

**Министерство науки и высшего образования Российской Федерации**  
федеральное государственное автономное образовательное учреждение высшего образования  
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**Отчет**

по лабораторной работе №4 «Разработка структурной диаграммы информационной системы»

по дисциплине «Проектирование информационных систем»

Автор: Ковалев Г. П.

Факультет: ПИН

Группа: К3341

Преподаватель: Духанов А.В.



Санкт-Петербург 2025

Я делал работу, используя ChatGPT, я выдал ему 2, 3 лабораторную, как историю, и написал данный промпт:

UML1.pdf  
PDF

смотри, теперь как отсюда мне нужно построить диаграмму классов, Число компонент или классов не менее пяти. Для диаграммы компонент должно быть хотя бы одна вложенность. Отображение баз данных или иных элементов - в соответствии с требованиями к системе. приведи пример как можно сделать для моего проекта такую диаграмму

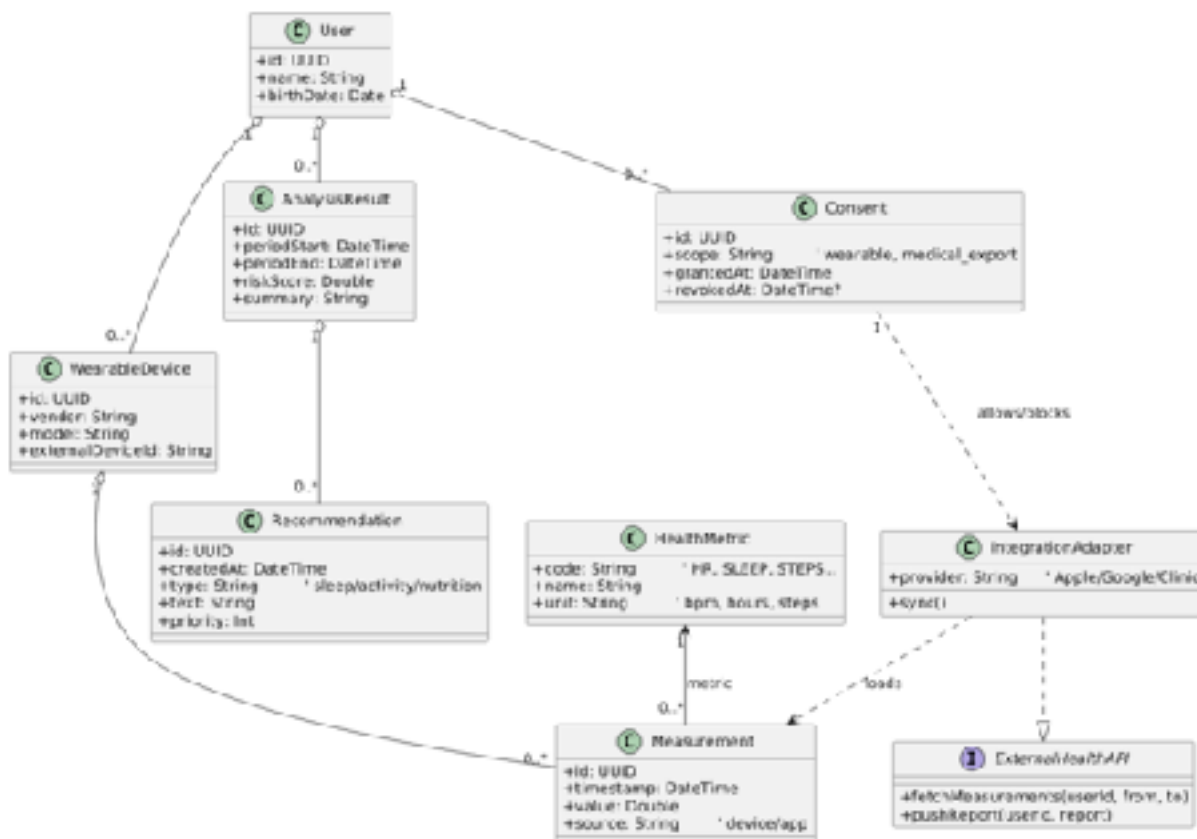
Думал на протяжении 8s

Ниже — пример, как из твоего ТЗ по HealthNavigator собрать (1) диаграмму классов (≥5 классов) и (2) диаграмму компонентов (≥5 компонентов + есть вложенность). Логика берётся из функций/модулей: сбор данных с носимых устройств → анализ → рекомендации → интеграции и согласия пользователя. [view\\_tz.pdf](#)

**1) Пример диаграммы классов (PlantUML)**

Идея: классы отражают данные (пользователь, измерения, метрики, согласия) и результаты обработки (анализ, рекомендации), плюс связь с устройствами/интеграциями.

Как результат была выдана данная диаграмма:



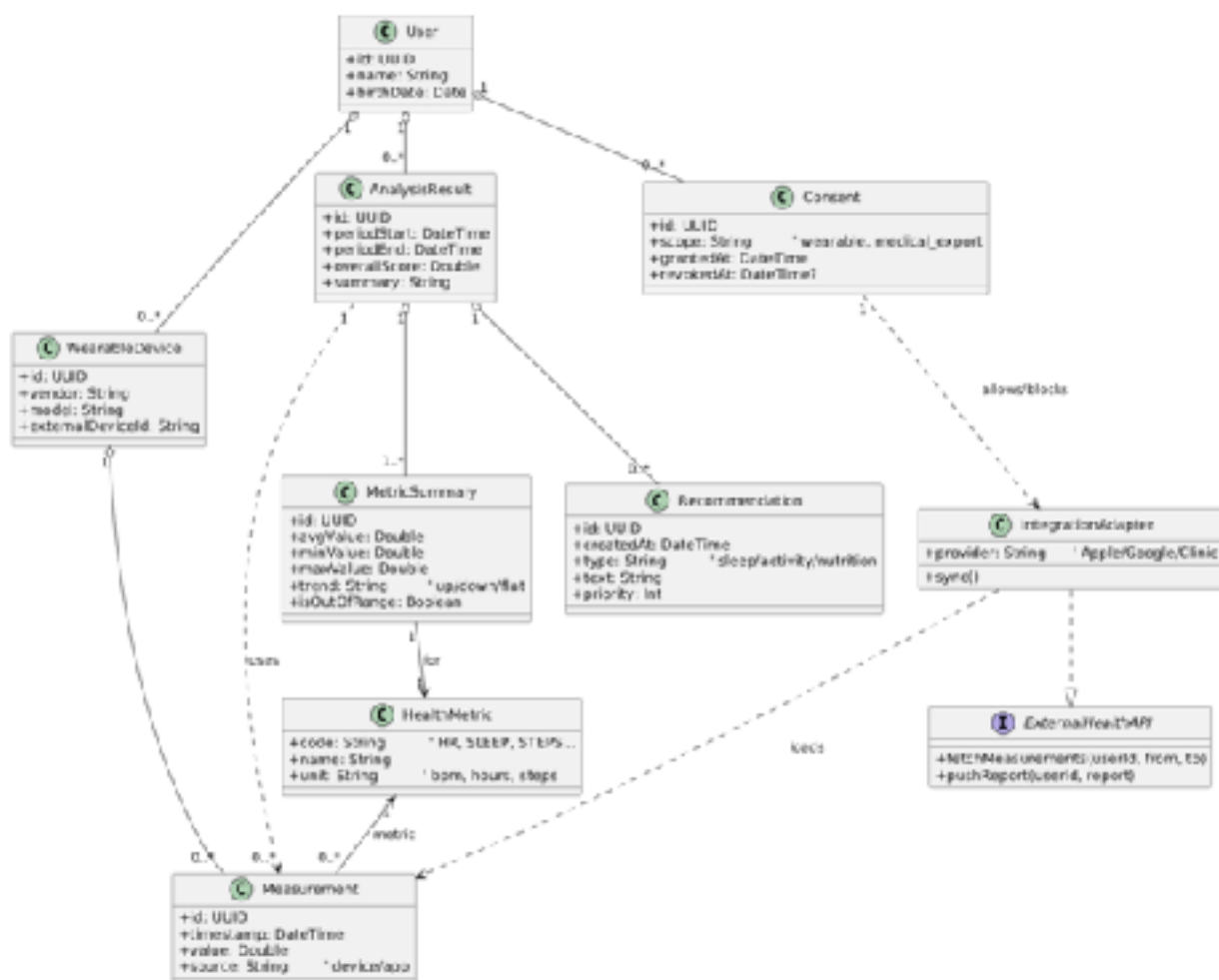
Данную диаграмму можно взять за основу, однако она совсем неверна. Нет никакой связи между HealthMetric и AnalysisResult, непонятно, откуда берутся данные для анализа, по данной диаграмме они берутся от User, однако они должны приходить от HealthMetric. Я решил исправить это написав данный промпт:

я делаю диаграмму классов, исправь код, который ты мне отправил, от HealthMetric нет никакой связи ни с User, ни с AnalysisResult, эти данные должны как-то уходить в анализ, давай это исправим

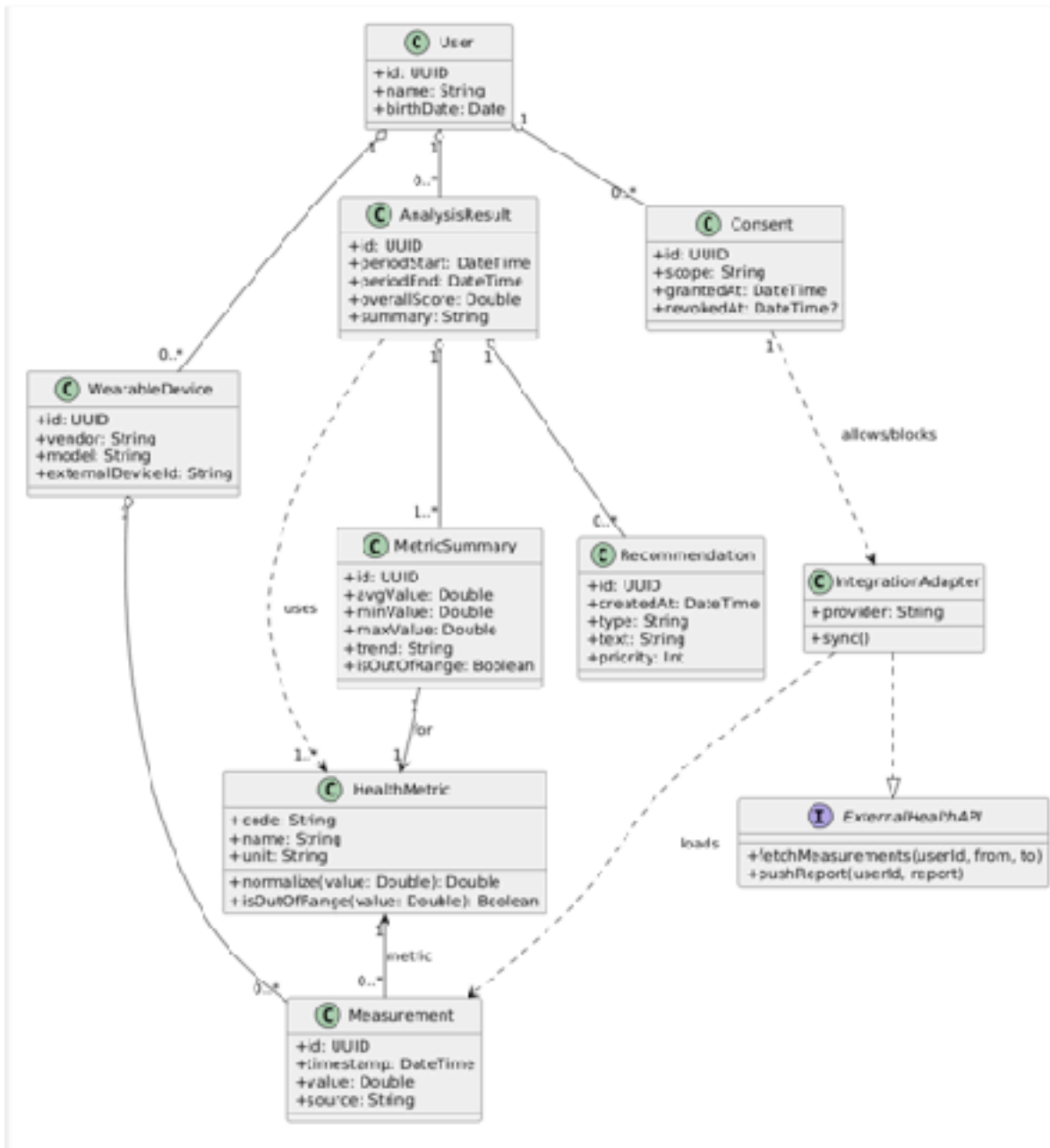
Понял — в текущем варианте HealthMetric действительно "висит" только через Measurement, а связь с анализом неочевидна. Логичнее показать, что:

- анализ использует измерения (в т.ч. по конкретным метрикам),
- результат анализа хранит агрегаты/оценки по метрикам (или хотя бы ссылки на метрики, которые анализировались).

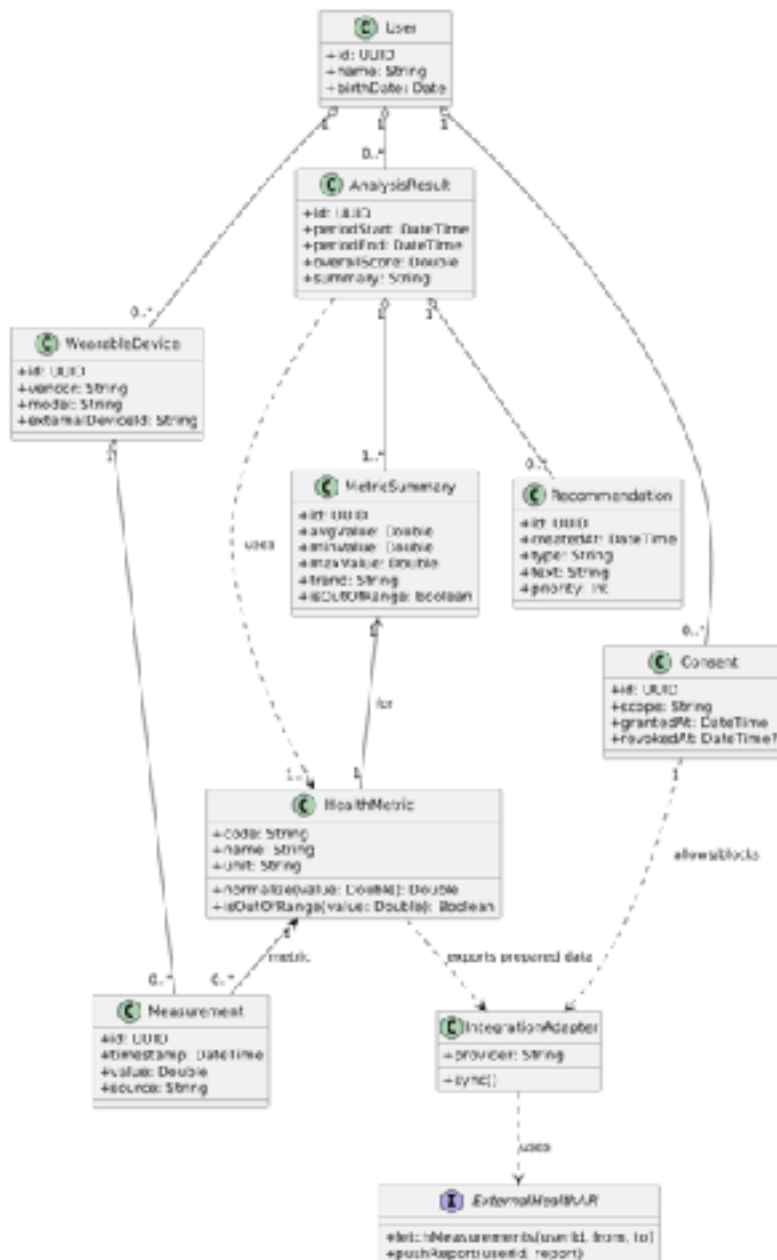
Самый аккуратный UML-способ — добавить класс MetricSummary (или MetricAggregate) и связать его с AnalysisResult, HealthMetric и (опционально) с Measurement. Тогда видно "поток" данных: Measurement → AnalysisResult → MetricSummary → Recommendation.



В данной диаграмме классов связь между классами AnalysisResult и Measurement была исключена. Это обусловлено тем, что первичная обработка «сырых» данных измерений выполняется классом HealthMetric, который инкапсулирует логику нормализации и проверки показателей. В объект AnalysisResult поступают уже агрегированные данные в виде объектов класса MetricSummary, сформированные на основе соответствующих метрик.



В результате анализа связей было выяснено, что неправильные связи были между классами Measurement и IntegrationAdapter, по моей логике данные должны уходить из HealthMetric. Так же я поменял стрелочку между HealthMetric и MetricSummary, все таки данные уходят именно Summary. По итогу получилась следующая диаграмма:



Распишу ее логику. В данной диаграмме классов пользователь (User) владеет носимыми устройствами (WearableDevice) и управляет своими согласиями (Consent), которые определяют допустимые сценарии обработки и передачи данных. Носимые устройства формируют измерения (Measurement), каждое из которых связано с определённой метрикой здоровья (HealthMetric), инкапсулирующей логику нормализации и проверки значений. Класс HealthMetric выполняет первичную обработку данных измерений и используется при формировании агрегированных показателей (MetricSummary). Объекты MetricSummary входят в состав результата анализа (AnalysisResult), который представляет обобщённую оценку состояния здоровья пользователя за заданный период. Класс AnalysisResult использует логику метрик (HealthMetric) для интерпретации агрегированных данных и на их основе формирует рекомендации (Recommendation). Подготовленные и нормализованные

данные метрик могут передаваться в компонент интеграции (IntegrationAdapter) для внешнего обмена. Перед выполнением интеграции адаптер проверяет наличие действующего согласия (Consent), после чего осуществляет взаимодействие с внешними медицинскими или платформенными API (ExternalHealthAPI).

Конечная диаграмма достаточно точно отображает ту логику, по которой, по моему мнению, должна работать программа. Мне очень понравилось работать с языком PlantUML.