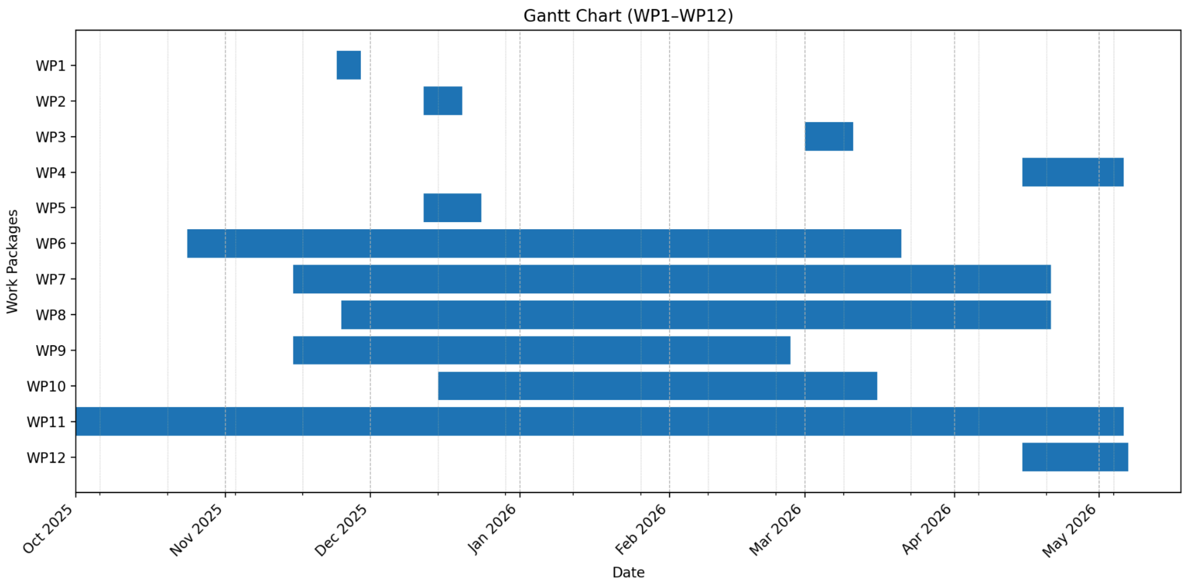


Fig. 24: Work Package Gantt Chart

4.4. Ensuring Proper Teamwork

The ScubaMind project is a comprehensive Senior Design Project that brings together software development, mobile systems, artificial intelligence, and mental health. Therefore, the goal is for teamwork to be carried out in a planned and organized manner to ensure the project progresses smoothly.

4.4.1 Defining Roles and Responsibilities

At the beginning of the project process, the responsibilities of the team members were clarified, and the work plan was created using a Work Package (WP) structure. A leader was assigned to each work package, but the active contribution of all team members to the process was considered essential. Thanks to this approach:

- Tasks were clarified,
- The workload was distributed evenly,
- The project was prevented from being dependent on a single person.

4.4.2 Task Tracking and Project Management

Trello was used for planning, distributing, and tracking tasks. Task cards were created for each work package on Trello and these cards were divided into the following stages:

- To Do
- In Progress
- Review
- Done

This structure allowed team members to easily track both their own responsibilities and the overall progress of the project. Furthermore, delays could be detected early because deadlines and task owners were clearly defined.

4.4.3 Communication and Coordination

Weekly meetings were held to ensure team coordination. Meetings were also held with the mentor instructor every two weeks. These meetings included:

- Evaluating technical progress,
- Discussing encountered problems,
- Coordinating inter-module dependencies.

Git-based version control was used in the code development process; changes were made through branches and merged using pull requests. This process contributed to peer review within the team and the maintenance of code quality.

4.5. Ethics and Professional Responsibilities

Due to the fact that the ScubaMind project works with highly sensitive data regarding users' mental health, ethical and professional responsibilities are central to the project process. The team approached this project not merely as a technical software development process, but also as a system design that could directly impact human lives and psychological well-being. Therefore, all design decisions were made with ethical principles, professional responsibilities, and user safety in mind.

4.5.1 Clinical Expert Opinion and Ethical Awareness

Throughout the project, it was acknowledged that depression should be understood not only through technical signals but also within its clinical and psychological context. To this end, a face-to-face meeting was held with Liora Psychology Clinic. During this meeting, the team had the opportunity to hear from experts about:

- How depression is clinically assessed,
- The limits within which digital applications can be used in the field of mental health,
- The difference between a software system performing "risk estimation" and making a "clinical diagnosis".

Based on the feedback received from clinical experts, the outputs presented to the user were designed to increase awareness and encourage seeking professional support, rather than making definitive judgments.

4.5.2 User Safety and Risk of Misinterpretation

One of the biggest ethical risks in mental health applications is users misinterpreting system outputs. In particular, phrases like "Depression Risk Score" can be perceived by users as a definitive diagnosis. This can lead to unnecessary anxiety or incorrect decisions. To mitigate this risk:

- All outputs are clearly presented as "risk," "estimate," and "behavioral insight,"
- The interface emphasizes that the system is not a substitute for a doctor,
- Users are advised to seek professional support if needed.

This approach aims to both fulfill ethical responsibility and prevent negative psychological impacts on the user.

4.5.3 Privacy, Data Protection, and User Control

One of the most important components of ScubaMind's ethical responsibilities is user privacy. The system has been designed from the outset according to the principles of Privacy by Design and Local-First Architecture. In this context:

- All analyses are performed using on-device processing,
- Raw data (audio, keystrokes, sensor logs) is not sent to the cloud,
- Users can revoke permissions at any time (Right to be Forgotten).

This approach aims to prevent users from feeling monitored and to foster trust in the system. Especially in the mental health field, trust is considered critical for system adoption.

4.6. Planning for New Knowledge and Learning Strategies

The ScubaMind project is a study that requires team members not only to utilize their existing knowledge and skills but also to continuously learn new information throughout the project. Therefore, the project was also approached as a planned learning-oriented engineering process.

4.6.1 Domain Knowledge Learning

The field of mental health is not one where progress can be made solely through technical knowledge. Therefore, the team identified understanding the clinical and psychological dimensions of depression as a significant learning objective. The interview with Liora Psychology Clinic was a valuable learning resource in this regard. Through this interview, the team gained a better understanding of:

- How depression symptoms can vary from person to person,
- How digital data can be used as a tool to support clinical assessment,
- The potential impact of misinterpretations on the user.

This information contributed to a more careful and responsible design of the system.

4.6.2 Technical Knowledge and Applied Learning

During the project, many new technologies were used, including Flutter, Native Android (Kotlin), NLP models, on-device inference, and sensor-based data collection. Team members were familiar with some of these technologies, but they were learned by everyone thanks to the enthusiasm and contributions of all team members during the application development process. The team adopted a learning-by-doing approach and immediately applied the learned information to the application. Thanks to this approach: Theoretical knowledge was reinforced with practice and technical problems encountered were turned into learning opportunities.

4.6.3 Team Learning and Knowledge Sharing

Since team members focused on different areas, knowledge sharing was seen as an important learning strategy. In weekly meetings: Each team member shared developments in their area of responsibility and newly learned concepts were transferred to other team members. In this way, the knowledge base within the team increased in a balanced way, and a common understanding was formed throughout the project.

4.6.4 Feedback-Oriented Learning

In the ScubaMind project, the learning process was not limited to internal team work but was also supported by feedback from experienced individuals in the field.

In this context, meetings were held with an innovation expert and a mentor from the Global Turks AI community during the project. In these meetings, the project's overall vision, technical architecture, AI components, and positioning in the mental health field were shared in detail. The feedback given by the mentors focused particularly on the system's innovation potential, scalability, and real-world applicability. Based on the feedback received:

- It was emphasized that the project should be considered not only as an academic study but also as a usable decision-support system in real life;
- The on-device processing and local-first architecture approach was confirmed as a strong choice in terms of user trust and ethics;
- The multimodal fusion approach was noted as one of the most distinctive and innovative aspects of the system;
- The importance of using simple and understandable language in the outputs presented to the user was particularly emphasized.

This feedbacks has contributed to a reassessment of the system design and the definition of some components with clearer boundaries. As a result, feedback from academic advisors, clinical experts, and mentors with industry experience has helped the ScubaMind team make more informed decisions, both technically and ethically.

5. Glossary

Accelerometer: A phone sensor that measures linear acceleration; used to infer movement intensity and activity patterns.

Acoustic Features: Quantitative properties extracted from voice (e.g., pitch, energy, speaking rate) used for emotion-related analysis.

AI Companion (LLM Companion): Optional conversational support component powered by a Large Language Model, providing empathetic dialogue and guidance.

AI Inference (On-Device Inference): Running a trained ML model to produce predictions directly on the user's device (not in the cloud).

AI Module (Embedded Core): The on-device analysis core that processes multimodal inputs and generates a depression risk output.

Alternative Flow: A non-main path in a scenario/use case describing what happens when a step fails or differs (e.g., permission refusal).

Android Background Service: A system component that can run tasks in the background (e.g., sensor polling, wearable sync) even when the UI is not active.

API (Application Programming Interface): A defined interface that allows software components to communicate (e.g., Health Connect API).

App Usage Category: A privacy-preserving grouping of apps (e.g., Communication, Games) used to compute behavioral signals without exposing exact app names.

App Launch Frequency: How often apps are opened; can indicate routine changes or withdrawal patterns.

Batch Processing: Aggregating data over time and processing it periodically to reduce energy and compute cost.

Behavioral Baseline (Personal Baseline): A user-specific “normal” reference built from historical data to interpret changes as deviations rather than absolute thresholds.

Behavioral Markers: Measurable signals reflecting mental state changes (e.g., reduced mobility, disrupted sleep, increased screen time).

Behavioral-Only Mode: A degraded operating mode where the system computes outputs using only available non-audio/text modalities (e.g., steps, sleep, usage).

Bias (Model Bias): Systematic performance differences across user groups (dialect, culture, behavior) that can lead to unfair or inaccurate risk estimates.

BLoC (Business Logic Component): A Flutter state-management pattern for separating UI from logic (used conceptually for modularity and maintainability).

CBT (Cognitive Behavioral Therapy): A structured therapy approach; in apps it often appears as guided exercises and prompts (not a replacement for therapy).

Class Model: A UML model representing system classes, attributes, and relationships (static structure).

Cloud Offloading: Sending computation to external servers; ScubaMind minimizes this to protect privacy (except optional LLM API usage).

Cognitive Load: Mental effort required to use the app; the UI is designed to minimize it for users experiencing low motivation or stress.

Consent (Informed Consent): Explicit permission from the user after being clearly informed what data is collected, why, and how it’s used.

Context Feeding: Passing summarized state signals into the LLM prompt (with user permission) to generate more personalized, safe responses.

Context Window: The amount of prior information an LLM can consider when generating responses; impacts cost and safety constraints.

Crisis Escalation: A safety behavior where the system stops normal interactions and directs users to professional/emergency resources when high risk is detected.

Data Minimization: Privacy principle of collecting only the minimum data required to provide the intended functionality.

Data Sovereignty: User control over their data (permissions, retention, deletion, and sharing).

Data Sparsity: Missing data caused by denied permissions, device limitations, or irregular use; requires robust handling.

Database Encryption (Data at Rest): Encrypting stored data (e.g., Room DB) to protect confidentiality on the device.

Decision-Support System: A tool that supports awareness and decisions but does not provide clinical diagnosis or treatment.

Depression Risk Score (Risk Score): A continuous or categorical estimate of depression risk derived from multimodal signals; presented as "insight," not diagnosis.

Digital Ecosystem (Digital Mental Health Ecosystem): The landscape of existing mental health solutions (mood journals, meditation apps, chatbots).

Digital Interaction Patterns: Behavioral signals extracted from phone usage (screen time, unlocks, app launches).

Digital Phenotyping: Passive measurement of behavior and physiology via personal devices to infer mental health-related patterns.

Dynamic Model: A UML-style model describing system behavior over time (e.g., activity diagrams, sequence diagrams).

DBT (Dialectical Behavior Therapy): A therapy approach; in apps, often used as guided coping exercises (not a replacement for clinical care).

Embedded Model: A model deployed to run locally on a device (mobile CPU/NPU), typically optimized for speed and energy.

Energy Efficiency: Minimizing battery usage while collecting and analyzing data continuously.

Entry Condition: A prerequisite that must be true before a scenario/use case begins.

Exit Condition: A condition that indicates the scenario/use case has completed successfully (or reached a terminal state).

Ethical Constraints: Limitations imposed to protect users (privacy, non-diagnostic framing, crisis protocols, fairness).

FHIR (HL7 FHIR): A healthcare data standard used to structure and represent health-related records for interoperability (adopted as a conceptual basis for storing outputs).

Flutter Module (UI Layer): Cross-platform UI layer that visualizes insights, manages user settings, and initiates actions.

Functional Requirement: A requirement describing what the system must do (features and behaviors).

Fusion Model (Multimodal Fusion Model): A model that combines multiple modality feature vectors into one prediction (risk score).

Gantt Chart: A schedule visualization for work packages and tasks over time.

Gap Analysis: Identifying shortcomings in current solutions compared to the proposed system goals.

GDPR: EU regulation governing personal data processing; relevant due to sensitive health-related data.

Graceful Degradation: Continuing to provide meaningful outputs when some data sources are unavailable (partial permissions, missing sensors).

Gradient Accumulation: Training technique that simulates larger batch sizes by accumulating gradients across steps to reduce memory use.

Health Connect API: Android interface for reading health metrics (sleep, HR, activity) from wearables and health apps.

Heart Rate (HR): Beats per minute; used as a physiological indicator.

HRV (Heart Rate Variability): Variations between heartbeats; often associated with stress and recovery.

Hardware Sensors: Physical sensors on devices (accelerometer, gyroscope, step counter, GPS) used for passive data.

IEEE 11073: A standard family for communication and data exchange with personal health devices (used conceptually for wearable data standardization).

IMS (Input Method Service): Android framework for implementing custom keyboards.

Insight Dashboard: UI section that displays risk scores and trends through charts/indicators.

INT8 Quantization: Converting model weights/activations to 8-bit integers to speed up inference and reduce memory use (may reduce accuracy).

Interoperability: Ability to exchange and use information across systems; influenced by standards like FHIR.

Journal Entry (Text Journaling): Optional user-provided text used for semantic analysis and supportive reflection.

Just-in-Time Notification: A notification delivered at a context-appropriate time to reduce disruption and increase engagement.

Keyboard Dynamics (Typing Dynamics): Privacy-preserving features from typing behavior (speed, rhythm, intensity) without storing raw text.

KVKK: Turkey's data protection law; relevant due to sensitive personal data processing.

Latency: Delay between user action and system response (e.g., UI target ≤ 300 ms).

LLM (Large Language Model): A generative model used for conversational support and guidance.

Local Area Network (LAN) Offloading: Plan B approach where heavy computation is sent to a user's local PC over the home network (not the internet).

Local-First Architecture: Data collection, storage, and analysis are primarily performed on-device to maximize privacy and reliability.

LoRA (Low-Rank Adaptation): Parameter-efficient fine-tuning technique that trains small adapter layers instead of the full model to reduce compute and memory needs.

Longitudinal Analysis: Trend analysis across days/weeks/months to detect meaningful behavioral change patterns.

Maintainability: Ease of modifying and extending the system over time (architecture, modularity, testability).

Market Outlook: Forward-looking assessment of market size and growth that motivates feasibility.

Microphone Access: Permission-controlled audio capture used for speech feature extraction (optional and sensitive).

Mobility Metrics: Measures derived from location and movement (e.g., step count, routine disruption).

Modality: A data source type (text, audio, mobility, physiology, usage).

Modality Dropout: Training/operation strategy where the model is designed to handle missing modalities (supports graceful degradation).

Multimodal: Using multiple data types together (sensor + wearable + usage + optional text/audio).

Native Android Layer (Data & Storage): Kotlin-based layer managing background collection, storage, encryption, and data transfer to AI inference.

Network Independence (Offline Mode): Core features remain functional without internet access.

NLP (Natural Language Processing): Methods/models that analyze text to extract sentiment and semantic meaning.

NPU (Neural Processing Unit): Dedicated mobile hardware accelerator for ML inference (device-dependent).

Notification Service: Local mechanism that triggers reminders and context-aware prompts.

Object Model: A model describing objects and their relationships/attributes (often aligned with UML).

Offline Training Constraint: Restriction where sensitive datasets cannot be uploaded to cloud services, limiting training to local/on-premise compute.

ONNX Runtime: A runtime for executing optimized ML models; commonly used for on-device inference.

Onboarding: Initial guided setup that explains the system, collects consent, and configures preferences.

Passive Data Collection: Collecting behavior/physiology signals in the background with minimal user effort.

Permission Control: UI and logic for enabling/disabling specific sensors/data streams and managing privacy settings.

PHQ-9: A 9-item depression questionnaire; used as a conceptual reference for presenting risk-aligned outputs (not a diagnosis).

Platform Dependency: Constraint caused by using Android-specific services and permissions (initially Android-focused).

Privacy by Design: Designing the system from the start to minimize data exposure and maximize user control.

Prosody Model: Model/feature pipeline analyzing voice characteristics (intonation, pitch variability, speaking rate).

Pseudo Requirement: A provisional assumption/guiding constraint that is not yet a finalized requirement but informs design decisions.

Pruning: Removing model parameters to reduce size and speed up inference (can affect accuracy).

Prompt Engineering: Designing prompts and context rules to keep LLM responses safe, relevant, and cost-efficient.

Quantization: Compressing model representations (e.g., FP32 → INT8) for faster on-device inference and reduced memory.

Questionnaire Bias (Recall Bias / Social Desirability Bias): Self-report limitations where people forget, misremember, or underreport due to social pressure.

Radius of Gyration: A mobility metric summarizing how widely a person moves across locations over time; can correlate with depressive symptoms.

Recall Bias: Systematic errors caused by imperfect memory in self-reports.

Reliability: Ability of the system to behave consistently and avoid data loss or failures in background operation.

Report Generation (Doctor-Sharable Report): Creating a summary (e.g., PDF) of trends and insights for optional sharing with clinicians.

Risk Analysis (Risk Assessment): Estimating likelihood/severity of concern based on signals; explicitly not a clinical diagnosis.

Room Database: Android persistence library over SQLite; used for structured local storage (with encryption).

Routine Index: A mobility-derived metric measuring regularity of daily movement patterns and deviations from routine.

Safety Guardrails: Rules and constraints that prevent harmful LLM outputs and ensure appropriate responses for mental health contexts.

Scenario: Narrative description of how a user/system achieves a goal (often tied to use cases).

Screen Time: Duration of phone use; can reflect behavioral shifts relevant to mood changes.

Sensor Manager: Background service that polls sensors (steps, GPS variance, accelerometer) and computes mobility-related features.

Sentiment Analysis: Extracting emotional polarity/affect from text or transcribed speech.

Sequence Diagram: UML diagram showing interactions between components over time (a type of dynamic model).

Soft-Tone Theme: UI design approach using calming colors and low visual intensity for mental health contexts.

Social Desirability Bias: Tendency to underreport symptoms to appear “better” during self-reporting.

Speech Processing Module: Pipeline that converts speech to text (ASR) and extracts acoustic/prosodic features.

Standards (Engineering Standards): Established guidelines (e.g., ISO/IEC/IEEE 29148, UML 2.5.1, ISO/IEC 27001, HL7 FHIR, IEEE 11073) used to structure requirements and system design.

Supportability: Ability to support multiple Android versions/devices and handle dependency changes over time.

System Model: A formal representation of system structure/behavior (use cases, class/object models, dynamic models, UI mock-ups).

Token Cost: Pricing model for LLM APIs based on input/output token counts; constrains LLM companion usage.

Transformer: A deep learning architecture commonly used for NLP tasks, producing embeddings and contextual understanding.

Transcription (ASR): Automatic Speech Recognition converting audio to text, enabling linguistic analysis.

Trend Visualization: Charts/graphs showing changes over time (daily/weekly/monthly) for behavior and risk outputs.

UML 2.5.1: Unified Modeling Language standard used to represent system diagrams (use case, class, sequence/activity diagrams).

Unlock Count: How many times the phone is unlocked; a digital behavior indicator.

UsageStats Manager: Android service/module that reads aggregated usage statistics (screen time, app launches) under permission control.

Usability: Ease and calmness of interaction, including low tap count, intuitive onboarding, and clear visuals.

User Profile: Local user-specific configuration and preferences (not necessarily cloud-backed).

User Retention Policy: Rules for how long local data is kept before deletion or summarization (user-controlled).

Volatile Memory Processing: Handling sensitive raw inputs (keystrokes/audio) briefly in memory to extract features without permanently storing raw content.

Wearable Integration: Synchronizing wearable data (HR, HRV, sleep, activity) via system health APIs to enrich context.

Whisper.cpp: A lightweight C/C++ implementation of Whisper for on-device ASR (mentioned as a possible toolchain example).

Work Package (WP): A structured unit of project planning that groups tasks, responsibilities, dates, and deliverables.

Zero-Padding: Filling missing modality vectors with zeros to allow models to run when some inputs are unavailable (graceful degradation).

6. References

[1] World Health Organization, *Depression: Key Facts*, Geneva, Switzerland, 2023. Available: https://www.who.int/news-room/fact-sheets/detail/depression

[2] J. M. Gotlib and I. Joormann, "Cognition and depression: Current status and future directions," *Annual Review of Clinical Psychology*, vol. 6, pp. 285–312, 2010. doi: https://doi.org/10.1146/annurev.clinpsy.121208.131305

[3] Daylio Team, *Daylio – Mood Tracker & Journal*, 2024. Available: https://daylio.net/

[4] Headspace Inc., *Headspace – Meditation and Mindfulness App*, 2024. Available: https://www.headspace.com/

[5] Wysa Ltd., *Wysa – AI Mental Health Support*, 2024. Available: https://www.wysa.com/

[6] Youper Inc., *Youper – Emotional Health Assistant*, 2024. Available: https://www.youper.ai/

[7] Calm.com, Inc., *Calm – Meditation and Sleep App*, 2024. Available: https://www.calm.com/

[8] L. Canzian and M. Musolesi, “Trajectories of depression: Unobtrusive monitoring of depressive states by means of smartphone mobility traces analysis,” in *Proc. 2015 ACM Int. Joint Conf. Pervasive and Ubiquitous Comput. (UbiComp ’15)*, Osaka, Japan, Sept. 2015, pp. 1293–1304.