Software Requirements Specification

for

Parson's Problems Appliance for Learning Management Systems (PPALMS)

Version 2.0

Prepared by Al Yaqdhan Al Maawali, Mulki Yusuf, Ahmed Al Raisi, Fahia Tabassum

CSCI 5801, College of Science and Engineering, University of Minnesota-Twin Cities

October 17th, 2022

Table Of Contents

Revision History

Name	**Date**	**Reason For Changes**	**Version**
AlYaqdhan	10/17/22	Updates based on the organization and completeness section of the review report.	2.0
Ahmed	10/17/22	Updates based on the special issues section of the review report.	2.0
Fahia	10/17/22	Updates based on the correctness of the review report.	2.0
Mulki	10/17/22	Update based on the quality attribute of the review report.	2.0

1. Introduction

1.1 Purpose
The purpose of this document is to show the user requirement specification to the intended system. The document includes an overview of the system, an overall description of the system, interface requirements, features, and other requirements.

1.2 Document Conventions
Document is numbered by content and titles are in bold. Hyperlinks are bold, blue, and underlined.

1.3 Intended Audience and Reading Suggestions
The document is written in user language for all to understand. It is intended for the developers and users of the system to read. Both developers and users are suggested to read the document starting from the first section to the last.

1.4 Product Scope
The PPALMS system is a system to be used by instructors to assess students' knowledge on code writing exams. The system will exclude students from writing code from scratch and instead make them rearrange/match pieces of written code. The goal of the system is to give students problems that will prepare them better for the industry.

1.5 References
- use-cases
- Requirements Review
- Review Report Checklist

2. Overall Description

2.1 Product Perspective
PPALMS [1] is a new, self-contained product.

2.2 Product Functions
- Select source code
- Annotate lines for inclusion/exclusion
- Form and order line tuples
- Select question type
- Generate all possible variations of the source code matching question type
- Select intended LMS [2] target

- Create an appropriate output file
- Allow reuse of problem

2.3 User Classes and Characteristics
There are 3 user classes: Instructors, researchers and TAs. The most important user classes for this system are the instructors and TAs. Instructors and TAs have the exact same access and role. They can access anything the system is able to do. However, researchers would only make use of usage statistics (how often a certain problem question is created for example), so they wouldn't need access to selecting source code and creating questions.

2.4 Operating Environment
In terms of operating system, the system shall work on the latest versions of Windows, Linux, and Mac OS. Since the program is not large, any working computer running on the mentioned operating systems should be able to launch and use this system. The computer should have enough storage to save the output files.

2.5 Design and Implementation Constraints
The only security constraint at the moment is that the program should NOT communicate with LMS because of security. It shouldn't upload the output file directly to Canvas for example. The program should only give the output file to the user and the user then decides what to do with the file. In terms of memory, the program will have a limit on the possible number of variations so the program can run smoothly and not crash.

2.6 User Documentation
As of now there is no user documentation to be delivered.

2.7 Assumptions and Dependencies
An assumption related to the program is that we expect the user to input working code from the start. The program is not expected to fix broken code. Also as stated in section (2.5), the program is not expected to give all possible variations if the initial code is very large. There would be a limit on the number of variations to ensure that the program runs smoothly.

3. External Interface Requirements

This section of the document explores the software design, user interface, hardware interface and communication interface of the product. The specifications mentioned in this part shows the guideline of the project's software requirements, implementation and testing.

3.1 User Interfaces

The Parson's Problems Appliance for Learning Management Systems(PPALMS) has been specifically designed to fulfill all the requirements of the users. The main focus of this project is to make the system convenient for users so that they can organize, rearrange, and modify their source code into different new problems for students. The interaction of users to this software is mainly focused on allowing the user to select a piece of source code from their machine, enabling a variety of mechanisms to generate as many variations as possible of the selected type from the provided source and collecting the generated questions for import into LMSs.

The home screen of the system consists of many functions that allows the user to select a source code from their machine so that they can make variations of the selected source code. Every screen will show the back button so that the user can return to the previous page or homepage if they want to modify anything. The source code file should be .txt/.doc files. An error message is shown if the user uploads any other file. The second part of the screen allows the user to annotate any specific lines of the code to include or exclude. In the homescreen, there is an option for the users to select question types where they can generate all possible variations of the source code according to the question type. There is an option to select the intended LMS target. There is a help option too for new users.The help option is marked with a question mark and is shown in every screen of the system.

3.2 Hardware Interface

The Parson's Problems Appliance for Learning Management Systems(PPALMS) has been intended to be a mobile/web system for the Linux,windows or mac platform and which will be supported on any internet based devices. This learning management system (LMS) is a software application or web-based technology used to plan, implement and assess a specific learning process.The PPALMS has been intended to support events from local applications and the main server.

3.3 Software Interfaces

PPALMS is developed under the web operating system. The interaction between this system with other LMS(such as blackboard, canvas or etc) is dependent on the user's course number and the user's institution ID number . The user has to link the exact course to PPALMS so that it is accessible to the students.

3.4 Communications Interfaces

The system will have a network server that is web based and will be using html language. The HTTP server will use a push protocol to push notifications of updates onto the LMS the user will be using. Whenever the user opens the PPALMS software in their browser, a pull protocol will be used to retrieve and sync the latest events in their LMS.

4. System Features

4.1 Code rearranger

This is one of the important features of the system. The system must have this functionality to establish the user's goal.

4.1.1 Description and Priority

The code rearranger of the system gives the user the option to select a specific part of the code. Specifically, it gives the user an option to include or exclude specific parts of the code so that they are able to ask questions. This feature has high priority because the main goal of the user is to permute the specific code part into different types of questions. The feature has high priority and low risk because the system will use every possible way to permute the source code so that it meets the expectation of the user.

4.1.2 Stimulus/Response Sequences

Stimulus: User requests to upload a source code file from his local device

Response: The system uploads the file for further applications

Stimulus: The user requests to annotate specific lines for inclusion or exclusion

Response: The system includes the specific lines as instructed by the user

4.1.3 Functional Requirements

Requirement #: 1 **Requirements Type**: 1 **Use Case**: 1

Date: 10/17/22

Introduction: The first action performed at PPALMS is importing the source the code the user wants to include in the question. PPALMS must support this basic functionality.

Rationale: The client stated in the document given that the system should allow the user to input a source file.

Author: AlYaqdhan Al Maawali **Source**: Elicitation session, User requirement doc

Inputs: Source code file

Requirement Description: The user shall input the source code. The functionality mentioned will be working smoothly if and only if:

1. The user uploads the correct type of file.
2. File size is 1 mb or less.

If the source code file is allowed, then the user will proceed. If not, then error is returned and the user is given an option to input a file again.

Outputs: None

Persistent Changes: source code file is in the system.

User Satisfaction: 5 **User Dissatisfaction:** 5

Related Requirements: N/A

Conflicts: None

Support Materials: None **Test Cases:** TBD, assigned to all, due: Design Phase

---

Requirement #: 2 **Requirements Type**: 1 **Use Case**: 2

Date: 10/03/22

Introduction: One of the actions performed at PPALMS is to include or exclude source code so that the user can modify the source code to make questions. PPALMS must support this functionality.

Rationale: The client indicates that the inclusion and exclusion of the source code to make questions is a must.

Author: Fahia Tabassum **Source**: Elicitation session, User requirement doc

Inputs: source code

Requirement Description: The user shall be able to include and exclude source code. The functionalities mentioned above will be working smoothly if and only if:

1. The user uploads the correct type of file (req 1).

2. The user selects a specific part of the source code to include and leaves the rest for exclusion.

If the user annotated at least two lines, then they will proceed. If the user annotated one line or none, then an error message is displayed and the user is asked to annotate lines again.

Outputs: annotated lines are shown

Persistent Changes: Annotated lines are kept and rest is removed

User Satisfaction: 5 **User Dissatisfaction:** 5

Related Requirements: 1

Conflicts: None

Support Materials: None **Test Cases:** TBD, assigned to all, due: Design Phase

---

Requirement #: 3 **Requirements Type**: 1 **Use Case**: 3

Date: 10/18/22

Introduction: The user should be able to form and order line tuples. Users should be able to group sets of lines together, so there are lines that can be rearranged and lines that cannot be rearranged.

Rationale: Some code needs to be kept together and cannot be rearranged. The user mentioned this functionality in the elicitation session.

Author: AlYaqdhan Al Maawali **Source**: Elicitation session, User requirement doc

Inputs: source code, annotated lines

Requirement Description: The functionality mentioned will be working smoothly if and only if:

1. The user uploaded the correct type of file (req 1).
2. The user annotated at least two lines for inclusion (req 2).

If the user didn't tuple lines together, then the user will proceed normally without any line tuples. If the user did line tuples, then lined tuples will stay together and not be rearranged in the process.

Outputs: None

Persistent Changes: Tupled lines are always kept together

User Satisfaction: 5 **User Dissatisfaction:** 5

Related Requirements: 1, 2

Conflicts: None

Support Materials: None **Test Cases:** TBD, assigned to all, due: Design Phase

---

4.2 Making Different Types of Question

This is one of the important features of the system. The system must have this system to establish the user's goal.

4.2.1 Description and Priority

The main purpose of this feature is to make different types of questions based on the user's choice. The questions are made by permuting the source code from the user's source code file. The requirement of this feature to be done is for the user to select a specific part of the code and choose the type of the question he wants. There are multiple options of questions in the system such as MCQ[3] , matching, ordering, completing the code, expected outcome of the code etc. This feature has high priority because the main goal of the user is to make various types of questions for the students.

4.2.2 Stimulus/Response Sequences

Stimulus: The user requests to select question types

Response: The system responds by giving the user different options such as MCQ,matching, ordering, completing the code, expected outcome of the code etc.

Stimulus: The user asks for different possible variations of the source code according to the question they want to make

Response: The system permutes the source code in all possible ways and gives the user various options to select on which one he/she wants to ask a question.

4.2.3 Functional Requirements

Requirement #: 4 **Requirements Type**: 1 **Use Case**: 4

Date: 10/03/22

Introduction: The user should be able to specify what type of question they want to generate. PPALMS has to be able to generate variations based on the type of question picked such as MCQ, matching, ordering, completing the code, expected outcome of the code etc.

Rationale: User stated in the initial requirement document and the elicitation session that the user has the option to pick the type of question they want to generate variations of.

Author: Fahia Tabassum **Source**: Elicitation session, User requirement doc

Inputs: source code, annotated lines, lines tupled (if applicable), type of question

Requirement Description: The functionalities mentioned above will be working smoothly if and only if:

1. An appropriate file is uploaded to the system (req 1).
2. The user included a specific part of the source code or excluded unnecessary code (req 2).
3. The user selected one type of question from the options given by the system.

The user cannot proceed from this step until they have picked a question type.

Outputs: variations of the code that matches the question type.

Persistent Changes: variations generated based on question type picked by the user.

User Satisfaction: 5 **User Dissatisfaction:** 5

Related Requirements: 1, 2

Conflicts: None

Support Materials: None **Test Cases:** TBD, assigned to all, due: Design Phase

---

4.3 Selecting the LMS and Generating the output file

This is one of the important features of the system. The system must have this system to establish the user's goal.

4.3.1 Description and Priority

This feature is very important for the user because the user surely wants to link the system and the output files to his intended LMS. Selecting the user's intended LMS target and creating the expected output file are the main goals of this feature. This feature has high priority because if the user can not connect to the intended LMS and can not get the intended output file then the system fails to establish the requirement of the user.

4.3.2 Stimulus/Response Sequence

Stimulus: The user requests to link the intended LMS target to the system

Response: The system asks for the user's institution's credentials and id so that the system can link the system to the intended LMS

Stimulus: The user requests for an appropriate output file

Response: The system gives an output file with the expected types of questions from the source code

4.3.3 Functional Requirements

Requirement #: 5 **Requirements Type**: 1 **Use Case**: 5

Date: 10/03/22

Introduction: Selecting the LMS and Generating the output file is one of the most important features of the system. Selecting the user's intended LMS target and creating the expected output file are the main goals of this feature. PPALMS must support this functionality.

Rationale: User stated that the system shall allow the user to select the intended LMS target. The final goal of the system is to generate an output file, which is also part of the requirement.

Author: Fahia Tabassum **Source**: Elicitation session, User requirement doc

Inputs: User's choice on the LMS, type of output file

Requirement Description: The user shall be able to choose the LMS target and pick the desired type of output file. The functionalities mentioned above will be working smoothly if and only if:

1. The user selects his/her LMS
2. The user chooses his/her desired type of output file
3. The user is able to sync the LMS and the output file

If the LMS is selected and the output file is requested, PPALMS should have made the output file and should have synched the output file to the LMS.

Outputs: The output file to be synched with the LMS

Persistent Changes: The output file will be generated and it will be synched to the LMS.

User Satisfaction: 5 **User Dissatisfaction:** 5

Related Requirements: N/A

Conflicts: None

Support Materials: None **Test Cases:** TBD, assigned to all, due: Design Phase

---

Requirement #: 6 **Requirements Type**: 1 **Use Case**: 6

Date: 10/18/22

Introduction: The PPALMs system must be able to generate a problem for a user, and be able to reuse it in the future.

Rationale: With the goal of convenience to the user, the user had suggested that they would like to be able to use a problem set more than once.

Author: Ahmed Al Rai**si** **Source**: Elicitation session, User requirement doc

Inputs: None

Requirement Description: The system shall allow the user to reuse problems by:

1. Presenting the Instructor/TA the option to see the last 10 problems generated
2. Allowing the regeneration of a problem that has already been generated into a previous problem set.

Outputs: A problem set that has been previously generated.

Persistent Changes: The new problem generated will appear as the most recent problem generated within the last 10 problems.

User Satisfaction: 5 **User Dissatisfaction:** 5

Related Requirements: N/A

Conflicts: None

Support Materials: None **Test Cases:** TBD, assigned to all, due: Design Phase

---

Requirement #: 7 **Requirements Type**: 1 **Use Case**: 7

Date: 10/18/22

Introduction: The PPALMs system must be able to generate statistics for a given question type.

Rationale: When the user selects the statistics from a problem type, the statistics regarding that question type will be shown.

Author: Ahmed Al Raisi **Source**: Elicitation session, User requirement doc

Inputs: User selects question type.

Requirement Description: The system shall allow the user to get the problem type statistics if and only if:

1. The user is logged in to a valid and verified account, with their school credentials linked to it.
2. The user selects a question type.

Outputs: Statistics regarding question type.

Persistent Changes: The statistics will be shown to the user.

User Satisfaction: 5 **User Dissatisfaction:** 5

Related Requirements: N/A

Conflicts: None

Support Materials: None **Test Cases:** TBD, assigned to all, due: Design Phase

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The system shall support each user utilizing the desktop application on the latest versions of MacOS, Windows, or Linux. The system's response time shall depend on the internet connection of the user. The desktop computer and the memory size shall also depend on the user.

5.2 Safety Requirements

The source code for the system will be revised regularly by the research and development team. The system will include a bug and error reporter so the research and development team can keep track of bugs and safeguard the system from potential data loss.

5.3 Security Requirements

The System requires that only instructors and teacher assistants are able to assign problems and view solutions. There is no distinction between the security privileges of instructors and teacher assistants in this system.

5.4 Software Quality Attributes

- **Availability:** The system shall be available for question selection, formation, annotation, and importation into a LMS at all times.
- **Adaptability:** The system shall accommodate users with user-friendly software features that do not require additional training for using the system.
- **Reusability:** The system shall sustain the potential re-use of problems.
- **Usability:** The system shall be a desktop application that is able to download the output files to the computer.
- **Correctness:** The system shall contain the correct number of selected questions, types of questions, and any annotations made by the user.
- **Interoperability:** The system shall save the changes made by the user. Changes made by other users will not be saved across systems. Users can only change the information on their desktop.

5.5 Business Rules

- Users: Instructors and teacher assistants have the same roles and responsibilities in the system. These users have access to all of the features within the system.
- Research and development team: This team has access to the system's features as well as the code base for the same. Only the research and development can make system changes such as bug improvements and updates to the system's interface.
- Admin: The system has no designated system administrator. Default user is all that is necessary.

5.6 Internalization Issues

The system requirements are abstract enough that it allows the system to be implemented in any language to be used within the system, this eliminates any internalization issues at this stage.

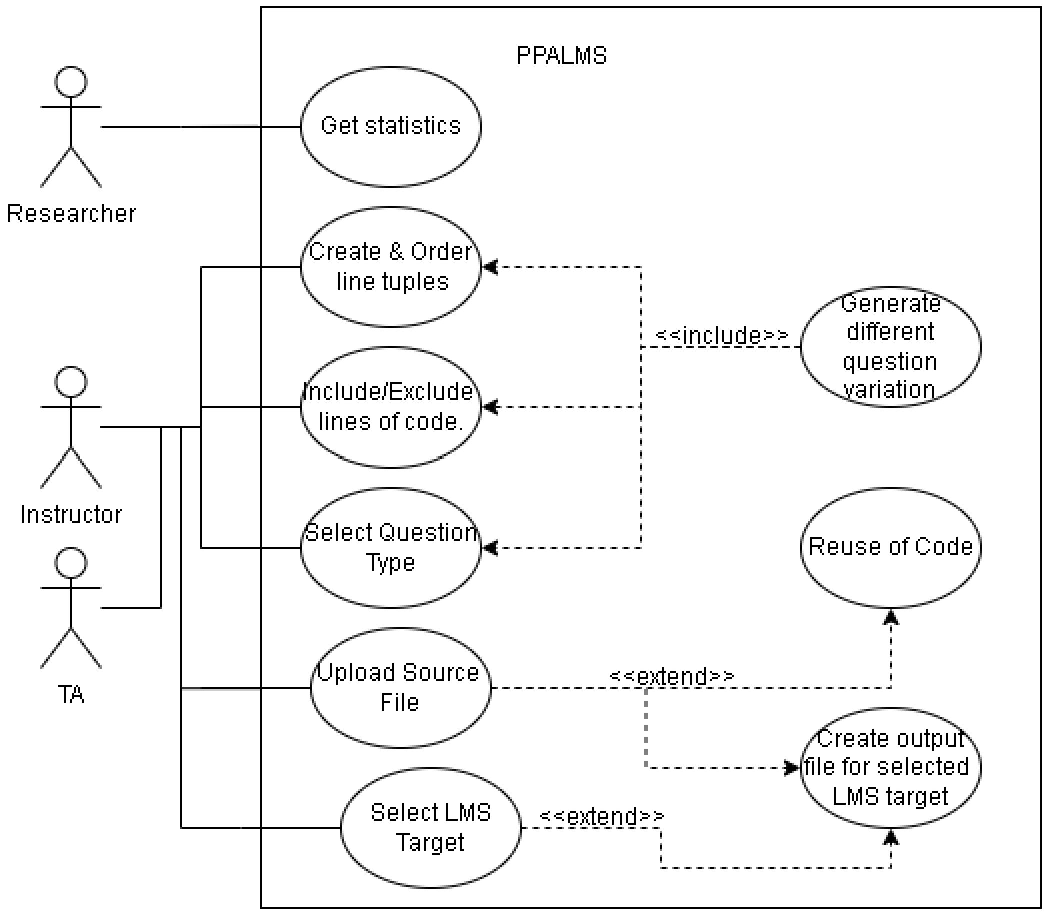
Appendix A: Glossary

1. **PPALMS**: Parson's Problems Appliance for Learning Management Systems
2. **LMS**: Learning Management System
3. **MCQ:** Multiple choice question

Appendix B: Models

1. Use Case Diagram

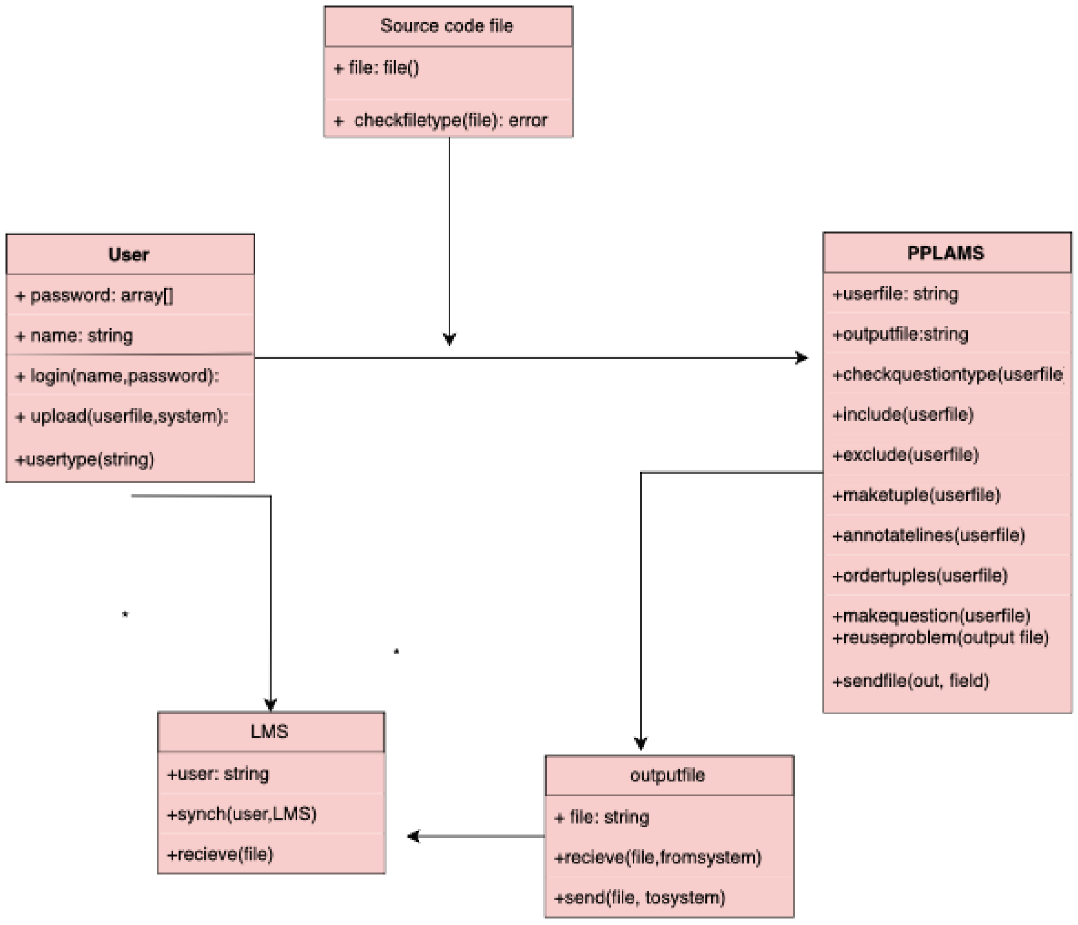
The use case diagram below demonstrates how the Instructor or the Teaching assistant will use the PPALMS system to accomplish the goal of creating different code question types efficiently. We can see the relationships between the use cases are mainly defined by either an <<include>> or <<extend>> tag.



2. System Model

2.1 Class information

2.1.1 Class Diagram



2.1.2

Class Descriptions: Classes in the PPALMS

1. Class: User

- Purpose: the user class uploads the source code file in the system. The user has to login using his username and password and he has to synchronize his school LMS to the system
- Constraints: None
- Persistent: No(created at system initialization from other available data)
- Attribute description:
    - Password: an array of strings/integers
    - Name: string
    - Description: will ask for the username and password to login to the system
- Method Description:
    - login(name,password): will give the user an access to the system
    - upload(userfile,system): will let the user upload his file to PPALMS
    - usertype(string): will check the user type among researcher, TA, instructor

2. Class: PPALMS
    - Purpose: To modify the source code and make an output file
    - Constraints: None
    - Persistent:No (created at system initialization from other available data)
    - Attribute Description:
        - Userfile(string): the
        - outputfile(string):
    - Method Description:
        - check_question_type(userfile): checks the type of the question user wants to make
        - include(userfile): includes the code segment from the userfile
        - exclude(userfile): excludes the code segments from the userfile
        - make_tuple(userfile): makes tuple in the code segments from the userfile
        - annotate_lines(userfile): annotates code segments from the userfile
        - order_tuples(userfile): orders the segments
        - make_question(userfile): makes the question from the code segments
        - reuse_problem(outputfile):reuses the problems
        - sendfile(output,LMS):sends file to the LMS

`3. Class:SourceCodeFile

- Purpose:To select the correct type of source code file
- Constraints: None
- Persistent: No(created at system initialization from other available data)
- Attribute description:
    - user: string
- Method Description:
    - synch(user,LMS): will synch user's LMS to the system
    - receive(file): receives the file from the system

4. Class: outputfile

- Purpose: To receive file from PPALMS and to send that file to LMS
- Constraints: None
- Persistent: No(created at system initialization from other available data)

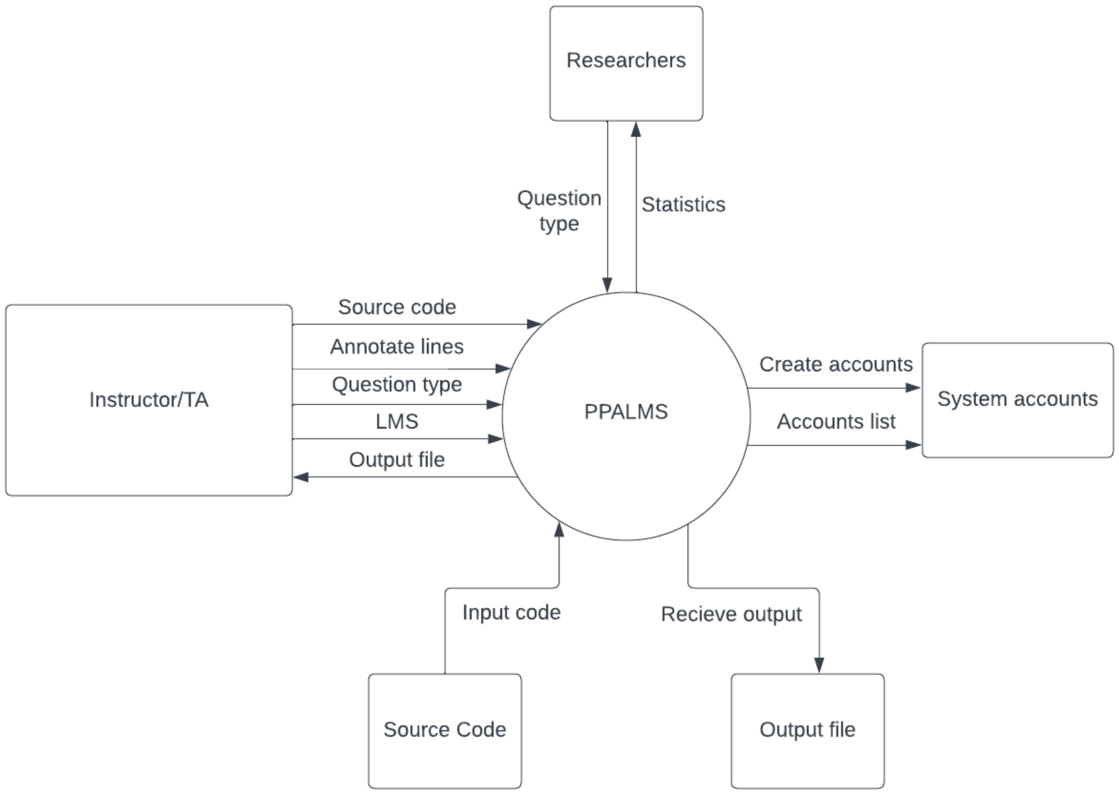
- Attribute description:
    - File: string
- Method Description:
    - receive(file,PPALMS): receive the file from the system
    - send(file, tosystem): send that file to the LMS

5.Class: Source Code File

- Purpose: provide the user with specific type of file
- Constraints: None
- Persistent: No(created at system initialization from other available data)
- Attribute description:
    - File: choose the specific file type
- Method Description:
    - checkfiletype(file): check the type of the file. If it's not .txt/.pdf then will print error message

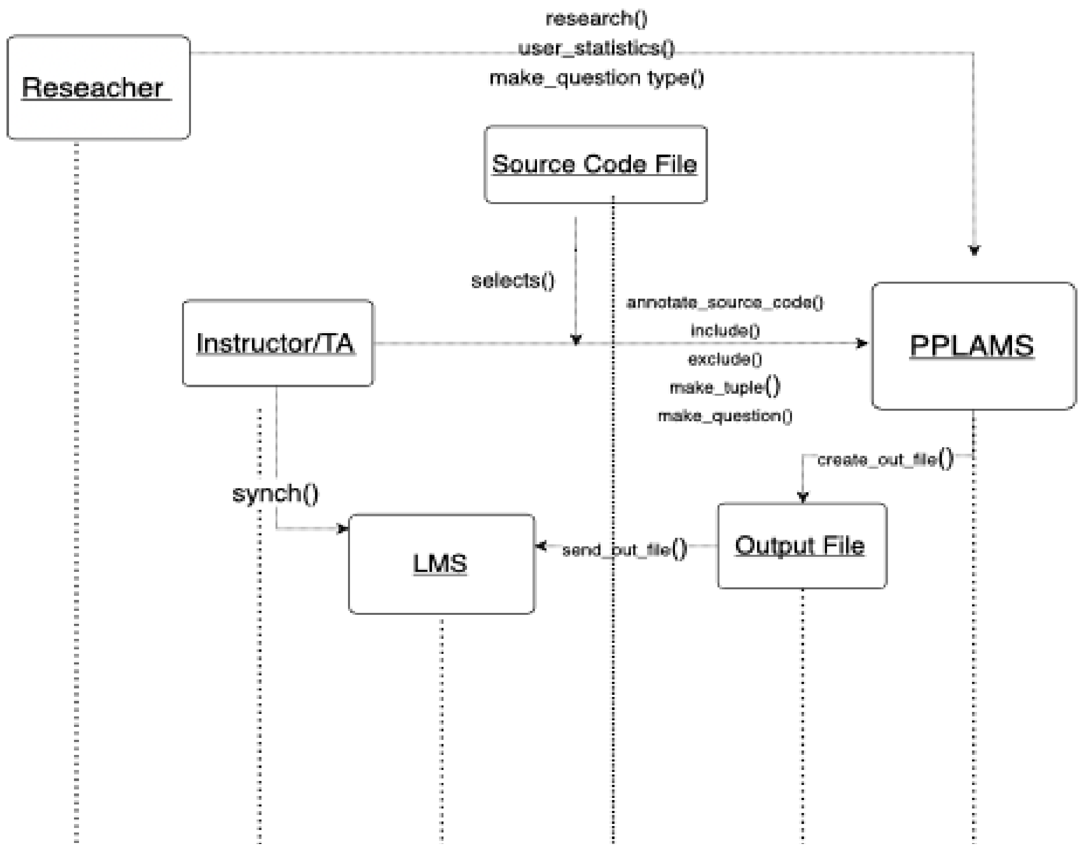
2.2 Context Model

The context model shows the flow of information and data between the system and the different entities. As can be seen in the diagram, PPALMS communicates between the users, system accounts database, input code, and output file. The instructors input the source code, annotate lines, select question types, select LMS target, and in return get the output file. Researchers select the question type and get statistics based on their selection. The system also allows for creation of accounts and viewing account lists by communicating with the system accounts database. Lastly, data flows to the system from source code and from the system to the output file.

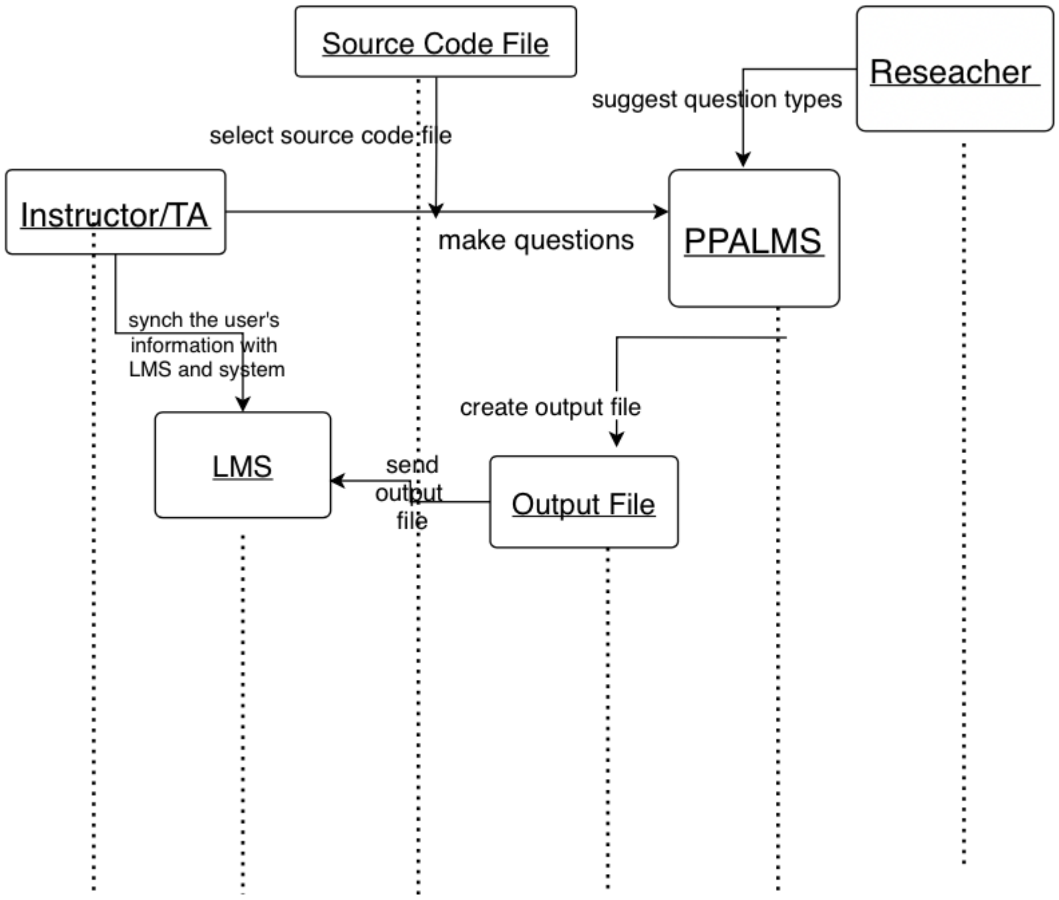


2.3 Interactions Model

The sequence interaction diagrams below illustrate the sequence of messages between the objects of PPALMS. The instructors/TA use the source code file in the PPALMS system so that they can annotate, include or exclude the source code. The researchers pretty much researches and uses user statistics to make the types of questions in the system. In the PPALMS, the questions are made based on the different choices of the user. When the system makes the output file then it sends the file to the LMS which is already synched with the Instructor/TA's information and school credentials.



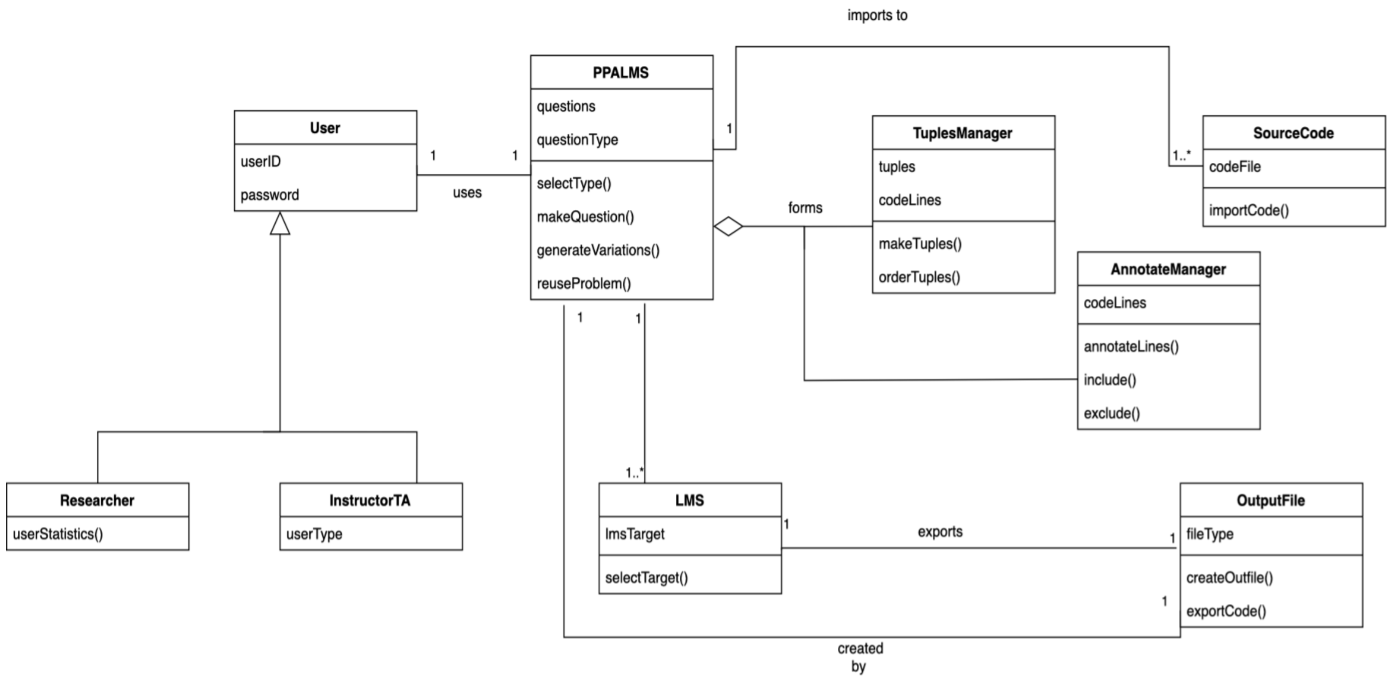
Sequence Diagram 1



Sequence Diagram 2

2.4 Structure Model

The structure model below illustrates a diagram with the relationships of the classes in the PPALMS system. The user class has a generalization relationship with the Researcher and InstructorTA classes because the subclasses (Researcher and InstructorTA) are an extension of the superclass User. Both of the subclasses have attributes from User but also have specific attributes and operations. A researcher in this system is only concerned with the usage statistics and the Instructor or TA create and select questions. This diagram models that. Furthermore, The classes TuplesManager and AnnotateManager have an aggregation relationship with PPALMS. The reason for this is because these two classes are parts of the larger PPALMS class. They included operations that are part of a whole objective. For example, these classes are responsible for forming tuples and annotating lines. Also, there are a number of other association relationships. The class SourceCode has an association with PPALMS since the source code is imported into the PPALMS system. Many source code files can be imported into the system. OutputFile has one with LMS since it exports into any LMS target capable in this system. There is 1 file exported to the selected target. OutputFile also has an association relationship with PPALMS since the class creates an output file for the PPALMS system. 1 file is created with each use. The classes PPALMS and LMS also have an association relationship. The system is able to select from a number of LMS targets. Lastly, 1 user uses the PPALMS system at a time (with each desktop application), so there is also an association relationship there. This diagram also demonstrates names of attributes in each respective class as well as the operations or functions in those classes.


imports to
PPALMS
questions
questionType
selectType()
makeQuestion()
generateVariations()
reuseProblem()
User
userID
password
uses
1
1
TuplesManager
tuples
codeLines
makeTuples()
orderTuples()
SourceCode
codeFile
importCode()
1..*
forms
AnnotateManager
codeLines
annotateLines()
include()
exclude()
Researcher
userStatistics()
InstructorTA
userType
LMS
lmsTarget
selectTarget()
1..*
exports
OutputFile
fileType
createOutfile()
exportCode()
created by