# Template code review

## Overview

**RenewService -** service which holds logic of renew process, all constructor arguments are injected using Spring dependency injection container. So class is automatically instantiated/

**ReturnService -** service which holds logic of return process.  Again, all constructor arguments are automatically injected.

## Goals

Encapsulate and implement all logic of Return and Renew operations

## Usage Scenarios

Return service:

- User can initiate return process
- Librarian can confirm return of the document

Renew Service:

- User can renew taken document

## Class Diagrams

## Code

## Categories for review

**1. Design decisions**

**2. API design**

**3. Architecture (including inheritance hierarchy, if any)**

**4. Implementation techniques, in particular choice of data structures and algorithms**

**5. Exceptions handling - Contracts**

**6. Programming style, names**

**7. Comments and documentation**

1 Programming style, names

[1.0] Reviewer1: Nice work, but as Yyy said "Hello! Every code have a piece to be improved", after recollecting Yyy's quote, I have a suggestion:

[1.1] Reviewer1: Use additional constant for VP renew time(line 62), instead  of WEEK.

[1.1.1] Owner: Week is amount of time which can be applied not only to VP, but also to feature roles

[1.1.2] Reviewer1: But if we wanna to change amount of time of renew only for VP ? Maybe it’s simpler to change one additional constant, which corresponds to renew time

for VP , instead of searching 62’nd line of code and changing this line of code. WEEK shouldn’t be deleted (it may be useful).
[1.1.3] Owner: I see your point, it might be changed [FIXED]

2. Programming style, names
   [2.0] Reviewer1: As Xxx said: "Why calculate the integral, if the answer is clearly understandable?". Following his words I suggest the improvement:
   [2.1] Reviewer1: line 18 , it’s clearly understandable that checkoutService is checkout service. Maybe this line of code - useless ?
   [2.1.1] Owner: It’s Javadoc, so it must cover all of the fields.
   [2.1.2] Reviewer1: ok
   [2.1.3] Owner: [closed]
3. Programming style
   [3.0] Reviewer1: Use tabs instead of spaces. As Eiffel studio and one of the best object oriented programming language shows, they are more convenient.
   [3.0.1] Owner: Java code conventions doesn’t specify which type of indentation to use: tabs or spaces, but in case of usage tabs, they should be 8 spaces, so i find it terrible. [closed]

4. Implementation techniques, in particular choice of data structures and algorithms.
   [3.0] Reviewer1:  It's better to use the LocalDate class (Java 8), instead of outdated Date.
   [3.0.1] Owner: Date still is not deprecated, so it’s okay to use it. But the LocalDate got nice API, i will think about it.
   [3.0.2] Reviewer1: There are exist group of people, who still told that COBOL is not deprecated, so, think about it )
   [3.0.3] Owner: [closed]