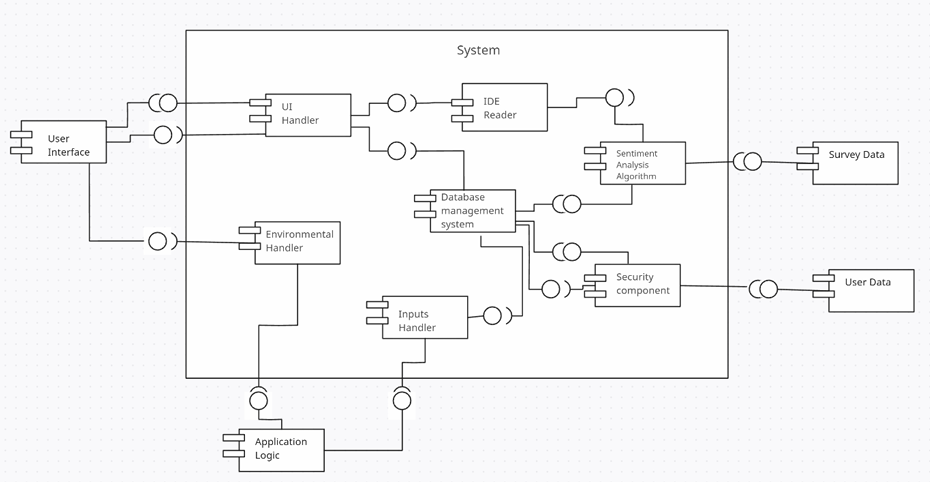
**III.1. Overview**

The current project builds upon the architecture and development efforts of previous iterations, focusing on integrating and improving two primary applications: 1) a Qualtrics-based web survey report that generates personalized personality assessments, and 2) a mobile self-monitoring app that logs emotional events daily. Our architectural design follows the client-server model, which ensures scalability, modularity, and reliable interaction between the client-side user interfaces and the server-side logic that handles data processing, storage, and analysis.

The client-server model was chosen for several reasons. First, this pattern allows for clear separation of concerns: the client focuses on user interaction and inputs, while the server handles data processing, sentiment analysis, and storage. Given the large volume of data and the need for event-based functionality (logging, monitoring, and real-time feedback), this model ensures efficient data handling and user experience. It also facilitates the integration of advanced features, such as treatment recommendations and clustering algorithms in the mobile app, making it flexible for future enhancements.

The system consists of several key components that work together to provide real-time, personalized feedback to users. On the client side, users interact with the Qualtrics-based survey and self-monitoring mobile app to input their emotional events or complete personality assessments. These inputs are processed by the server, which stores the data securely, analyzes it using sentiment analysis and other algorithms, and generates relevant feedback. The feedback is then delivered back to the client as personalized reports or recommendations.

The UML components illustrate the interaction between client-side (UI) and server-side components (application logic, DBMS, and security), as well as the flow of data in the system. The diagram highlights how different modules interact to achieve the goals of the system, from user input to data processing and feedback generation.

****

**III.2. Subsystem Decomposition**

### **2.1. [UI Handler]**

#### **2.1.1. Description**

The UI Handler is responsible for managing all interactions between the user and the system. It handles input from both the web-based Qualtrics survey and the mobile self-monitoring app. This subsystem ensures that users can submit their survey responses and emotional event logs, and it relays this information to the appropriate back-end components for further processing. It also manages the output by displaying personalized feedback or reports generated by the system.

#### **2.1.2. Concepts and Algorithms Generated**

The UI Handler does not utilize complex algorithms but instead focuses on input validation and user interaction. It uses basic validation mechanisms to ensure that the input data (survey responses, logs) is in the correct format and passes the necessary checks before being sent to the system. The selection of this solution was driven by the need for an efficient and responsive user interface that can process user inputs in real-time and provide an intuitive experience.

#### **2.1.3. Interface Description**

Services Provided:

| Service Name | Service Provided To | Description |
| --- | --- | --- |
| Input Collection | Input Handler | Collects and validates user input (survey data or emotional events) and forwards it to Input Handler. |
| Display Feedback | Application Logic | Displays feedback or reports generated by the system back to the user. |

Services Required:

| Service Name | Service Provided From |
| --- | --- |
| Processed Feedback | Application Logic |
| Input Validation | Input Handler |

### **2.2. [Environmental Handler]**

#### **2.2.1. Description**

The Environmental Handler manages external data interactions, ensuring that the system can adapt to the environment it operates in. It gathers data from sources such as mobile sensors, location data, and other external APIs, which is used to enhance the user experience in the self-monitoring app. For example, it might adjust recommendations based on the user's environment.

#### **2.2.2. Concepts and Algorithms Generated**

The Environmental Handler utilizes location-based algorithms to detect user context and adjust app behavior. It uses a combination of external APIs and real-time data gathering techniques. The decision to use these algorithms was based on the need to provide a dynamic and context-aware experience for users, especially in the self-monitoring app.

#### **2.2.3. Interface Description**

Services Provided:

| Service Name | Service Provided To | Description |
| --- | --- | --- |
| Location Data | Application Logic | Provides location-based information to adjust app behavior or user feedback. |
| External Data Updates | Input Handler | Supplies data from external APIs, like weather or location data, to augment user inputs. |

Services Required:

| Service Name | Service Provided From |
| --- | --- |
| Input Processing | Input Handler |
| External API Data | External Sources |

### **2.3. [Input Handler]**

#### **2.3.1. Description**

The Input Handler processes all user inputs submitted through the UI. It ensures the correct format and validity of the data before sending it to the Application Logic for further processing, such as report generation or emotional analysis.

#### **2.3.2. Concepts and Algorithms Generated**

The Input Handler employs data validation algorithms to ensure that all inputs conform to the expected format and values. This was chosen to maintain the integrity of data entering the system and prevent errors during further processing.

#### **2.3.3. Interface Description**

Services Provided:

| Service Name | Service Provided To | Description |
| --- | --- | --- |
| Validated Input | Application Logic | Sends validated user inputs (survey or emotional logs) for further processing. |
| Data Preprocessing | Sentiment Analysis Algorithm | Prepares user input data for sentiment analysis. |

Services Required:

| Service Name | Service Provided From |
| --- | --- |
| User Input Collection | UI Handler |
| Environmental Data | Environmental Handler |

### **2.4. [IDE Reader]**

#### **2.4.1. Description**

The IDE Reader collects data from the Integrated Development Environment (IDE) or external file sources, and supplies it to the relevant system components, such as for analysis or report generation.

#### **2.4.2. Concepts and Algorithms Generated**

No specialized algorithms are used here beyond standard file reading and parsing techniques. The primary objective is to accurately retrieve and transfer data.

#### **2.4.3. Interface Description**

Services Provided:

| Service Name | Service Provided To | Description |
| --- | --- | --- |
| Data Input | Application Logic | Reads data from files or external sources and forwards it for processing. |

Services Required:

| Service Name | Service Provided From |
| --- | --- |
| Data Storage Access | Database Management |

### **2.5. [Database Management System]**

#### **2.5.1. Description**

The Database Management System (DBMS) stores all user data, including survey responses and logs from the self-monitoring app. It ensures that data is efficiently stored, retrieved, and updated as needed, while providing security measures to protect user privacy.

#### **2.5.2. Concepts and Algorithms Generated**

The DBMS uses standard database indexing and encryption techniques to handle large volumes of data securely and efficiently. The trade-off considered was between performance and security, resulting in the selection of algorithms that prioritize data integrity and protection.

#### **2.5.3. Interface Description**

Services Provided:

| Service Name | Service Provided To | Description |
| --- | --- | --- |
| Data Storage | Application Logic | Provides secure storage for user inputs and logs, as well as survey results. |
| Data Retrieval | Sentiment Analysis Algorithm | Retrieves relevant user data for analysis. |

Services Required:

| Service Name | Service Provided From |
| --- | --- |
| Processed Data Storage | Application Logic |
| Data Access | IDE Reader |

### **2.6. [Sentiment Analysis Algorithm]**

#### **2.6.1. Description**

This subsystem performs sentiment analysis on the data collected from users. It evaluates survey responses and daily logs, extracting emotional patterns and providing insights that are used in generating feedback and recommendations.

#### **2.6.2. Concepts and Algorithms Generated**

The sentiment analysis algorithm relies on natural language processing (NLP) techniques to extract emotional cues from user inputs. This allows for personalized feedback based on users' emotional states. The algorithm chosen was based on its accuracy and ability to handle large-scale input data efficiently.

#### **2.6.3. Interface Description**

Services Provided:

| Service Name | Service Provided To | Description |
| --- | --- | --- |
| Sentiment Analysis | Application Logic | Provides emotional analysis based on user inputs, assisting in generating personalized reports. |

Services Required:

| Service Name | Service Provided From |
| --- | --- |
| Input Data | Input Handler |

### **2.7. [Security Component]**

#### **2.7.1. Description**

The Security Component handles the security of user data throughout the system, providing encryption, authentication, and secure communication between subsystems to ensure that all user interactions and data storage remain protected.

#### **2.7.2. Concepts and Algorithms Generated**

The Security Component uses encryption algorithms (such as AES) and authentication protocols (OAuth) to ensure secure access and data transmission. Trade-offs between performance and security were considered, with a focus on safeguarding sensitive user data without significantly impacting system performance.

#### **2.7.3. Interface Description**

Services Provided:

| Service Name | Service Provided To | Description |
| --- | --- | --- |
| Data Encryption | Database Management | Ensures all stored data is encrypted and secure from unauthorized access. |
| User Authentication | UI Handler | Manages login credentials and ensures that only authorized users can access the system. |

Services Required:

| Service Name | Service Provided From |
| --- | --- |
| User Data Encryption | Database Management |