Toegepaste Informatica

1TX

2022-2023

Back-End Development

Assignment Book: Introduction – User Story & Class Diagram

G. Jongen, J. Pieck, E. Steegmans, B. Van Impe

Overview

1. Intro to the project
2. User Story
3. Class Diagram

Intro to the project

Scope

Organization

Application “Book”

Product Goal

- Create the back-end for a book registration service.
- Books can be added, deleted, updated, searched….
- The delivered product must be working

Learning Goal

- Create an application in Spring Boot that implements the given user stories
- Use the technologies as learned in class in an open project

Remark

- Labo “User” will merge with ”Book” later on

Organization & Feedback

- Make a team consisting of 2 students (Toledo)
- Share your project on the GitHub repo created by our GitHub Classroom Assignment
- Get feedback on Tuesday during lesson

Evaluation

- Get score during the evaluation moments, based on the evaluation criteria found on toledo.
- Use the project as documentation during your exam
- More info: cfr presentation lesson 1

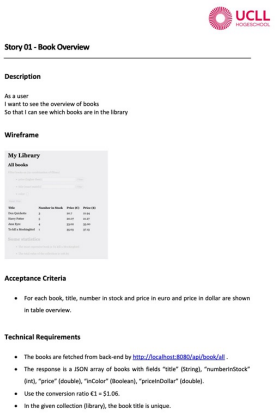
User Story

Definition

How to use

The project “book” is more open. Starting from a user story, you need to determine the structure of your project: which classes and methods are needed? How do I ensure that REST controller passes the right data to obtain the desired response?

User Story



Story 01 - Book Overview

Description

As a user
I want to see the overview of books
So that I can see which books are in the library

Wireframe

My Library
All books

Acceptance Criteria

- For each book, title, number in stock and price in euro and price in dollar are shown in table overview.

Technical Requirements

- The books are fetched from back-end by http://localhost:8080/api/book/all .
- The response is a JSON array of books with fields "title" (String), "numberInStock" (int), "price" (double), "inColor" (Boolean), "priceInDollar" (double).
- Use the conversion ratio €1 = $1.06.
- In the given collection (library), the book title is unique.

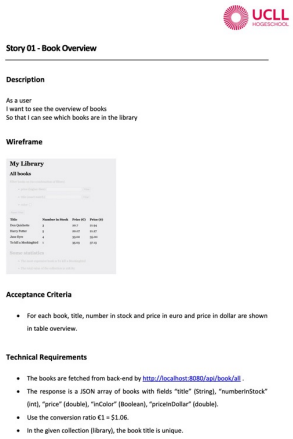
In software development and product management, a user story is an informal, natural language description of features of a software system.

They are written from the perspective of an end user or user of a system.

https://en.wikipedia.org/wiki/User_story

We start with the definition of a user story. As Wikipedia says, it is a description of features of a software system. The description is written in common readable language, so no technical terms are used.
The user story describes what the end user expects and is therefore result-oriented.

User Story



UCLL HOGESCHOOL

Story 01 - Book Overview

Description

As a user
I want to see the overview of books
So that I can see which books are in the library

Wireframe

My Library
All books

Acceptance Criteria

- For each book, title, number in stock and price in euro and price in dollar are shown in table overview.

Technical Requirements

- The books are fetched from back-end by http://localhost:8080/api/book/all .
- The response is a JSON array of books with fields "title" (String), "numberInStock" (int), "price" (double), "inColor" (Boolean), "priceInDollar" (double).
- Use the conversion ratio €1 = $1.06.
- In the given collection (library), the book title is unique.

Description

- As …: who is the end user
- I want to …: what task does he want to perform
- So that …: what is the result when the task is completed

Acceptance Criteria

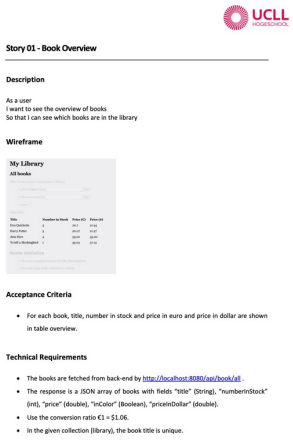
- Must all be fulfilled

Technical Requirements

- Technical information for developer
- Can include API contract

The description has a fixed structure: as … i want to … so that …. . Who is the end user? What task does he want to perform? What is the result when the task is completed?
The acceptance criteria describe detailed and measurable the goal of the story.
Finally, technical requirements can be added. They can give technical information how the story can/must be completed.

Use the Story



Story 01 - Book Overview

Description

As a user
I want to see the overview of books
So that I can see which books are in the library

Wireframe

My Library
All books

Acceptance Criteria

- For each book, title, number in stock and price in euro and price in dollar are shown in table overview.

Technical Requirements

- The books are fetched from back-end by http://localhost:8080/api/book/all .
- The response is a JSON array of books with fields "title" (String), "numberInStock" (int), "price" (double), "inColor" (Boolean), "priceInDollar" (double).
- Use the conversion ratio €1 = $1.06.
- In the given collection (library), the book title is unique.

- Find the underlying model
  - Classes "Book" and "BookService" for collection of books
  - Book must at least have properties for title, number in stock, price
  - Price in dollar must be calculated
- Get the requests your application must handle
  - api/book/all
- Get the format of the expected response (API)
  - Json with fields as described

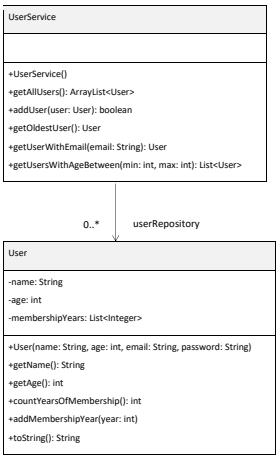
As a developer, you should use the story to determine the underlying model, in particular what classes are needed and what properties the objects should definitely have.
You will also find the requests your application needs to handle and details about the output: which format (e.g. list or single object), which fields, ...

Class Diagram

Definition and structure

You now have to technically translate the information derived from the story into domain classes.

Class Diagram



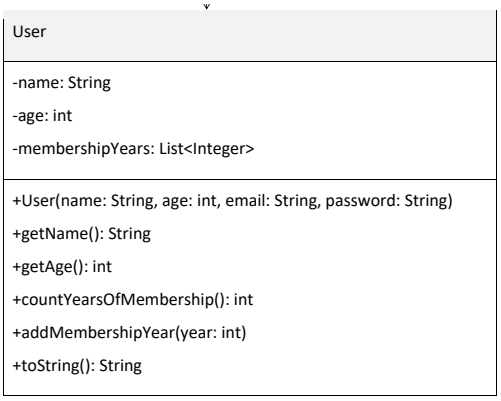
In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram

that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects

https://en.wikipedia.org/wiki/Class_diagram

For each class, you need to detail which fields (attributes) there are, what type they are of, what their scope is. For methods, you need to indicate what parameters there are (including type) and what their return value is (if any). Finally, you also need to define the relationship between the different classes.
We use a class diagram to do this.

Class Diagram



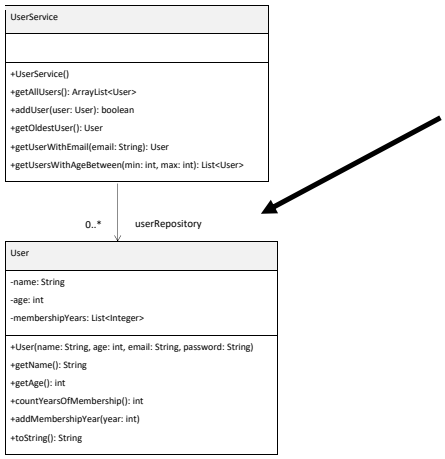
← Class Name

← Fields described as
- Visibility: "-" for private; "+" for public
- Field name
- Field type

← Methods
- Access: "-" for private; "+" for public
- Method name
- Between parantheses: parameters with type
- After ":": return value

You write the name of a class with a capital letter.
For each field list the visibility (with symbol "-" for private access and "+" for public access), the name (lower case) and type.
For a method, you specify the visibility, the method name (lower case except constructors), a list of parameters with their type and the return value (use void if there is none).

Class Diagram: Relationship between classes



"UserService"
- Has a field with name "userRepository"
- With zero or more instances (multiplicity 0…*) of "User"

Multiplicities

0	No instances (rare)
0..1	No instances, or one instance
1	Exactly one instance
1..1	Exactly one instance
0..*	Zero or more instances
*	Zero or more instances
1..*	One or more instances

Finally, the class diagram also describes the relationship between different classes. Along the (full) arrow, write the name of the field that contains the related objects.
The multiplicity stands for the range of number of objects that participate in the relationship.
In this lesson, we will limit ourselves to a one-to-many relationship: one UserService can contain multiple Users. In subsequent lessons, we will describe other types of relationship.
We will also restrict ourselves to association what implies that one object has the other object as a field. Other types of relationships will be covered later.

Next step

Follow the instructions of the task “Opdracht”.