**Class Diagram**

Based on the functional and non-functional requirements provided, the following classes and their public and private methods can be defined for the AI system:

1. **SensorDataProcessor** Class - This class is responsible for processing raw data from different sensors.

Public Methods:

* process\_sensor\_data(data: dict) -> dict: This method takes the raw sensor data as input and returns processed data in a dictionary format.

Private Methods:

* \_remove\_noise(data: dict) -> dict: This method removes the noise from the raw sensor data.
* \_clean\_data(data: dict) -> dict: This method cleans the raw sensor data and removes any outliers.

2. **NeuralNetworkTrainer** Class – This class is responsible for training the neural network with the preprocessed sensor data.

Public Methods:

* train\_network(data: dict) -> None: This method takes the preprocessed sensor data as input and trains the neural network with it.

Private Methods:

* \_create\_network() -> None: This method creates the neural network structure.
* \_update\_network(data: dict) -> None: This method updates the neural network parameters based on the training results.

3. **UserIntentPredictor** Class - This class is responsible for predicting the user's movement intentions.

Public Methods:

* predict\_movement\_intent(data: dict) -> str: This method takes the preprocessed sensor data as input and predicts the user's movement intention.

Private Methods:

* \_adjust\_network\_structure(data: dict) -> None: This method adjusts the neural network structure to better predict the personal intent of a real-world user.

4. **MovementFeedbackGenerator** Class - This class is responsible for providing feedback and visualization to the user.

Public Methods:

* generate\_feedback(data: dict) -> dict: This method takes the preprocessed sensor data as input and generates feedback in the form of graphs and charts of their movement patterns and progress.

Private Methods:

* \_generate\_graphs(data: dict) -> dict: This method generates graphs and charts of the user's movement patterns and progress.

5. **DataHandler** Class - This class is responsible for handling the data and its privacy and security.

Public Methods:

* handle\_data(data: dict) -> None: This method takes the preprocessed sensor data as input and handles the privacy and security of the user's personal data and movements and complies with relevant data protection and privacy laws and regulations.

Private Methods:

* \_store\_data(data: dict) -> None: This method stores the preprocessed sensor data in a secure and confidential manner.

The connections between classes are as follows:

* The SensorDataProcessor class is connected to the NeuralNetworkTrainer class and the UserIntentPredictor class.
* The NeuralNetworkTrainer class is connected to the UserIntentPredictor class.
* The UserIntentPredictor class is connected to the MovementFeedbackGenerator class.
* The DataHandler class is connected to all other classes to handle the data and ensure its privacy and security.