Book Addict

Analysis and Design Document

Student: Urcan Miruna -Stefania

**Group: 30238**

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <dd/mmm/yy> | <x.x> | <details> | <name> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[**Project Specification**](#_gjdgxs) **4**

[**The purpose of this application is to allow talented people publish their books. The readers have the possibility to discover new authors and satisfy the need to read. This application encourages them to work with their skills, based on a feedback given by the reader for their books, without working directly with a publisher. The author can upload a book, write a description and its type and set a price. He can also delete a book from its account.**](#_tccxt4agdnnw) **4**

[**Elaboration – Iteration 1.1**](#_30j0zll) **4**

[**Domain Model**](#_1fob9te) **4**

[Conceptual Architecture](#_2et92p0) 5

[Package Design](#_tyjcwt) 5

[Component and Deployment Diagrams](#_3dy6vkm) 6

[**Component Diagram**](#_j6qnw75bbhsu) **6**

[**Elaboration – Iteration 1.2**](#_1t3h5sf) **7**

[**Design Model**](#_4d34og8) **7**

[Dynamic Behavior](#_2s8eyo1) 7

[Class Design](#_17dp8vu) 7

[**Data Model**](#_3rdcrjn) **7**

[**Unit Testing**](#_26in1rg) **7**

[**Elaboration – Iteration 2**](#_lnxbz9) **7**

[**Architectural Design Refinement**](#_35nkun2) **7**

[**Design Model Refinement**](#_1ksv4uv) **7**

[[Refine the UML class diagram by applying class design principles and GRASP; motivate your choices. Deliver the updated class diagrams.]](#_44sinio) 7

[**Construction and Transition**](#_2jxsxqh) **8**

[**System Testing**](#_z337ya) **8**

[**Future improvements**](#_3j2qqm3) **8**

[**Bibliography**](#_1y810tw) **8**

# Project Specification

# The purpose of this application is to allow talented people publish their books. The readers have the possibility to discover new authors and satisfy the need to read. This application encourages them to work with their skills, based on a feedback given by the reader for their books, without working directly with a publisher. The author can upload a book, write a description and its type and set a price. He can also delete a book from its account.

The readers have the opportunity to read a book online and review it, after logIn. In order to read a book, they have to pay for it. They can search for books after book type, author or filter them by price( ascending or descending). After buying a book, the book is added to their Library, so they can re-read it. They can also delete a book from Library if wanted.

The administrators of the application, after logIn, can read the reviews for the books, and handle the pay process for a book.

This application is above others because it can handle both sides of this world of books: it is for both the writers and the readers.

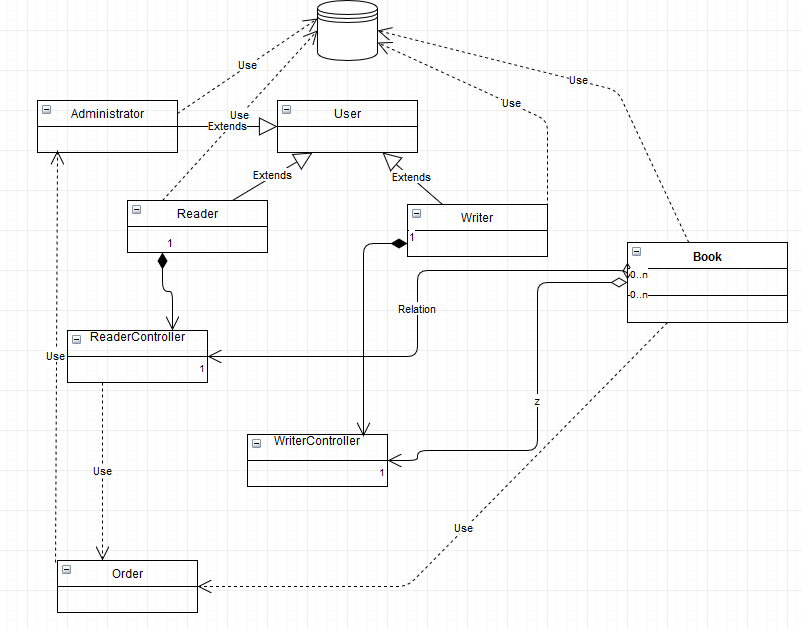
# Elaboration – Iteration 1.1

# Domain Model

*[Define the domain model and create the conceptual class diagrams]*

A domain model is a conceptual model of the domain that incorporates both behavior and data.

For this Application, we have two types of users: writers and reader. Both have access to books and all data is stored in a database.



For this application, the Domain Model has seven clases: Book, Reader, Writer, ReaderController, WriterController, Administrator, Order and User.

When a user is making an account, he can register as writer or reader. If it is registered as a reader, he is allowed to Order Books. He can read first pages before order. He can also review the book or make suggestions.

As a writer, you are allowed to publish a book of your own, as the application want to encourage the writers to publish their creations. A writer is allowed to publish a book, see the number of readers for his books and read the reviews. He can also delete a book.

The administrator of the application manages the pay methods and validate orders.

## Conceptual Architecture

*[Define the system’s conceptual architecture; use an architectural style and pattern - highlight its use and motivate your choice.]*

The application will be implemented using Layered Architecture. In order to implement a Layered Architecture, I will use four main packages: dao - the package where the connection with the database is implemented, model - package in which the data is modeled with the main attributes necessary to implement the operations accordingly. The next package is view, where the graphical interface is made in order to make the access to the application easier for the users. The last package is Controller, where all operations are implemented, in order to have the features needed by users to enjoy the application.

## Package Design

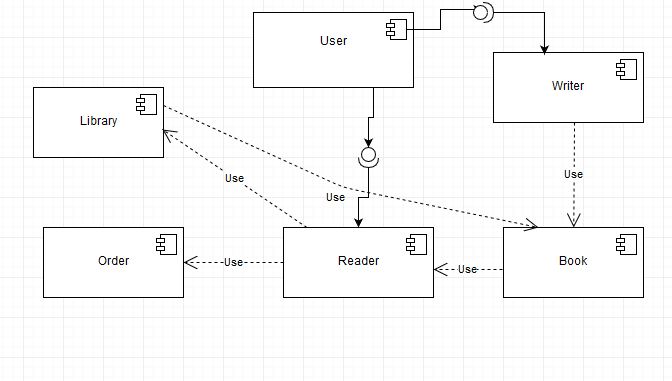
*[Create a package diagram]*

# 

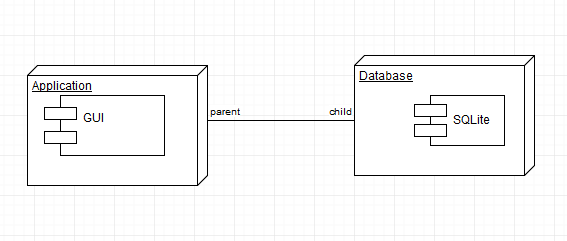
## Component and Deployment Diagrams

*[Create the component and deployment diagrams.]*

# Component Diagram



Deployment Diagram



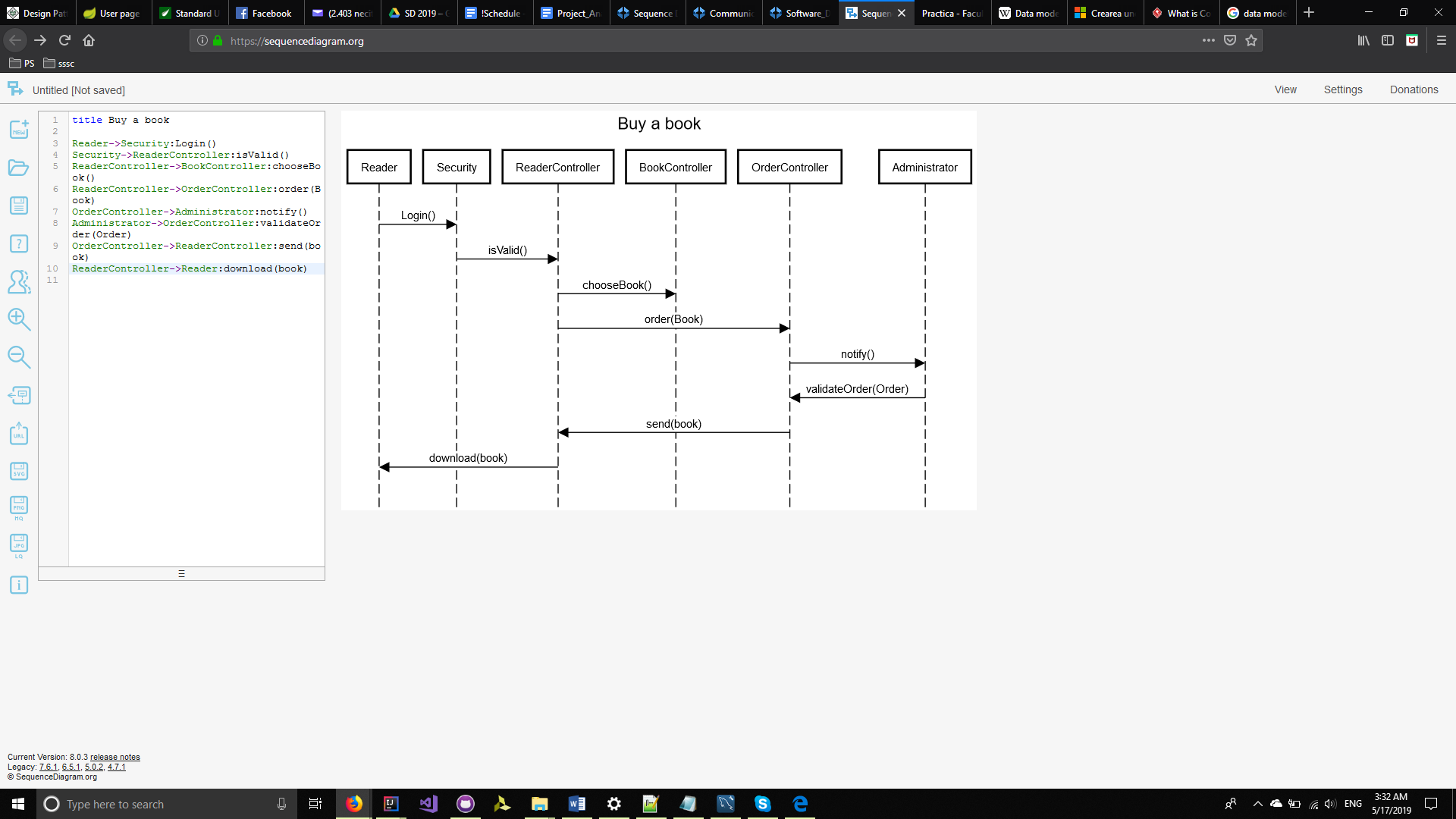
# Elaboration – Iteration 1.2

# Design Model

## Dynamic Behavior

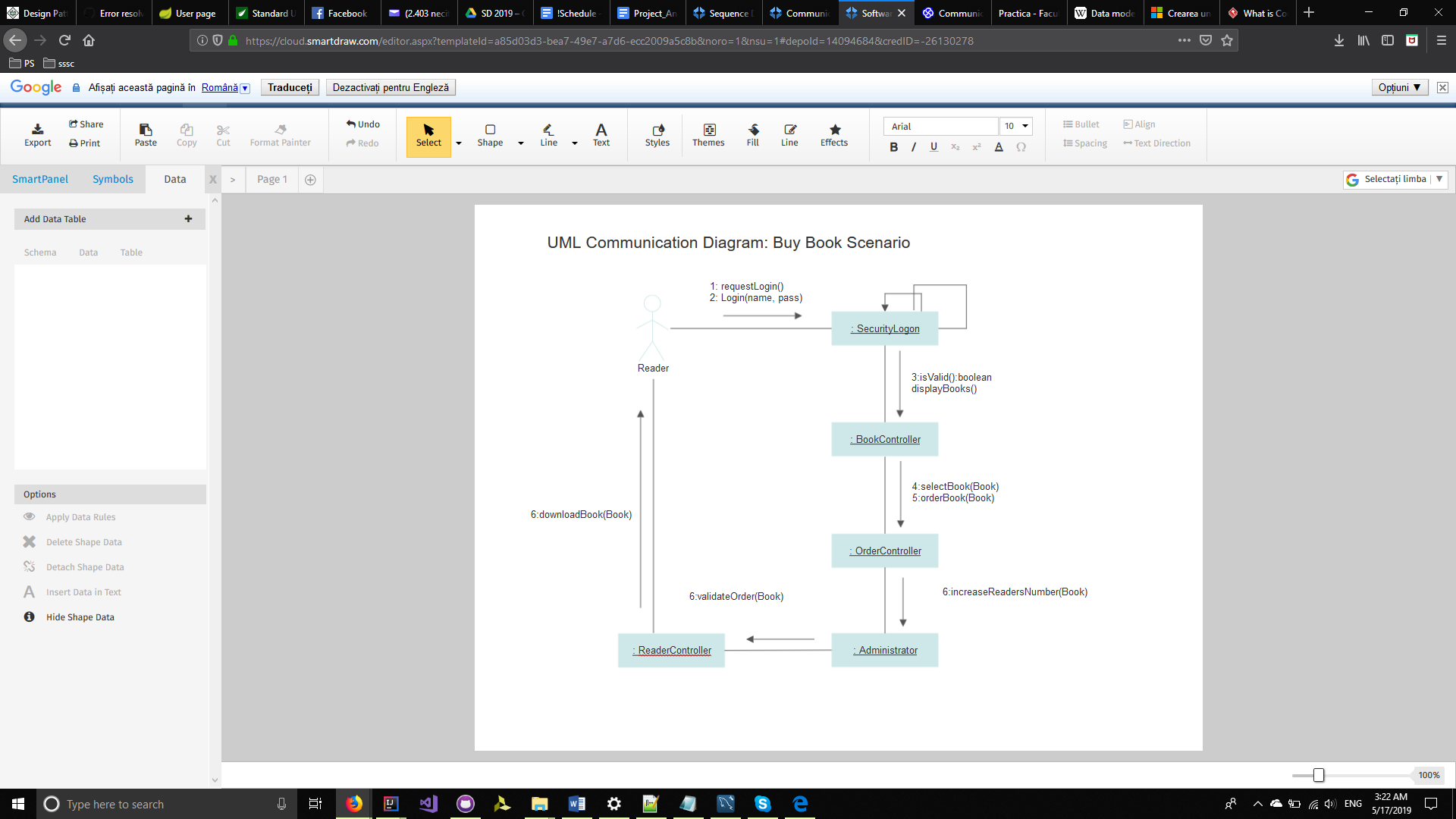
*[Create the interaction diagrams (1 sequence, 1 communication diagrams) for 2 relevant scenarios]*

Sequence diagrams describe interactions among classes in terms of an exchange of messages over time ad it shows object interactions arranged in time sequence.

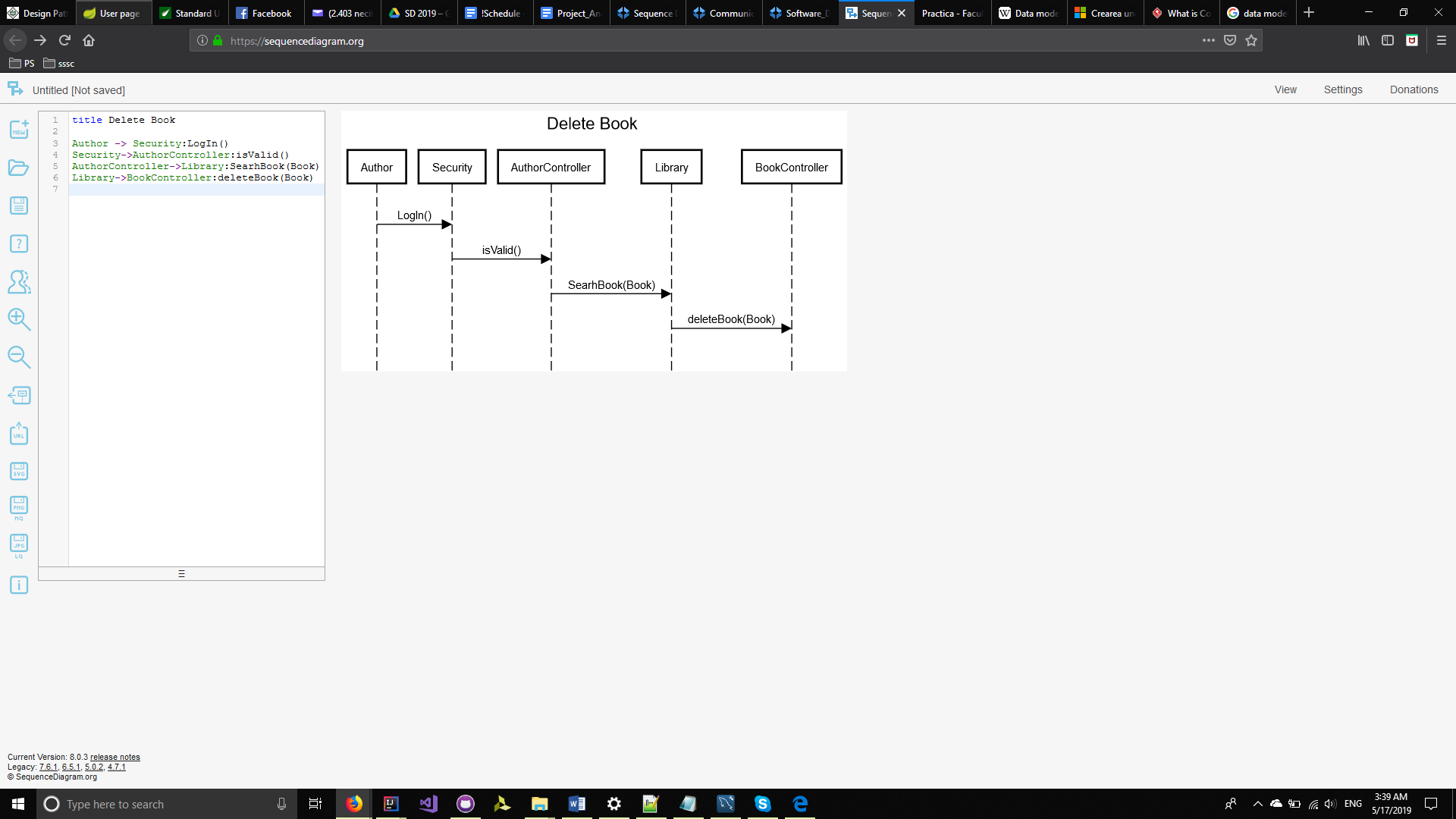


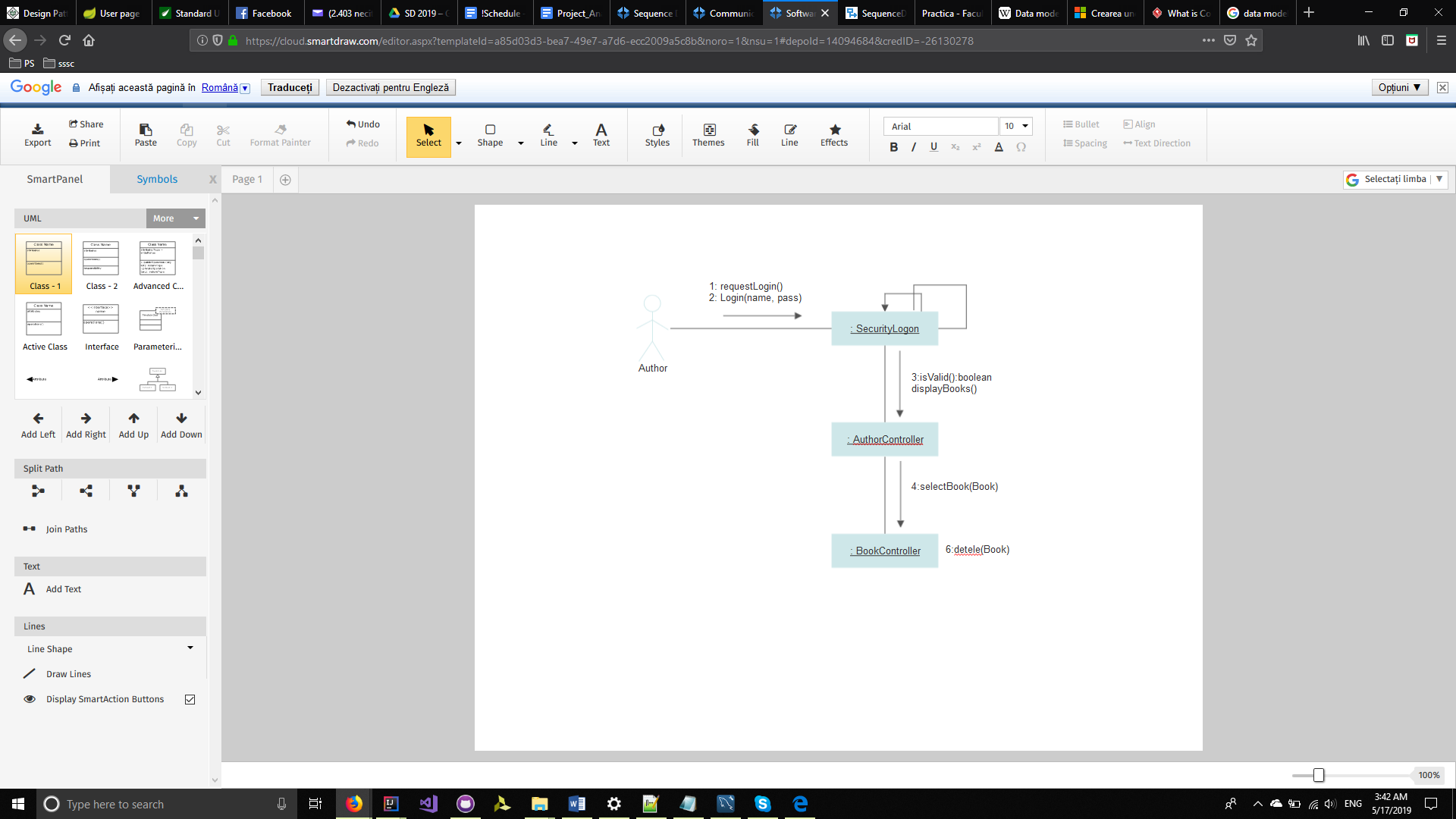
Sequence diagram :The reader buys a book

UML communication diagrams, like the sequence diagrams - a kind of interaction diagram, shows how objects interact.



An author can delete one of its books from his library.

****

****

## Class Design

*[Create the UML class diagram; apply GoF patterns and motivate your choice]*

For implementation, one of the patterns that I will use is Observer Design Pattern. In this case, when a reader is ordering a book, the administrators will be notified. The order has to be validated by an administrator user. Only after it is validated, the book can be added to the library of the reader.

# Data Model

*[Create the data model for the system.]*

# Unit Testing

*[Present the used testing methods and the associated test case scenarios.]*

# 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.]*

In order to test the system, we will implement test benches, which means that our application will test every operation in particular. Therefor, the advantage of using test benches is that you know exactly what operation in which case is not working, improving the quality of application.

# Future improvements

The application can be improved by adding the possibility for the administrators to filter the review, in order to make recommandation when a reader is looking for a book of its type or author. An improvement it will be if the authors could communicate with the readers in order to exchange their ideas, make suggestions and collaborate.

# Bibliography