Internet Banking Application

Analysis and Design Document

# Elaboration – Iteration 1.1

# Domain Model

The application is responsible for performing operations on bank accounts, so the following entities should be present:

* Client – holds personal information about clients, such as username, password, name, address, phone number, and a list of created accounts.
* Account – holds account-related information, such as the type of account (credit, debit, savings), the balance, creation/expiration date.
* Transaction – knows who made a transaction, from which account, what type of operation, the sum of money that was involved and the date in which the transaction was performed.

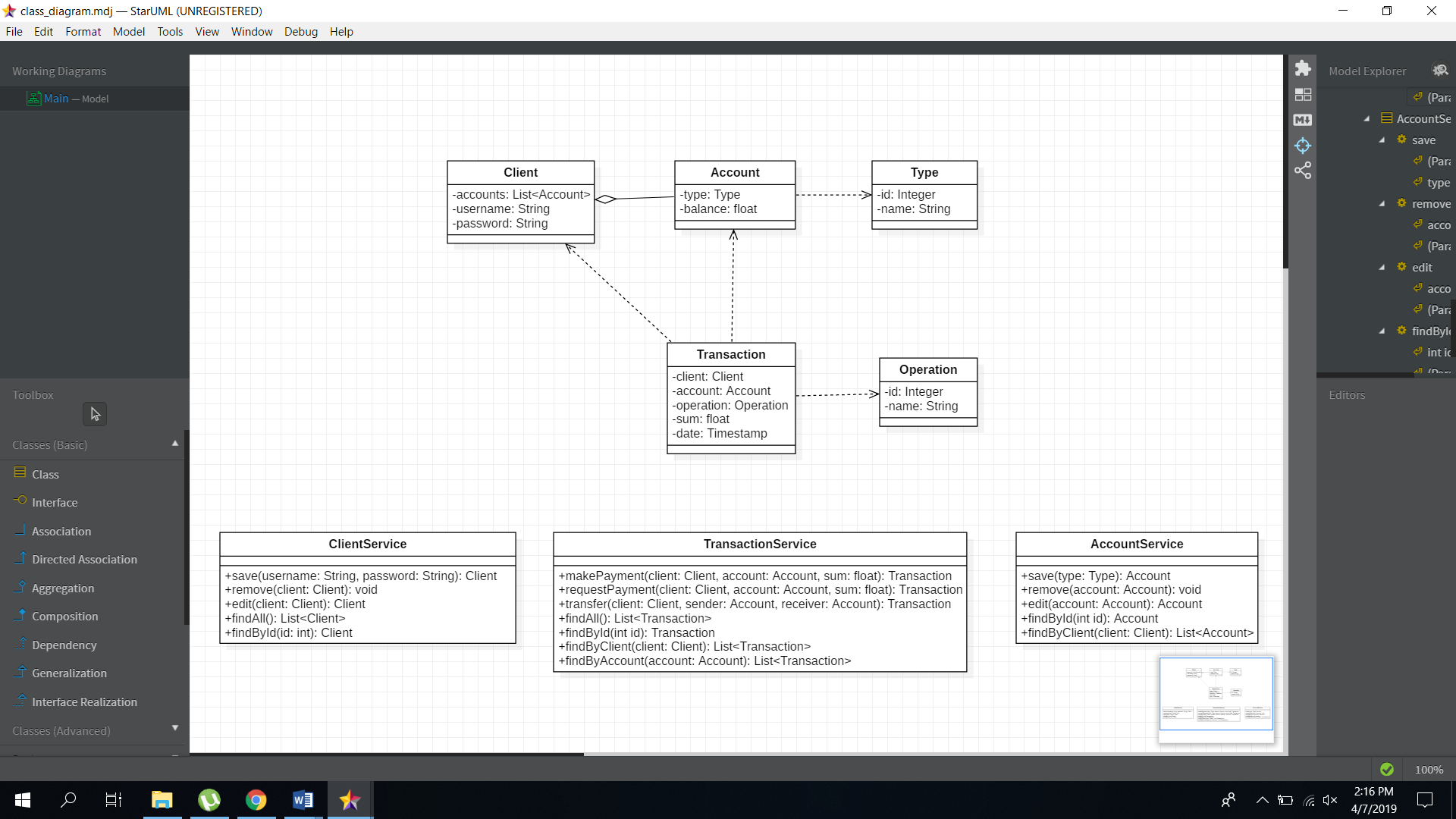


Figure 1‑1 Conceptual Class Diagram

# Architectural Design

## Conceptual Architecture

The system we are designing will be based on the Multi-layered architectural pattern, consisting of the following layers:

* Presentation Layer – responsible for the User Interface design
* Business Logic Layer – the layer that coordinates the application, processes commands, makes logical decisions and evaluations, and performs calcuations. It also moves and processes data between the two surrounding layers.
* Data Access Layer – here information is stored and retrieved from a database, and passed to the business layer where it is processed.

The reason for choosing this pattern is because it is fairly easy to understand and to implement, and I already have some experience implementing it. Another reason is that the system is modular, and can be developed concurrently (if a team was involved), and because each layer is separated and not mixed together, we can have a better understanding of what it does and how it works.

## Package Design

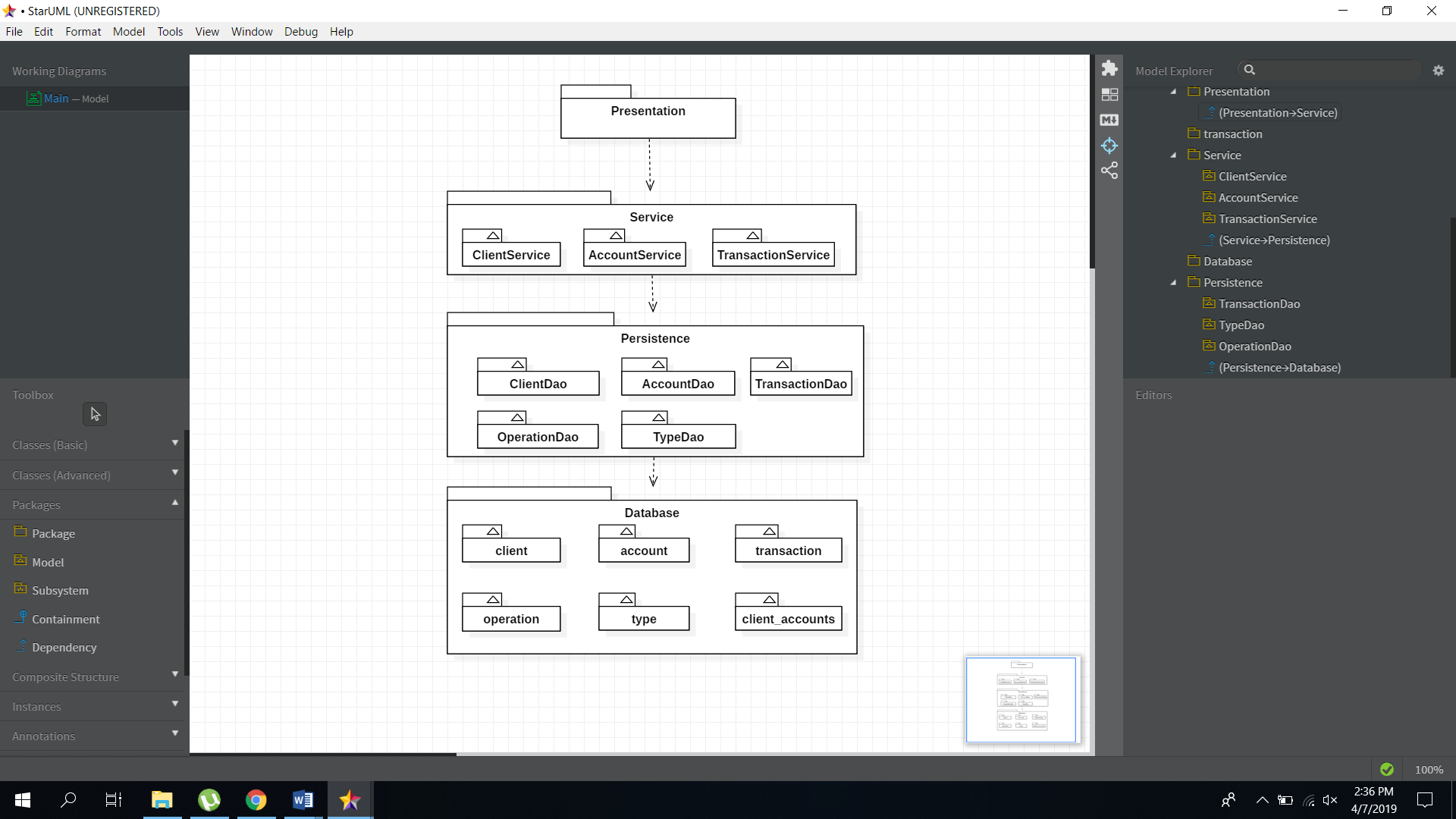


Figure 2‑1 Package design

## Component and Deployment Diagrams

# 

Figure 2.3‑1 Component Diagram

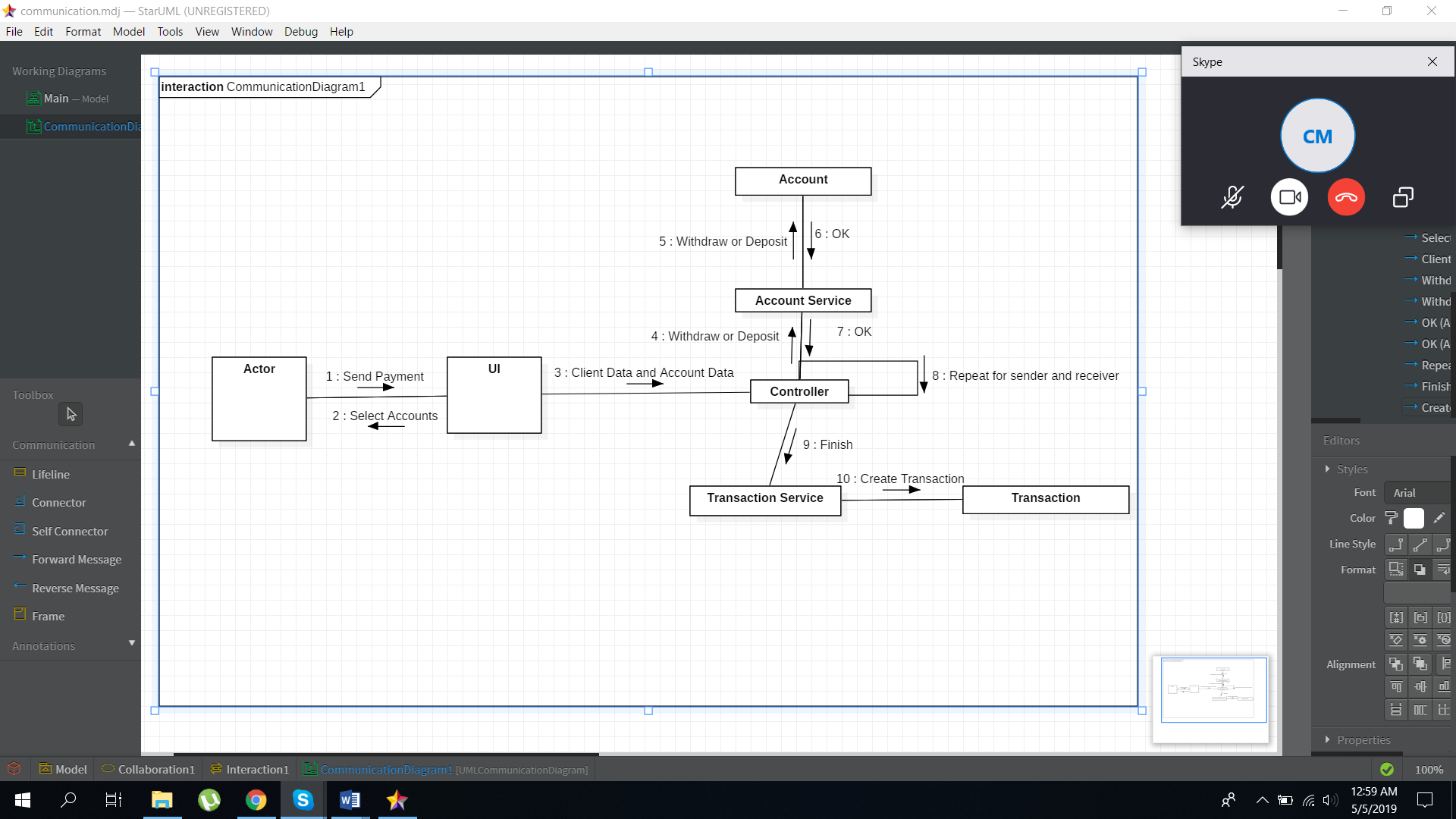
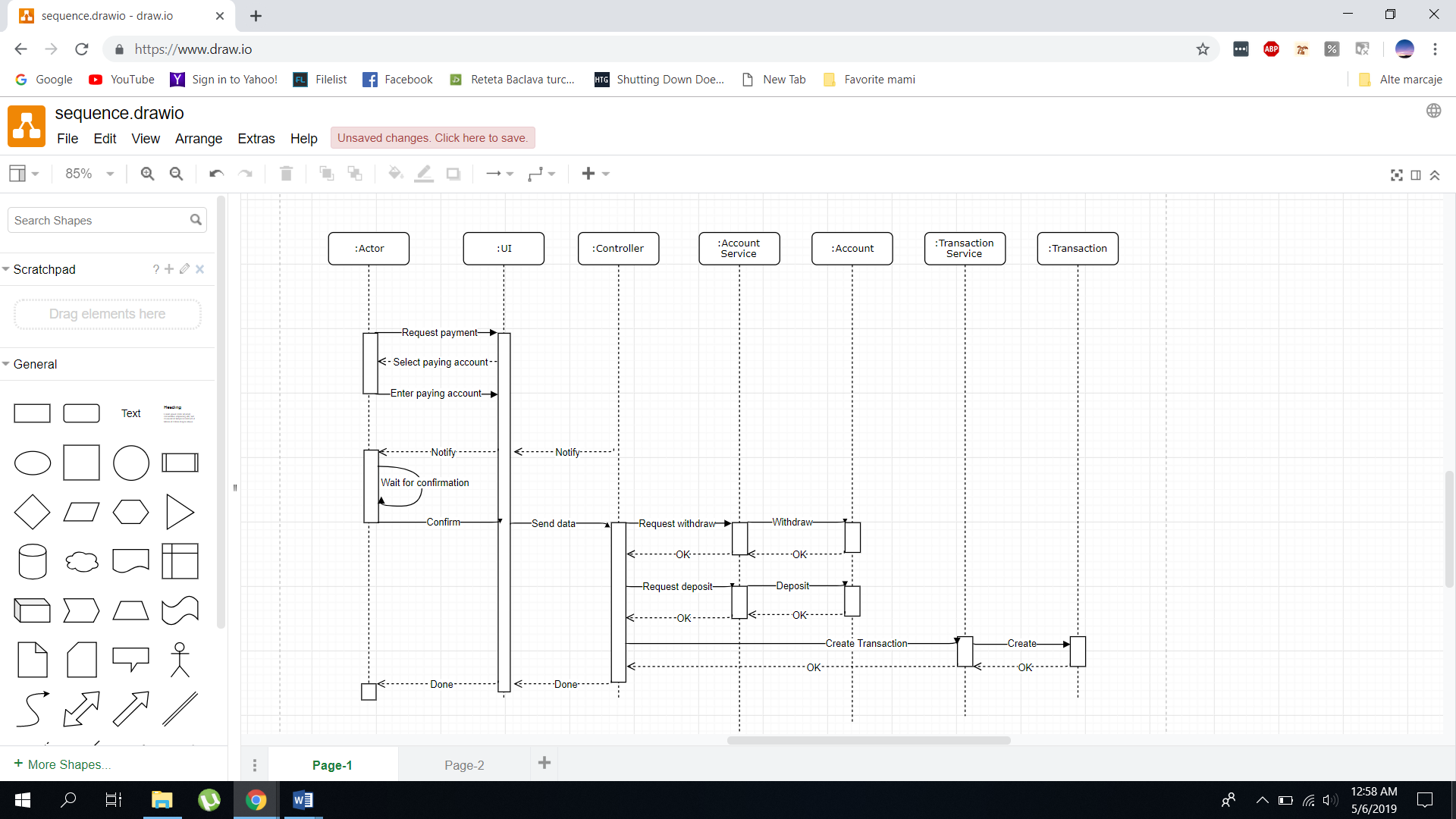
# 

Figure 2.3‑2 Deployment Diagram

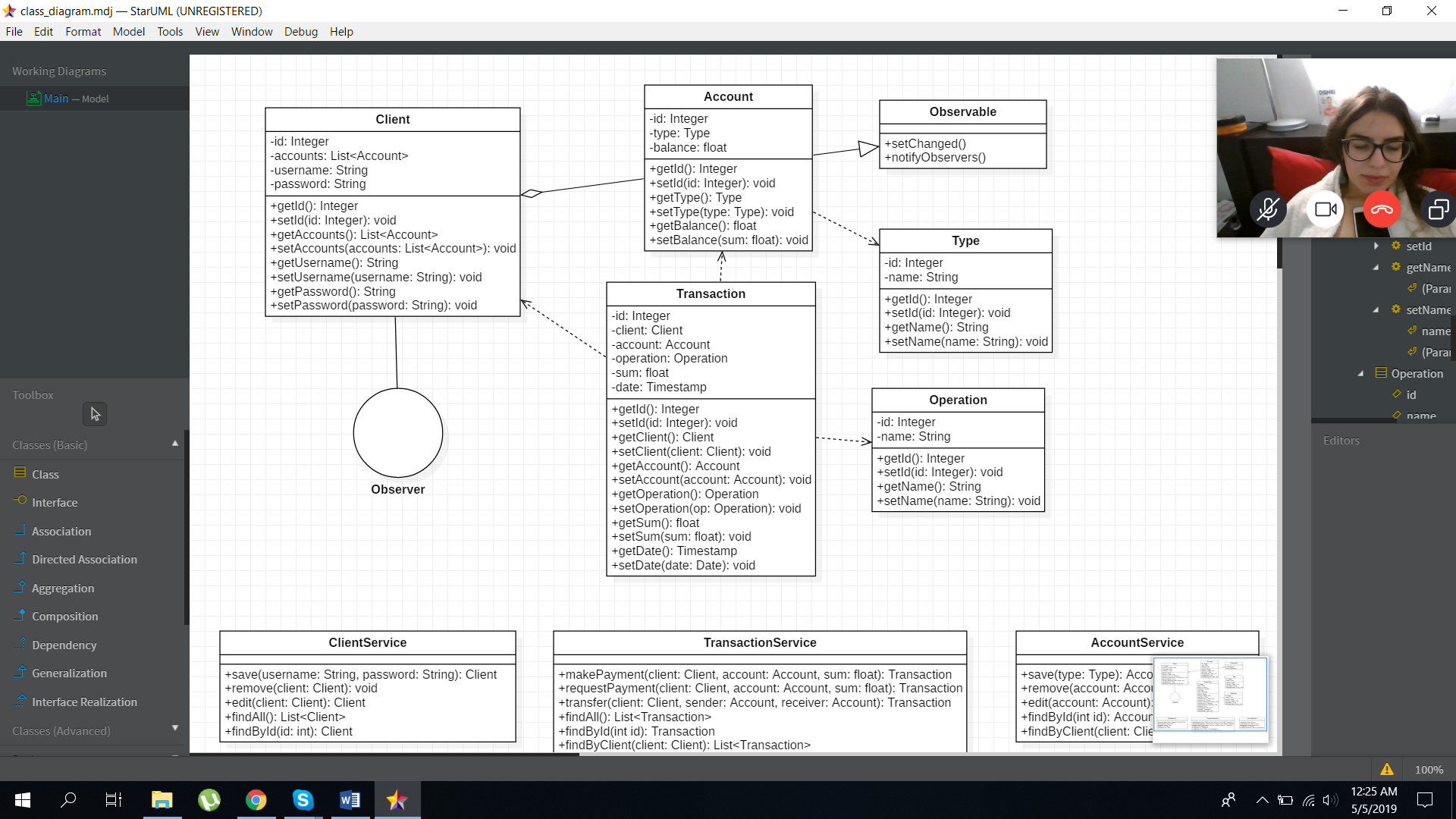
# Elaboration – Iteration 1.2

# Design Model

## Dynamic Behavior

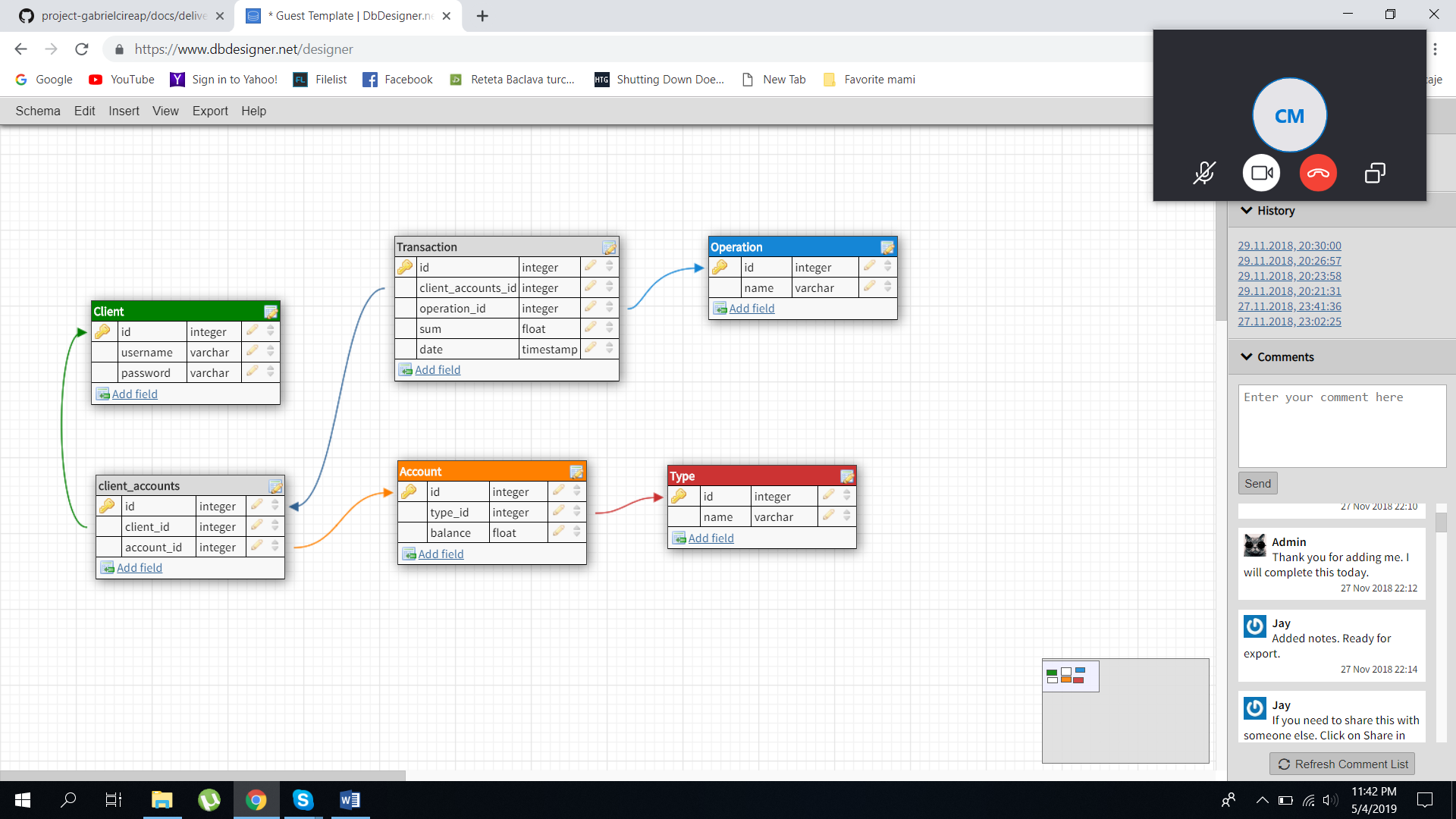
1. Send payment
2. Request a payment

## Class Design



The Design Pattern that will be applied throughout this project is the Observer Design Pattern. The pattern is used because it allows the client to view all changes made to his account (and could be notified of potential unauthorized accesses), and it is also easy to use and to understand.

# Data Model



# Test Strategy

The first steps in testing the application will be doing developer tests (the developer that writes the code usually does a little testing to ensure the proper functionality). After developer tests, unit tests using Junit or other frameworks will be used to make sure that each method runs in the expected way. After different modules of the system are implemented, component tests are applied to check the integrity of our system to date. Finally, whole system tests check the integrity and stability of the implemented system.

# Elaboration – Iteration 2

# Architectural Design Refinement

*[Refine the architectural design: conceptual architecture, package design (consider package design principles), component and deployment diagrams. Motivate the changes that have been made.]*

# Design Model Refinement

## *[Refine the UML class diagram by applying class design principles and GRASP; motivate your choices. Deliver the updated class diagrams.]*

# Construction and Transition

# System Testing

*[Describe how you applied integration testing and present the associated test case scenarios.]*

# Future improvements

*[Present future improvements for the system]*