Assignment

| ANALYSIS 3: OBJECT-ORIENTED MODELING (INFANL03-3 | INFANL23-3) |
| --- |

Educational Period 3 [2021-22]

| Public Library System | 📚 |
| --- | --- |

This assignment consists of the design and implementation of a Public Library System, which will be called PLS, afterward. The required functionalities of the system are according to the descriptions provided in this document, which might be slightly different from a real public library system; rather, these functionalities are defined according to the course intended learning outcome and assessment objectives. Such a library system usually involves a database, but here the assignment is limited to storing data in RAM and files. The system will consist of a console based (textual) interface that gives access to all functions of the system.

| Learning Objectives |
| --- |

## 

The learning objectives of the assignment and mapping to the intended learning outcome of the course are listed below:

1. To demonstrate the knowledge of object-oriented modeling in three phases of SDLC, including Analysis, Design, and Development (LO1, LO2, LO3).
2. To apply the knowledge of object-oriented modeling in a real-world problem (LO1, LO2, LO3, LO4).
3. To apply the knowledge of file processing in a real-world problem (LO5).

| Requirements of the System |
| --- |

## 

The whole issue of security is left out of scope of this assignment, unless explicitly stated in the requirements.

PLS should have a main interface for users in the console. With this interface, a user, either the **library administrator** (called admin afterward) or a **member** can login to the system. No option is available to users, if they are not logged in, yet.

## Library admin

Admin’s credentials should be defined (hardcoded), as below:

Username = “**admin**”

Password = “**admin123**”

(Admin’s username and password are used by your teacher to check the system and grade your assignment. Make sure they are correctly defined).

The minimum required functions for administrator are listed below:

* Functions related to members:
  + To see the list of members
  + To add, edit, and delete a member from the system
  + To check the status of book items loaned by a member
  + To load and add a list of members to the system, all at once (using a file)
* Functions related to catalog:
  + To check the catalog (list of books)
  + To add, edit, and delete a book to/from the catalog
  + To search a book item in the catalog
  + To load and add a list of books to the catalog, all at once (using a file)
* Functions related to book items:
  + To see the list of book items in the library
  + To add, edit, and delete a book item to/from the library
  + To search a book item and its availability in the catalog
  + To lend a book item to a member
* Functions related to system administration:
  + To make backups of the system
  + To restore a specific backup of the system

## Members

A member can be registered in the system by the admin. Username and password of members are also defined by the admin. Then, a member can use them to login into the system.

The minimum required functions for members are listed below:

* Functions related to catalog:
  + To check the catalog (list of books)
  + To search a book item in the catalog
* Functions related to book items:
  + To see the list of book items in the library
  + To search a book item and its availability in the catalog
  + To loan a book item
  + To return a loaned book item

## System Requirements

* Configuration of the system:
  + When the system starts, it must check the data file(s) for catalog, book items, members, etc.
  + If the data file(s) are already there, the system should recognise it, otherwise the system may need to properly initialize and configure the data file(s).
* Requirements related to loan administration:
  + A member can borrow a book for a maximum period of 1 month
  + A member can borrow a maximum of 3 books simultaneously
  + Book search function must accept book title or author as search key
  + Book search function must accept partial keys, lowercase, and uppercase letters (it should not be case-sensitive). For example, a search with author= “Hem” or “hemi” or “ming” must return “Ernest Hemingway”, if such author exists in the data file.
* Requirements related to system administration:
  + Name of backup files should contain the date of backup, plus a serial number for the multiple backup on the same date
* Other requirements:
  + File All data of PLS must be saved after every transaction of the system.
  + System should be able to prevent some obvious errors. For example, a member shouldn’t be repeated in the system, if such a user is already registered; or a book (or book item) should not be duplicated in the catalog (or library).
  + A member must not be allowed to loan the same book item more than once.
  + Member’s username and password are defined by library administrator
  + Username and password do not need any special formatting requirements, except a username must be lowercase. For example, a valid username can be “mike1998”, not “Mike1998”.
  + Menus should be chosen by one digit or one letter, only. For example:

| Enter 1 or 2 to choose a menu:  [1] Add member  [2] Search Book |
| --- |

Do not ask the user to enter “add member” or “add” to choose the menu.

| Two Phases of the Assignment |
| --- |

## 

The assignment is divided into two phases:

### Phase 1: Formative Assignment

In this phase, you can complete [components 1 and 2](#_um5bb54h8tci) described in the next section. In this phase, you will receive feedback from your instructor. This feedback will help you to ensure you are on the right track for the rest of the project. **This submission will not be graded**.

### Phase 2: Summative Assignment

In the second phase of your assignment, you must complete and deliver all components of the assignment ([All components 1, 2, 3, and 4](#_um5bb54h8tci) must be included in your final submitted file).

**This submission will be graded for a PASS or FAIL**.

| Your Assignment |
| --- |

## 

## Components

The assignment consists of four essential components:

1. **Software Requirements Specification (SRS)**

* Requirements Engineering (Elicitation, Verification, Validation, Documentation)

1. **System Design**

* Use Case diagram(s) [preferably implemented in Astah]
* Class Diagram(s) [preferably implemented in Astah]
  + Astah is recommended for the diagrams, but if you prefer other tools, you need to properly include the diagrams in your documents. See the [deliverables](#_nk2f1zks73de).
* explanation of the design in natural language.

1. **Development of the PLS in Python**
2. **A User Manual for the PLS**

## Software Requirements Specification

The **Requirements** document should cover at least the following stakeholders:

1. Library Administrator
2. Members

This document needs to demonstrate various stages of requirements engineering, including elicitation, analysis for verification, analysis for validation, and documentation). More explicitly, you need to clearly specify types and priorities of requirements.

All requirements must be correctly considered and implemented in the design and development phases. **Any mismatch between the requirements, design and implementation is considered as a negative point**.

## Design of the System

The design must consist of **UML Class Diagram(s)** and **Use Case diagram(s)**. Additionally, a short description of each diagram and their components, functionalities, justification, etc. is needed.

The Class Diagram **must** at least cover the following classes:

1. Catalog (contains the list of books)
2. Book (contains information of a book in the catalog)
3. BookItem (a paper copy of a book in the library)
4. LoanItem (a book which is loaned to a member)
5. Person (a general class for a stakeholders of PLS)
6. LibraryAdmin (a person which is defined as library admin, with specific responsibilities)
7. Member (a person who is a member of library)

You can define any additional classes if needed (but you should provide justification for it).

The relationships between the classes should be correctly and clearly depicted in the diagram. All attributes and methods of each class must be stated.

The **Use Case diagram** should cover all stakeholders that have direct access to the system, and all the main functions of the system.

**All components of your design must be accordingly implemented in the code. Any mismatch between the design and implementation is considered as an unsatisfactory design or implementation (check** [**grading marking Schemes**](#_bt9v74kheh72)**).**

## Developed System and its Functionalities

The implementation should consist of a **Python** program which is clearly an implementation of the design. It should consist of two main parts: the backend and the frontend. The backend should implement all data processing and data storage and retrieval. The frontend should be a user interface, which must be a console based (textual) interface, giving access to all the functions of the system. Additional classes, attributes and methods are allowed. **The system must be built only in Python 3 using only modules in the standard library of Python 3**.

You are not allowed to use any database management system, such as SQLite, MySQL, etc. Working directly with files is a requirement of the assignment.

All functional and non-functional requirements of the system are implicitly or explicitly provided throughout this document. It is your task to carefully read this document to elicit the requirements of the system.

## User Manual

A User Manual should be provided to explain how to use the system, for both the librarian and for subscribers. Any information needed for users of the system should be clearly provided in this document, including commands, shortcuts, usernames and passwords (if any), etc. This manual may contain screenshots, examples, diagrams. This information may also be used by your teacher to run, test, and assess the system.

**Do not suppose that the teacher will explore your code to find out how to run the system.**

| Submission |
| --- |

## 

## Formative Assignment:

For the formative assignment, you need to only submit two components of the system:

1. Software Requirements Specification
2. System Design

## Summative Assignment:

For the summative assignment, you need to submit all components of the system:

1. Software Requirements Specification
2. System Design
3. Developed system in Python
4. User manual

## Submission package

The delivery to be handed in must consist of **one zip-file**, named as below, depends of number of students in the group:

***studentnumber1****.****zip* (individual submission)**

***studentnumber1\_studentnumber2****.****zip* (group of 2 students)**

***studentnumber1\_studentnumber2\_studentnumber3****.****zip* (group of 3 students)**

The zip-file must contain:

1. A **pdf document**, called **PLS-Documentation.pdf**, containing:
   1. **Names** and **student numbers** of the team (maximum **3** students per team),
   2. The **Software Requirements Specification**,
   3. The **Class Diagram**,
   4. The **Use Case diagram**,
   5. [only for Summative] A **User Manual** explaining, with examples, how to use the system, for both the admin and members. **This is needed for your teacher to test and assess your code.**
2. [only for Summative] A directory called **PLS-SourceFiles**, containing all the **code files** and the **data files**, including one main file **PLS.py**. Starting the system should be done by running **PLS.py**.

## How to submit?

* Only submit it via the appropriate channels in MS Teams defined for your class.
* Students who re-sit the assignment from the last year need to be added to MS Teams. You can request it from the course coordinators, by Email or Chat on MS Teams.

**Important:** Do not send your assignment by email or chat or any other inaccurate channel, or it may be lost. Teachers are not responsible for a lost assignment submitted via other channels. If there is any problem in submission through MS Teams, you can contact your teacher (retakes can contact course coordinators). Your teacher or coordinators then guide you how to submit it.

## Deadlines

Formative Assignment: Submission deadline is **11 March 2022**.

Summative Assignment: Submission deadline is **16 April 2022**.

| ⚠ IMPORTANT NOTES  1. **Do not** include any **bulky** Python system files in the delivery. 2. The code must **only** use **standard library modules**. 3. The code must run **error-free**. 4. The code should **only** write to a **temporary storage** on the current (running) folder or a subfolder of it. 5. We encourage you to work in a **team of 2 or 3 persons**. However, individual work is also acceptable, if you prefer to do it individually, or you are not able to make a team (especially re-takers). 6. When working in a team, **only one team member (the team leader)** submits the assignment, and all group members submit a group-info message with names and student numbers of the entire team, clearly indicating who is the team leader. **Feedback will be given to the team leader only**, who will then communicate it to the other team members. 7. You are **not allowed** to work together with someone who has a different teacher! Here we mean the teacher in your schedule. If you are not sure, ask your teacher to advise you. |
| --- |

| Grading |
| --- |

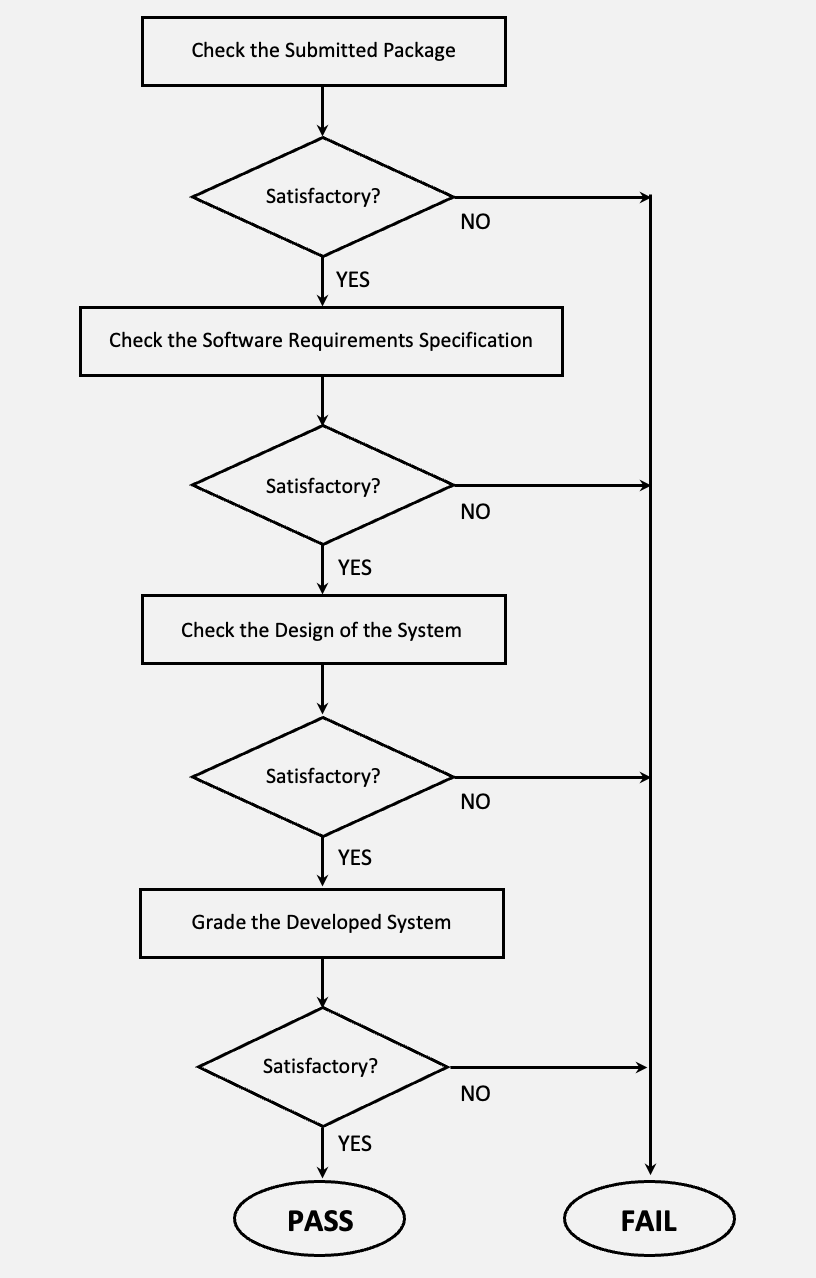
## 

The assignment will be evaluated as either **PASS** or **FAIL**. To successfully pass the course, students must pass the assignment together with passing the exam.

## Need a presentation?

Teacher may ask you for a presentation, if needed. In this case, attending this presentation is mandatory for all team members, in order to get a PASS.

## Grading Flowchart

Your assignment will be graded according to the following flowchart. Please note that if the submission is not satisfactory at any level, other criteria will not be checked anymore.

Assignment requirements:

1. Submitted package
2. Requirements document
3. Design of the system
4. Developed system

The requirements for each criteria are briefly explained, in the following sections.

This [Excel file](https://docs.google.com/spreadsheets/d/13RwnFdBfgZiH4nUN11UhYvEeg93m4Akr?rtpof=true&authuser=bashb%40hr.nl&usp=drive_fs) will be used by your teacher for grading. Download to see how it works.

## Marking Scheme

### Submitted package

The submitted package must fulfill the following criteria:

| **Criteria Description** | | **Satisfactory if:** | |
| --- | --- | --- | --- |
| C1.1 | Group composition and information | Group is composed of max. 3 persons.  Group members are from the same class.  Students' names and numbers are provided. | |
| C1.2 | Formats and names of files | One zip file is submitted, with the correct name format.  The zip file contains only: **PLS-SourceFiles** folder, **PLS.py**, and **PLS-Document.pdf**.  Any other files are located inside the PLS-SourceFiles folder. | |
| C1.3 | Completeness of documents | All required parts are included in **PLS-Document.pdf** file (Software Requirements Specification, Class Diagram, Use Case Diagram, User manual) | |
| C1.4 | User Manual | User manual contains clear instructions for users to run and use the system. | |

Please note that this criteria is not relevant to the learning objectives of the assignment. However, to ensure a fair grading system throughout all classes, these criteria will be strictly checked by teachers. Hence, even though the above criteria are technical requirements, failing any of them will lead to FAIL the assignment.

### Software Requirements Specification

The Software Requirements Specification must fulfill the following criteria:

| **Criteria Description** | | **Satisfactory if:** | |
| --- | --- | --- | --- |
| C2.1 | Stakeholders | All stakeholders (library administrator and members) are covered in the Software Requirements Specification. | |
| C2.2 | List of requirements | All requirements and constraints given in the assignment description are properly formulated.  All requirements and constraints are correct, complete, relevant, consistent, unambiguous, and testable.  All requirements and constraints are tagged. | |
| C2.3 | Categorization of requirements | Types of requirements are provided. | |
| C2.4 | Priority of requirements | Requirements are prioritized. | |

### Design of the System

The Design of the System must fulfill the following criteria:

| **Criteria Description** | | **Satisfactory if:** | |
| --- | --- | --- | --- |
| C3.1 | Class diagram | All required classes are included in the diagram.  All attributes and methods of all classes are included in the diagram.  All relations are correctly designed and depicted in the diagram. | |
| C3.2 | Use case diagram | All required use cases (functions) are included in the diagram.  All actors are included in the diagram.  All relations are correctly designed and depicted in the diagram. | |
| C3.3 | Object-Oriented design | There is evidence of using an object-oriented paradigm in the design of the system. For example, there is evidence of abstraction, encapsulation, and inheritance. | |
| C3.4 | Conformity | The design of the system complies with the requirements in SRS. | |

### 

### Developed System

The Developed System must fulfill the following criteria:

| **Criteria Description** | | **Satisfactory if:** | |
| --- | --- | --- | --- |
| C4.1 | Execution of the code | Code runs smoothly, and error-free.  The code is platform-independent, and runs on both on a Macbook or Windows machine. | |
| C4.2 | User interface | User interface provides user-friendly text-based menus in the console.  All required functionalities of the system are available through menus.  Two different menus for members and library administrators are provided, with different access levels on functions.  Sufficient information provided for the users. | |
| C4.3 | Functionality of the system | All required functionalities are fully implemented and operational without serious bugs or errors. | |
| C4.4 | Conformity | The developed code in Python complies with the design of the system. | |
| C4.5 | OOP principles | There is evidence of Object oriented programming in the code. For example, methods are encapsulated in classes, or use of global variables as few as possible, etc. | |

## Common Mistakes!

1. Do not simply copy and paste requirements from the assignment description to build SRS. Requirements of the assignment are different from requirements of the PLS system.
2. Do not just simply write the task of each menu as a user manual. It is not called a user manual. For example, the following explanation is not accepted as a user manual:

| **Add member:** this menu is used to add a member to the system.  **Add book:** this menu is used to add a book to the system. |
| --- |

This is not a user manual. Rather, it needs to explain how to use this menu with details (examples, screenshots, etc). For example what is allowed to be entered by the user, what is the format of username to be chosen, etc.

1. Make sure all three stages (requirement engineering, design, and development) comply with each other. If a class is designed in the class diagram, the same class with all attributes and methods in the design must be included in the code.
2. Do not write on any file or folder, except the current folder. For example, you are not to use such a commande in your code:

| **open("c:\plsbackup.csv", 'w')** |
| --- |

This is a serious offensive code, due to security concerns and leads to an immediate FAIL of the assignment.

1. All data of PLS must be saved after every transaction of the system. For example, if a new member or a new book is added to the system, after restarting the PLS system, this user or book must be valid. Keeping information in the RAM is not sufficient.

## Academic Integrity

Academic integrity means avoiding plagiarism and cheating, among other misconduct behaviours.

You are fully responsible for your submission. By submitting your assignment file, you confirm that all parts of your assignment are your own original work (except those which are explicitly stated and allowed in this assignment description). Those which are not requirements of the assignment could be used with proper citation of the reference. If you are not sure about it, you can always consult with your teacher.

Copying any part of this assignment , such as documents, figures, diagrams, requirements specification, python codes, user manual, etc. (from any sources, internet, books, weblogs, past year submitted works which are not yours) is strictly prohibited and will be considered as plagiarism. Any kind of academic dishonesty and plagiarism will lead to a FAIL for this course, and will be reported to the exam committee for further investigation and action.

You are also responsible to keep your work safe and private. In case of two similar submitted works, both submissions will be considered as plagiarism cases.

## Feedback

Students will receive short feedback from the teachers within three working weeks. In this feedback, not all details of grading could be explained. **Only the essential reason(s) for a FAIL will be shortly addressed**.

| **Important:**  For those who unfortunately got a FAIL, please note that teachers will provide you only with the feedback about which criteria you failed and why. In this case, the other criteria might not be checked, because if some mandatory criteria are unsatisfactory, the rest won't be checked anymore.  It means that there is no guarantee to get a PASS in your resit/retake, by fixing only the issues addressed in the feedback by the teacher. Rather, you must ensure all criteria (in addition to the issues addressed in the feedback) are satisfactory in the resit/retake submission. |
| --- |

## Any concern about your grade?

We advise you to carefully read the feedback given by your teacher and check if it complies with the detailed explanation of the grading scheme.

If you still have a valid and arguable concern about your grade, you can directly contact your teacher via email, within 5 working days after you received your feedback. After this period, grades will be uploaded to Osiris.