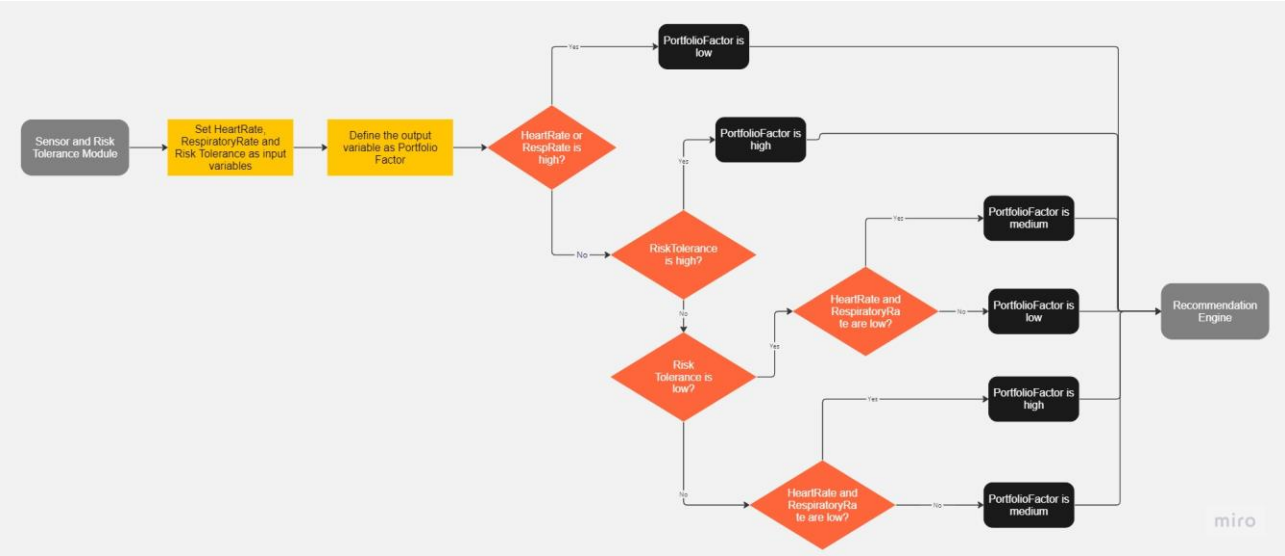
CSE 535 PROJECT 4 INDIVIDUAL REPORT

Alignment with Guardian Angel:

Our investment recommendation app's fuzzy logic system acts as the user's "Guardian Angel" by continuously evaluating their physical health, stress levels, and risk tolerance. This is consistent with the idea of the Guardian Angel because it actively observes and adjusts to the user's emotional and financial condition, offering customized suggestions to guarantee a safer and more customized investing experience. The user's real-time sensor data is used to extract the stress level and physical conditions. These are combined with the user's input of risk tolerance to determine the dynamic portfolio factor, which can be conservative, balanced, or aggressive.

Specifications:



Real-time data acquisition of biometric inputs (heart rate, breathing rate), as well as user risk tolerance, are part of the fuzzy system's control flow. After processing this data, the fuzzy logic engine determines a dynamic portfolio factor that can be low, medium, or high, indicating aggressive, balanced, or conservative trading strategies. This factor then influences the investment recommendations.

The sensor data module provides the heart rate and respiratory inputs, which are then transmitted to the fuzzy system. The user's risk tolerance is derived from the user input provided in the RiskToleranceActivity.

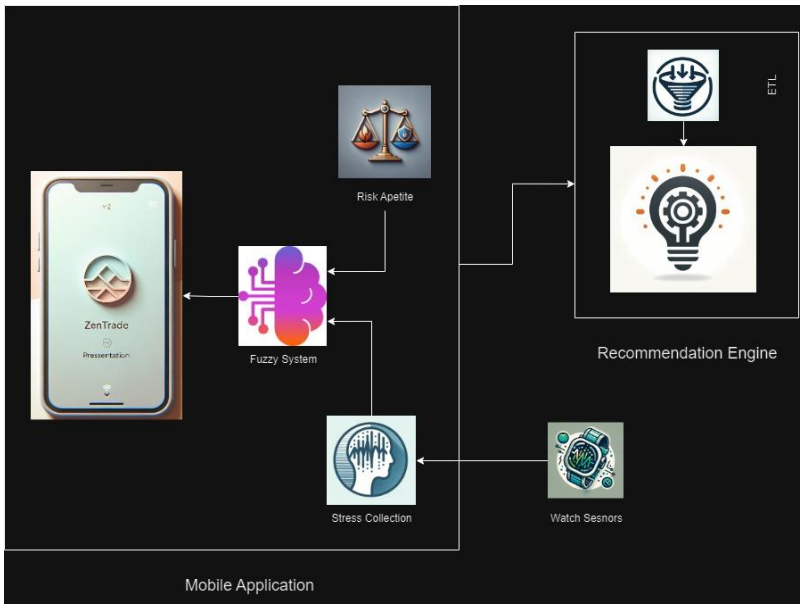
Design:

There are three primary modules in the architecture of the fuzzy system component:

The mobile application is in charge of gathering and analyzing data on heart rate, breathing rate, and the user's risk tolerance.

Fuzzy Logic System: Based on inputs, a dynamic portfolio factor is computed using a fuzzy system.

Recommendation Engine: Makes investment recommendations based on fuzzy's provided portfolio factor.



Easy updates and modifications are made possible by the design's emphasis on modularity. The entire investment recommendation system can be seamlessly integrated with the fuzzy logic engine because it is made to adjust to the changing states of the user.
Tech-stack Used: The inputs, output, and rules were defined in the FCL file by using the fuzzy control language. To call the fuzzy model and preprocess the input values, the Kotlin FuzzyEvaluate class was created.

Testing Strategies:

Tested individual functions within the fuzzy logic engine to ensure accurate calculations. Dynamic testing of different scenarios have been done taking into consideration the cases where the heartrate and respiratory rate can be classified as high, medium and low and giving different risk tolerance values. The testing has also been done by running the function several time by generating random values in a specified range to see the adaptability of the Fuzzy system. The test cases where an invalid file is passed or fcl file is not properly saved is also taken.
Simulated different stress and risk scenarios to verify the adaptability of the fuzzy system.

Example Test cases:

```
fun testFuzzyLowRisk() {
   val result = fuzzyEvaluate.fuzzy("low", 80, 18)
   Assert.assertEquals("Low", result) }

fun testFuzzyMediumRisk() {
   val result = fuzzyEvaluate.fuzzy("low", 130, 20)
   Assert.assertEquals("High", result) }
```

```
fun testFuzzyHighRisk() {
   val result = fuzzyEvaluate.fuzzy("medium", 50, 12)
   Assert.assertEquals("High", result) }
```

Navigating Challenges and Lessons learned:

Error-handling mechanisms need to be implemented to address inaccuracies in biometric data. Typecasting and lowercase conventions have been used to address these issues.

Developing an appropriate and effective rule set is challenging. Survey of multiple prospective users on how they would want their recommendations to be based on their preferences and conditions and updated the fuzzy rules based on the feedback.

Determining optimal membership function graphs and curve limits can be time consuming. Using the testcases to determine the appropriate outputs have been used based on which the parameters have been tuned.