Learning Management System for Programming Languages

Design Document

CSCE 247 Software Engineering

William Edwards
Casey Fainberg
Noah Parker
Austin Sanza

February 26th, 2023
Version 1.0

Table of Contents

System Overview

Our program is a Learning Management System which handles each user, course, and all related information hosted by the system, all of which are stored using JSON files. The data will be used and displayed to the user via the user interface, and the LMS class acts as an inbetween for the data and user interface. It will allow functionality such as creating and logging into accounts, creating and retrieving courses, saving the grade achieved on each course, and posting and retrieving comments and replies.

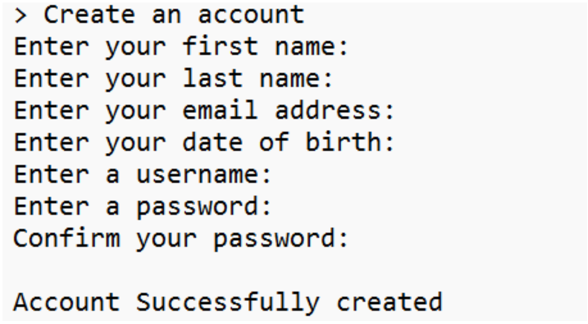
Environment Overview

This program is completely coded in Java on VSCode. Therefore, it runs on the Java console. All input will be handled through the Java console by a Scanner object in our program. The system will run on a single machine with all necessary data being saved in JSON files on the same machine.

User Interface

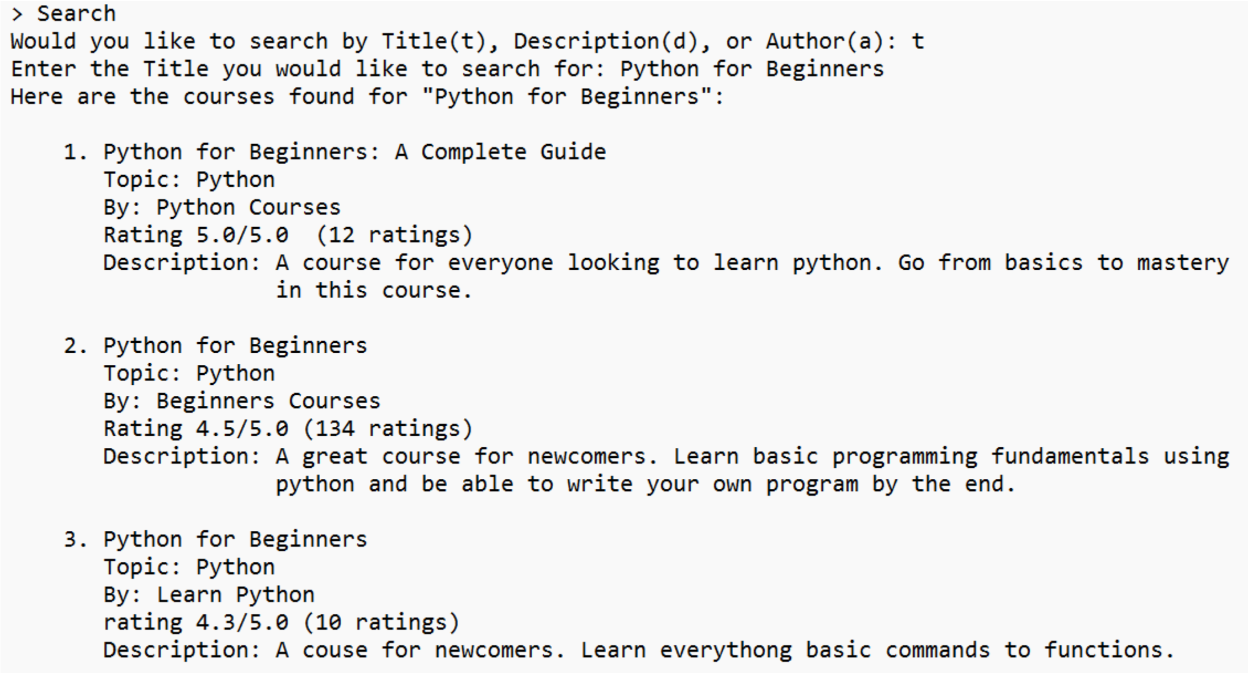
The user interface of the system will allow the user to communicate with the management system. It will be entirely text based and localized to the Java console. The > character is followed by one or more prompts for user input.

Creation of an account:

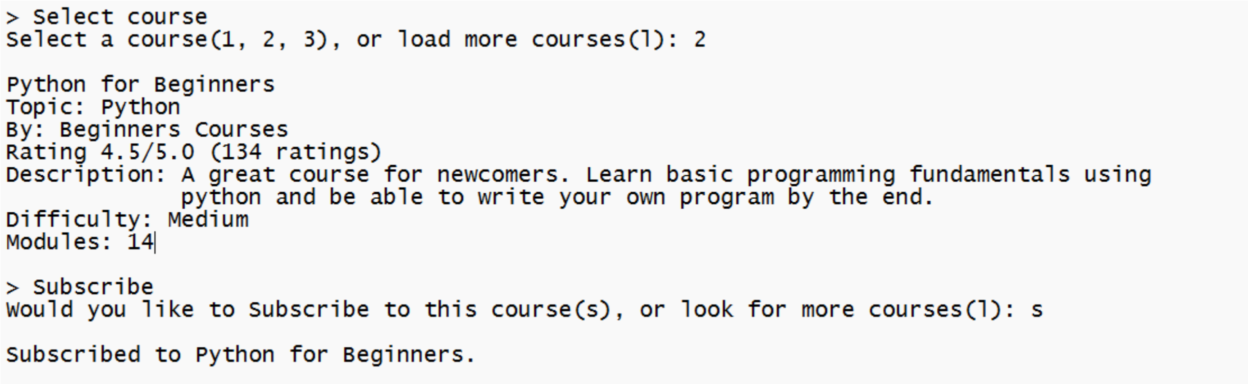


```
> Create an account
Enter your first name:
Enter your last name:
Enter your email address:
Enter your date of birth:
Enter a username:
Enter a password:
Confirm your password:

Account Successfully created
```

Searching for courses:



```
> Search
Would you like to search by Title(t), Description(d), or Author(a): t
Enter the Title you would like to search for: Python for Beginners
Here are the courses found for "Python for Beginners":

    1. Python for Beginners: A Complete Guide
       Topic: Python
       By: Python Courses
       Rating 5.0/5.0  (12 ratings)
       Description: A course for everyone looking to learn python. Go from basics to mastery
                    in this course.

    2. Python for Beginners
       Topic: Python
       By: Beginners Courses
       Rating 4.5/5.0 (134 ratings)
       Description: A great course for newcomers. Learn basic programming fundamentals using
                    python and be able to write your own program by the end.

    3. Python for Beginners
       Topic: Python
       By: Learn Python
       rating 4.3/5.0 (10 ratings)
       Description: A couse for newcomers. Learn everythong basic commands to functions.
```



```
> Select course
Select a course(1, 2, 3), or load more courses(l): 2

Python for Beginners
Topic: Python
By: Beginners Courses
Rating 4.5/5.0 (134 ratings)
Description: A great course for newcomers. Learn basic programming fundamentals using
             python and be able to write your own program by the end.
Difficulty: Medium
Modules: 14

> Subscribe
Would you like to Subscribe to this course(s), or look for more courses(l): s

Subscribed to Python for Beginners.
```

Creating a course:

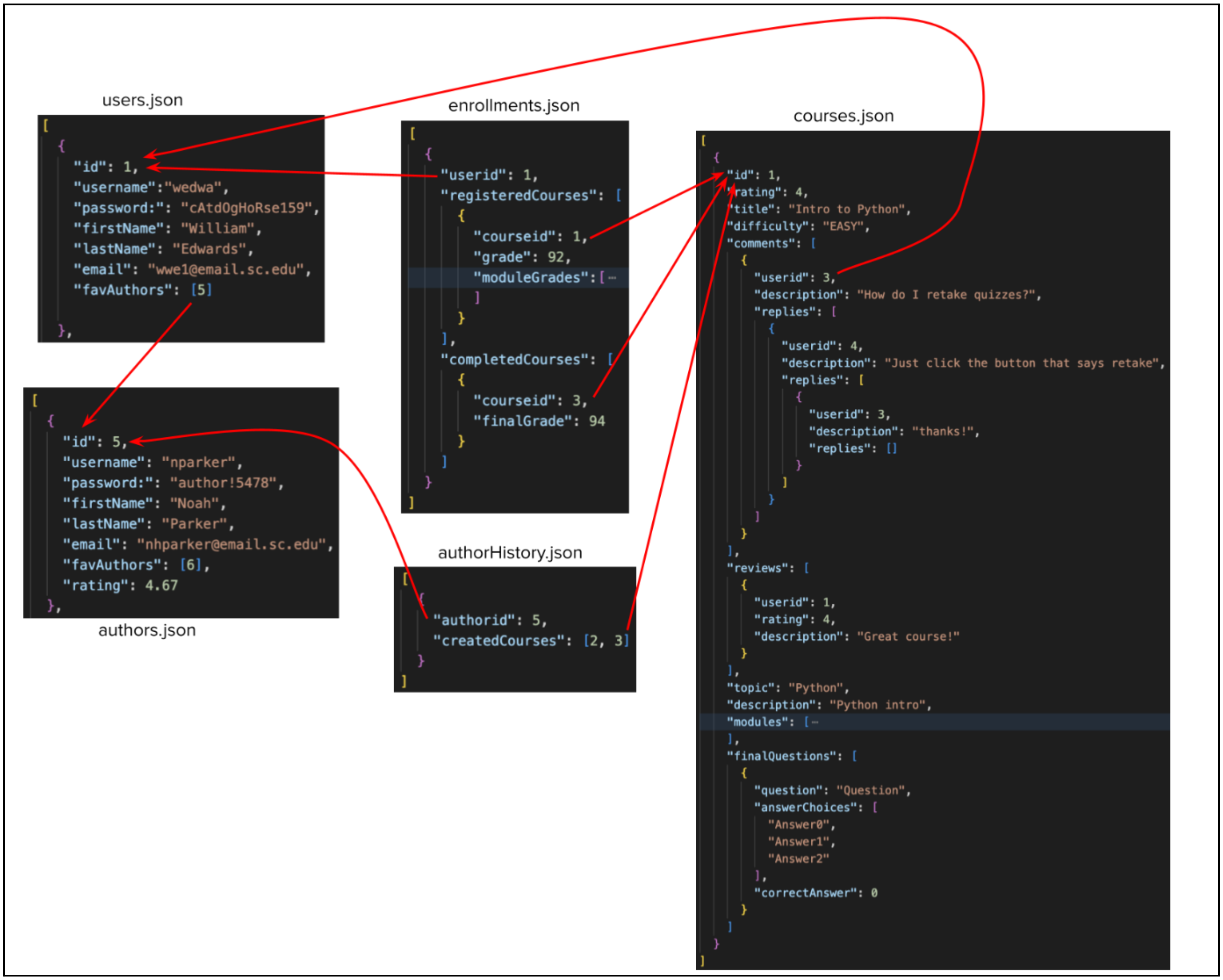
```
> Course Info
Enter the name of the course: Introduction to courses
Enter the Course Description (Feel free to use multiple lines, enter a blank line to end):
Introuduces the user to contents contained within courses of LMS
Enter the Course Difficulty (Easy, Medium, Hard): Medium

Module 1
> Type
Would you like to add a lesson(l) or assessment(a): l

Lesson 1
> Lesson info
Enter all text for the module (enter a line with only the contents (finished) to end):
Welcome to this introductory course about courses!

Each course will contain a collection of lessons teach you the material present.
At the end of each module, there will be a short quiz to test you on what you learned.
All assessments will be multiple choice, just type (1, 2, 3, 4) to answer.

finished
```

```
> Adding questions
Now you will create a quiz on the lesson, this quiz will only contain up to
Enter the question: Does every module have a quiz?
Enter the number of choices (up to 4): 2
Enter answer choice 1: Yes
Enter answer choice 2: No
Enter correct answer choice: 1

Would you like to add another question (max of 10 questions) (Y/n): n

Module 1 is complete

> Add more sections
Would you like to add another section (Y/n): n
```

```
> Creating the final exam
You will now create a final exam that can have up to 20 questions:
Enter the question: How many answer choices does this question have?
Enter the number of choices (up to 4): 4
Enter answer choice 1: 1
Enter answer choice 2: 2
Enter answer choice 3: 3
Enter answer choice 4: 4
Enter correct answer choice: 4

Would you like to add another question (max of 20 questions) (Y/n): n

Course Complete!
```

Data Storage

The data of our system will be stored in JSON files which will be kept on the local machine. We will have five JSON files: courses.json, users.json, authors.json, enrollments.json, and authorHistory.json. Each file contains a list of relevant objects, with each object having a unique ID. Below, we illustrate which files reference the IDs of objects in other files. The users.json file refers to authors to save a list of favorite authors. The enrollments.json file refers to users.json and courses.json to explain which courses the users are/were enrolled in. The authorHistory.json file refers to author.json and courses.json to explain which courses the authors created. Finally, courses.json refers to users.json to describe which user created comments and reviews. Because of these dependencies, we will first load authors.json, then users.json, then courses.json, then enrollments.json and authorHistory.json.



users.json

```
[
  {
    "id": 1,
    "username":"wedwa",
    "password:": "cAtdOgHoRse159",
    "firstName": "William",
    "lastName": "Edwards",
    "email": "wwe1@email.sc.edu",
    "favAuthors": [5]

  },
```

authors.json

```
[
  {
    "id": 5,
    "username": "nparker",
    "password:": "author!5478",
    "firstName": "Noah",
    "lastName": "Parker",
    "email": "nhparker@email.sc.edu",
    "favAuthors": [6],
    "rating": 4.67
  },
```

enrollments.json

```
[
  {
    "userid": 1,
    "registeredCourses": [
      {
        "courseid": 1,
        "grade": 92,
        "moduleGrades":[⋯
        ]
      }
    ],
    "completedCourses": [
      {
        "courseid": 3,
        "finalGrade": 94
      }
    ]
  }
]
```

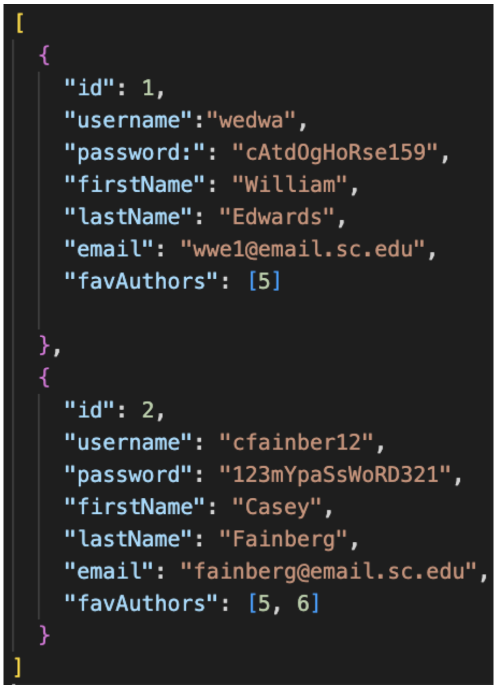
authorHistory.json

```
[
  {
    "authorid": 5,
    "createdCourses": [2, 3]
  }
]
```

courses.json

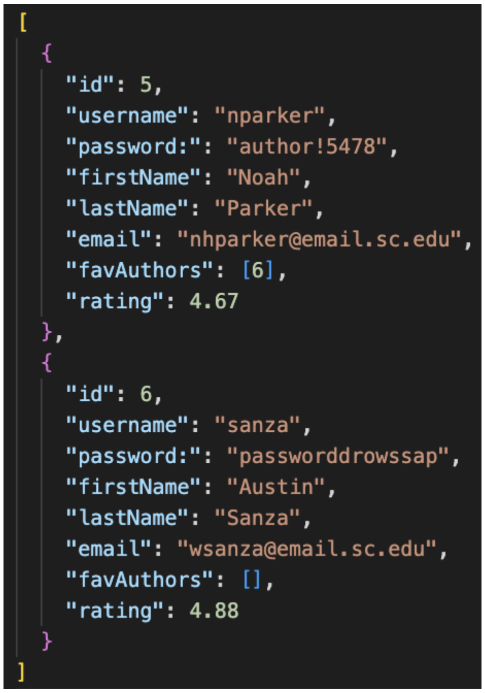
```
[
  {
    "id": 1,
    "rating": 4,
    "title": "Intro to Python",
    "difficulty": "EASY",
    "comments": [
      {
        "userid": 3,
        "description": "How do I retake quizzes?",
        "replies": [
          {
            "userid": 4,
            "description": "Just click the button that says retake",
            "replies": [
              {
                "userid": 3,
                "description": "thanks!",
                "replies": []
              }
            ]
          }
        ]
      }
    ],
    "reviews": [
      {
        "userid": 1,
        "rating": 4,
        "description": "Great course!"
      }
    ],
    "topic": "Python",
    "description": "Python intro",
    "modules": [⋯
    ],
    "finalQuestions": [
      {
        "question": "Question",
        "answerChoices": [
          "Answer0",
          "Answer1",
          "Answer2"
        ],
        "correctAnswer": 0
      }
    ]
  }
]
```

users.json



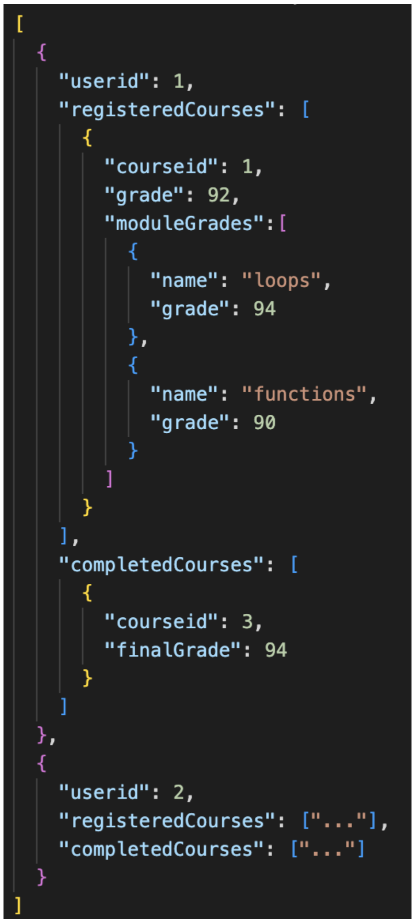
```
[
  {
    "id": 1,
    "username":"wedwa",
    "password:": "cAtdOgHoRse159",
    "firstName": "William",
    "lastName": "Edwards",
    "email": "wwe1@email.sc.edu",
    "favAuthors": [5]

  },
  {
    "id": 2,
    "username": "cfainber12",
    "password": "123mYpaSsWoRD321",
    "firstName": "Casey",
    "lastName": "Fainberg",
    "email": "fainberg@email.sc.edu",
    "favAuthors": [5, 6]
  }
]
```

authors.json



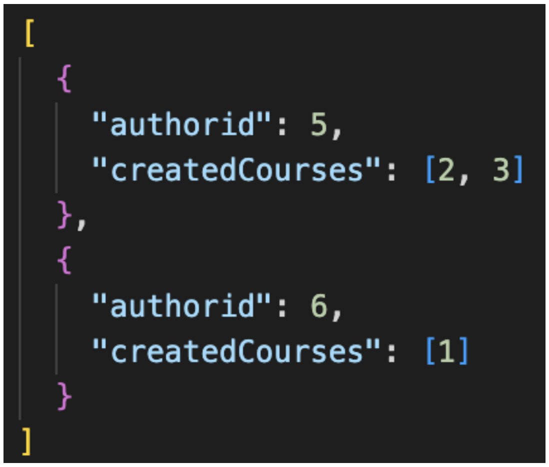
```
[
  {
    "id": 5,
    "username": "nparker",
    "password:": "author!5478",
    "firstName": "Noah",
    "lastName": "Parker",
    "email": "nhparker@email.sc.edu",
    "favAuthors": [6],
    "rating": 4.67
  },
  {
    "id": 6,
    "username": "sanza",
    "password:": "passworddrowssap",
    "firstName": "Austin",
    "lastName": "Sanza",
    "email": "wsanza@email.sc.edu",
    "favAuthors": [],
    "rating": 4.88
  }
]
```

enrollments.json



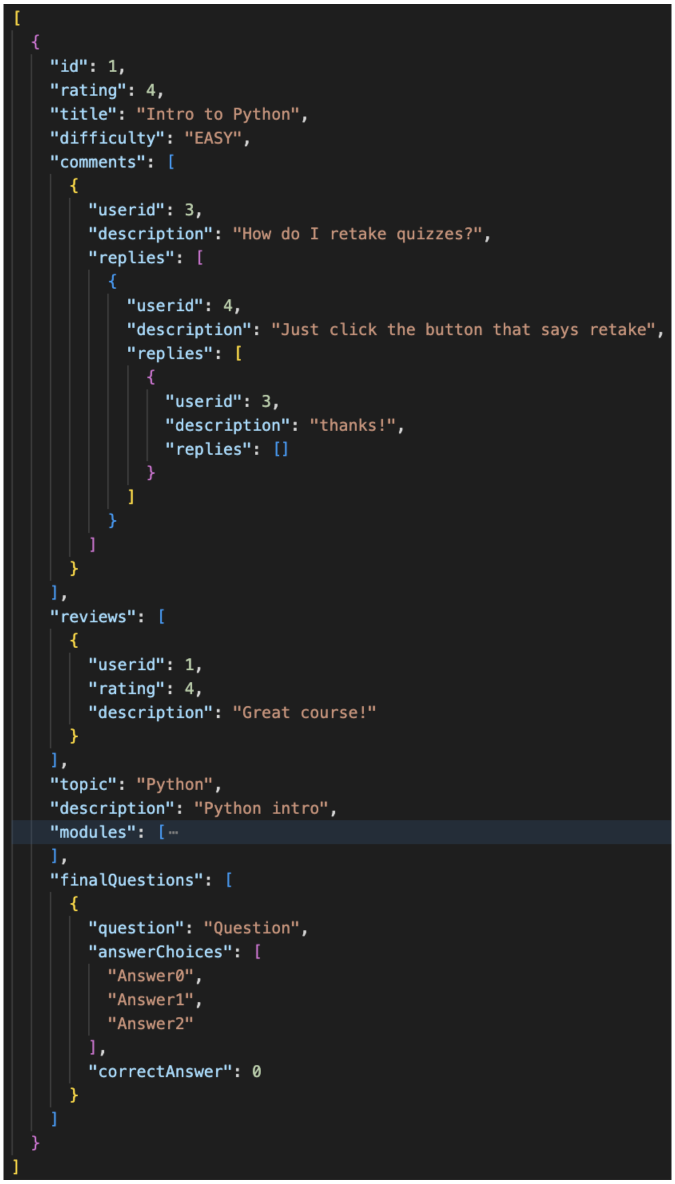
```
[
  {
    "userid": 1,
    "registeredCourses": [
      {
        "courseid": 1,
        "grade": 92,
        "moduleGrades":[
          {
            "name": "loops",
            "grade": 94
          },
          {
            "name": "functions",
            "grade": 90
          }
        ]
      }
    ],
    "completedCourses": [
      {
        "courseid": 3,
        "finalGrade": 94
      }
    ]
  },
  {
    "userid": 2,
    "registeredCourses": ["..."],
    "completedCourses": ["..."]
  }
]
```

authorHistory.json



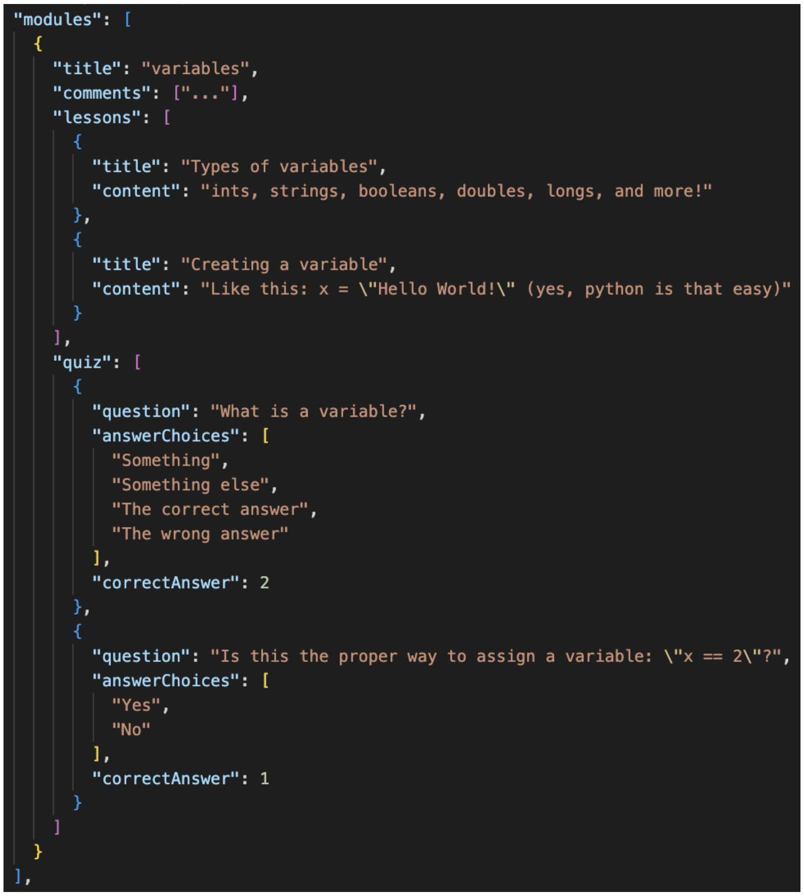
```
[
  {
    "authorid": 5,
    "createdCourses": [2, 3]
  },
  {
    "authorid": 6,
    "createdCourses": [1]
  }
]
```

Courses.json



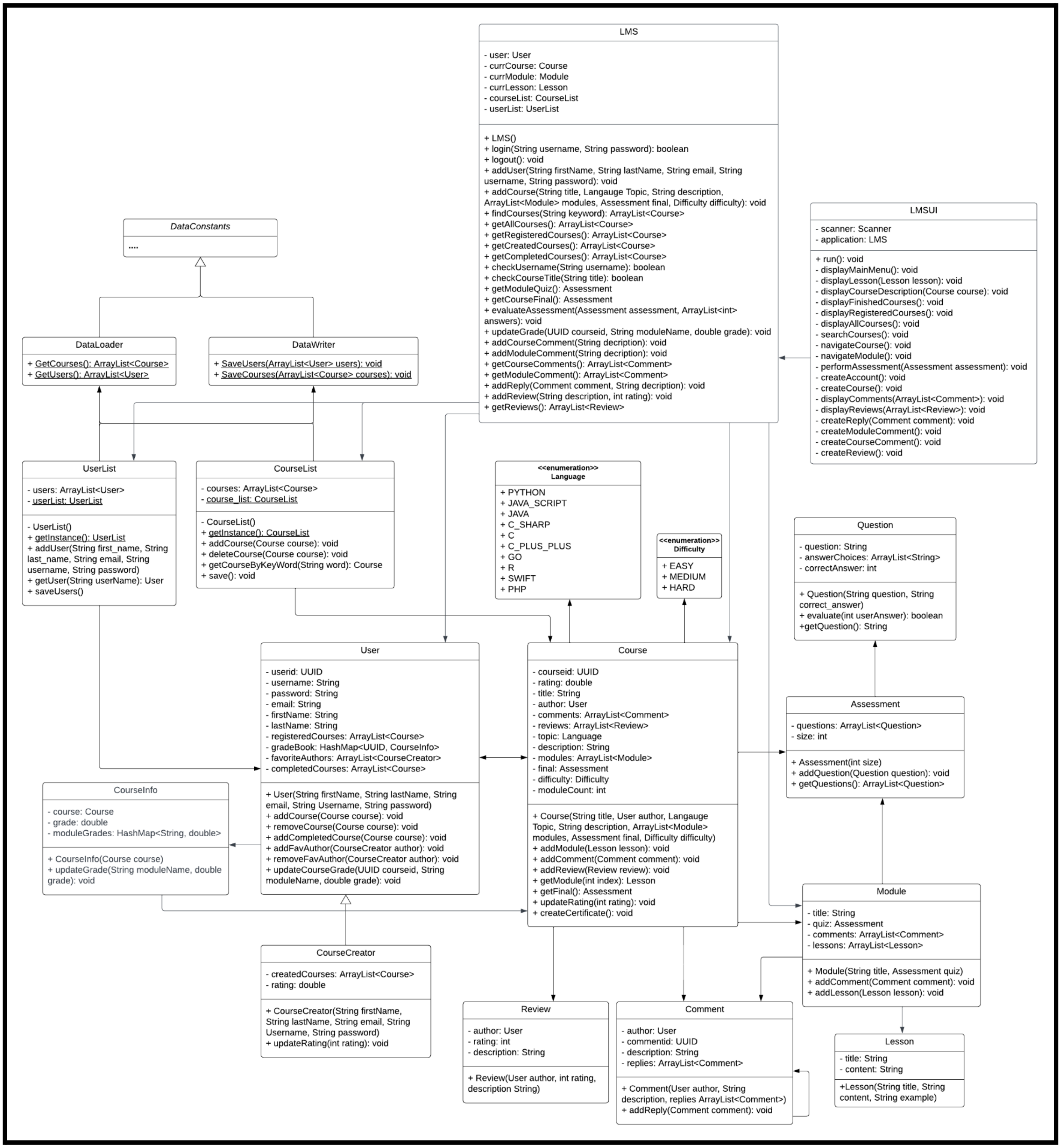
```
[
  {
    "id": 1,
    "rating": 4,
    "title": "Intro to Python",
    "difficulty": "EASY",
    "comments": [
      {
        "userid": 3,
        "description": "How do I retake quizzes?",
        "replies": [
          {
            "userid": 4,
            "description": "Just click the button that says retake",
            "replies": [
              {
                "userid": 3,
                "description": "thanks!",
                "replies": []
              }
            ]
          }
        ]
      }
    ],
    "reviews": [
      {
        "userid": 1,
        "rating": 4,
        "description": "Great course!"
      }
    ],
    "topic": "Python",
    "description": "Python intro",
    "modules": [⋯
    ],
    "finalQuestions": [
      {
        "question": "Question",
        "answerChoices": [
          "Answer0",
          "Answer1",
          "Answer2"
        ],
        "correctAnswer": 0
      }
    ]
  }
]
```

(module info within course object)

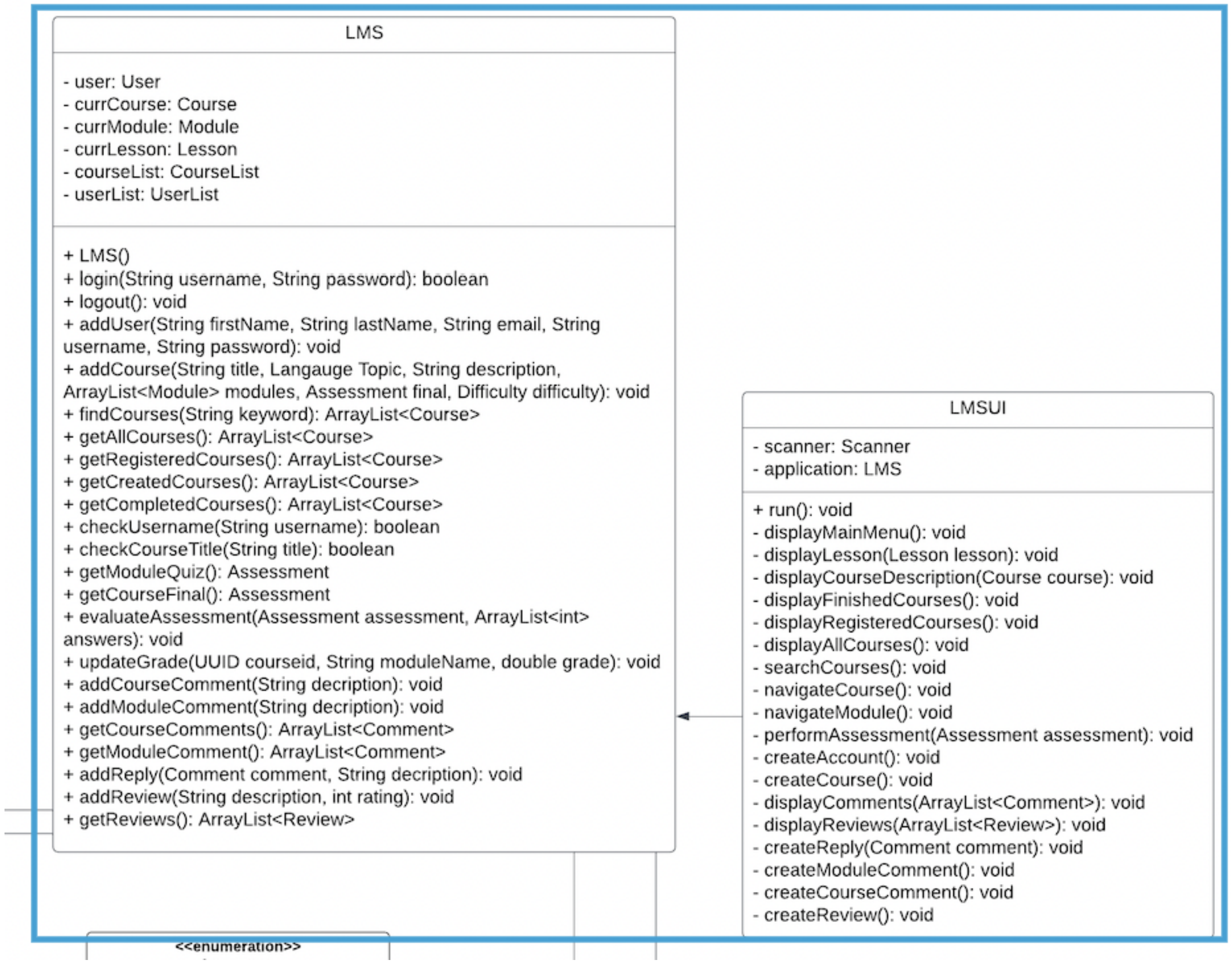


```
"modules": [
  {
    "title": "variables",
    "comments": ["..."],
    "lessons": [
      {
        "title": "Types of variables",
        "content": "ints, strings, booleans, doubles, longs, and more!"
      },
      {
        "title": "Creating a variable",
        "content": "Like this: x = \"Hello World!\" (yes, python is that easy)"
      }
    ],
    "quiz": [
      {
        "question": "What is a variable?",
        "answerChoices": [
          "Something",
          "Something else",
          "The correct answer",
          "The wrong answer"
        ],
        "correctAnswer": 2
      },
      {
        "question": "Is this the proper way to assign a variable: \"x == 2\"?",
        "answerChoices": [
          "Yes",
          "No"
        ],
        "correctAnswer": 1
      }
    ]
  }
],
```

Class Diagram


LMS
- user: User
- currCourse: Course
- currModule: Module
- currLesson: Lesson
- courseList: CourseList
- userList: UserList
+ LMS()
+ login(String username, String password): boolean
+ logout(): void
+ addUser(String firstName, String lastName, String email, String username, String password): void
+ addCourse(String title, Langauge Topic, String description, ArrayList<Module> modules, Assessment final, Difficulty difficulty): void
+ findCourses(String keyword): ArrayList<Course>
+ getAllCourses(): ArrayList<Course>
+ getRegisteredCourses(): ArrayList<Course>
+ getCreatedCourses(): ArrayList<Course>
+ getCompletedCourses(): ArrayList<Course>
+ checkUsername(String username): boolean
+ checkCourseTitle(String title): boolean
+ getModuleQuiz(): Assessment
+ getCourseFinal(): Assessment
+ evaluateAssessment(Assessment assessment, ArrayList<int> answers): void
+ updateGrade(UUID courseid, String moduleName, double grade): void
+ addCourseComment(String decription): void
+ addModuleComment(String decription): void
+ getCourseComments(): ArrayList<Comment>
+ getModuleComment(): ArrayList<Comment>
+ addReply(Comment comment, String decription): void
+ addReview(String description, int rating): void
+ getReviews(): ArrayList<Review>
LMSUI
- scanner: Scanner
- application: LMS
+ run(): void
- displayMainMenu(): void
- displayLesson(Lesson lesson): void
- displayCourseDescription(Course course): void
- displayFinishedCourses(): void
- displayRegisteredCourses(): void
- displayAllCourses(): void
- searchCourses(): void
- navigateCourse(): void
- navigateModule(): void
- performAssessment(Assessment assessment): void
- createAccount(): void
- createCourse(): void
- displayComments(ArrayList<Comment>): void
- displayReviews(ArrayList<Review>): void
- createReply(Comment comment): void
- createModuleComment(): void
- createCourseComment(): void
- createReview(): void
DataConstants
....
DataLoader
+ GetCourses(): ArrayList<Course>
+ GetUsers(): ArrayList<User>
DataWriter
+ SaveUsers(ArrayList<User> users): void
+ SaveCourses(ArrayList<Course> courses): void
UserList
- users: ArrayList<User>
- userList: UserList
- UserList()
+ getInstance(): UserList
+ addUser(String first_name, String last_name, String email, String username, String password)
+ getUser(String userName): User
+ saveUsers()
CourseList
- courses: ArrayList<Course>
- course_list: CourseList
- CourseList()
+ getInstance(): CourseList
+ addCourse(Course course): void
+ deleteCourse(Course course): void
+ getCourseByKeyWord(String word): Course
+ save(): void
<<enumeration>>
Language
+ PYTHON
+ JAVA_SCRIPT
+ JAVA
+ C_SHARP
+ C
+ C_PLUS_PLUS
+ GO
+ R
+ SWIFT
+ PHP
<<enumeration>>
Difficulty
+ EASY
+ MEDIUM
+ HARD
Question
- question: String
- answerChoices: ArrayList<String>
- correctAnswer: int
+ Question(String question, String correct_answer)
+ evaluate(int userAnswer): boolean
+getQuestion(): String
User
- userid: UUID
- username: String
- password: String
- email: String
- firstName: String
- lastName: String
- registeredCourses: ArrayList<Course>
- gradeBook: HashMap<UUID, CourseInfo>
- favoriteAuthors: ArrayList<CourseCreator>
- completedCourses: ArrayList<Course>
+ User(String firstName, String lastName, String email, String Username, String password)
+ addCourse(Course course): void
+ removeCourse(Course course): void
+ addCompletedCourse(Course course): void
+ addFavAuthor(CourseCreator author): void
+ removeFavAuthor(CourseCreator author): void
+ updateCourseGrade(UUID courseid, String moduleName, double grade): void
Course
- courseid: UUID
- rating: double
- title: String
- author: User
- comments: ArrayList<Comment>
- reviews: ArrayList<Review>
- topic: Language
- description: String
- modules: ArrayList<Module>
- final: Assessment
- difficulty: Difficulty
- moduleCount: int
+ Course(String title, User author, Langauge Topic, String description, ArrayList<Module> modules, Assessment final, Difficulty difficulty)
+ addModule(Lesson lesson): void
+ addComment(Comment comment): void
+ addReview(Review review): void
+ getModule(int index): Lesson
+ getFinal(): Assessment
+ updateRating(int rating): void
+ createCertificate(): void
Assessment
- questions: ArrayList<Question>
- size: int
+ Assessment(int size)
+ addQuestion(Question question): void
+ getQuestions(): ArrayList<Question>
CourseInfo
- course: Course
- grade: double
- moduleGrades: HashMap<String, double>
+ CourseInfo(Course course)
+ updateGrade(String moduleName, double grade): void
Module
- title: String
- quiz: Assessment
- comments: ArrayList<Comment>
- lessons: ArrayList<Lesson>
+ Module(String title, Assessment quiz)
+ addComment(Comment comment): void
+ addLesson(Lesson lesson): void
CourseCreator
- createdCourses: ArrayList<Course>
- rating: double
+ CourseCreator(String firstName, String lastName, String email, String Username, String password)
+ updateRating(int rating): void
Review
- author: User
- rating: int
- description: String
+ Review(User author, int rating, description String)
Comment
- author: User
- commentid: UUID
- description: String
- replies: ArrayList<Comment>
+ Comment(User author, String description, replies ArrayList<Comment>)
+ addReply(Comment comment): void
Lesson
- title: String
- content: String
+Lesson(String title, String content, String example)

LMS and UI



LMS

- user: User
- currCourse: Course
- currModule: Module
- currLesson: Lesson
- courseList: CourseList
- userList: UserList

+ LMS()
+ login(String username, String password): boolean
+ logout(): void
+ addUser(String firstName, String lastName, String email, String username, String password): void
+ addCourse(String title, Langauge Topic, String description, ArrayList<Module> modules, Assessment final, Difficulty difficulty): void
+ findCourses(String keyword): ArrayList<Course>
+ getAllCourses(): ArrayList<Course>
+ getRegisteredCourses(): ArrayList<Course>
+ getCreatedCourses(): ArrayList<Course>
+ getCompletedCourses(): ArrayList<Course>
+ checkUsername(String username): boolean
+ checkCourseTitle(String title): boolean
+ getModuleQuiz(): Assessment
+ getCourseFinal(): Assessment
+ evaluateAssessment(Assessment assessment, ArrayList<int> answers): void
+ updateGrade(UUID courseid, String moduleName, double grade): void
+ addCourseComment(String decription): void
+ addModuleComment(String decription): void
+ getCourseComments(): ArrayList<Comment>
+ getModuleComment(): ArrayList<Comment>
+ addReply(Comment comment, String decription): void
+ addReview(String description, int rating): void
+ getReviews(): ArrayList<Review>

LMSUI

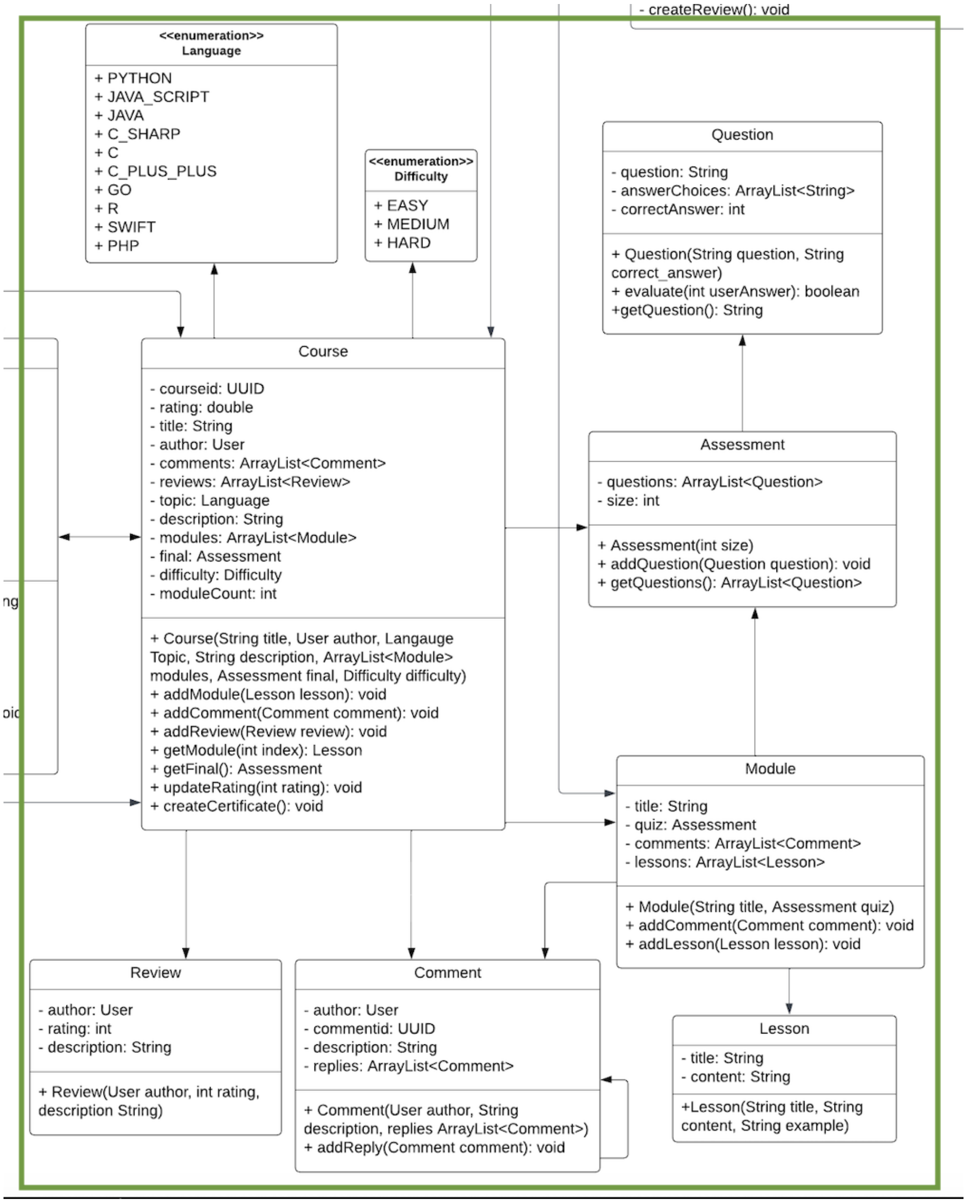
- scanner: Scanner
- application: LMS

+ run(): void
- displayMainMenu(): void
- displayLesson(Lesson lesson): void
- displayCourseDescription(Course course): void
- displayFinishedCourses(): void
- displayRegisteredCourses(): void
- displayAllCourses(): void
- searchCourses(): void
- navigateCourse(): void
- navigateModule(): void
- performAssessment(Assessment assessment): void
- createAccount(): void
- createCourse(): void
- displayComments(ArrayList<Comment>): void
- displayReviews(ArrayList<Review>): void
- createReply(Comment comment): void
- createModuleComment(): void
- createCourseComment(): void
- createReview(): void

Description:

This part of the system handles the high-level interactions between the user and the LMS. The user will be able to create an account, navigate the menu, interact with courses, leave comments, etc. User input will be handled by the system through a Scanner object in the UI. The UI's primary job is to prompt the user with options about how to navigate the LMS. Then, the UI contacts the LMS class to handle all of the high level operations that the user may choose to do through the UI.

Courses

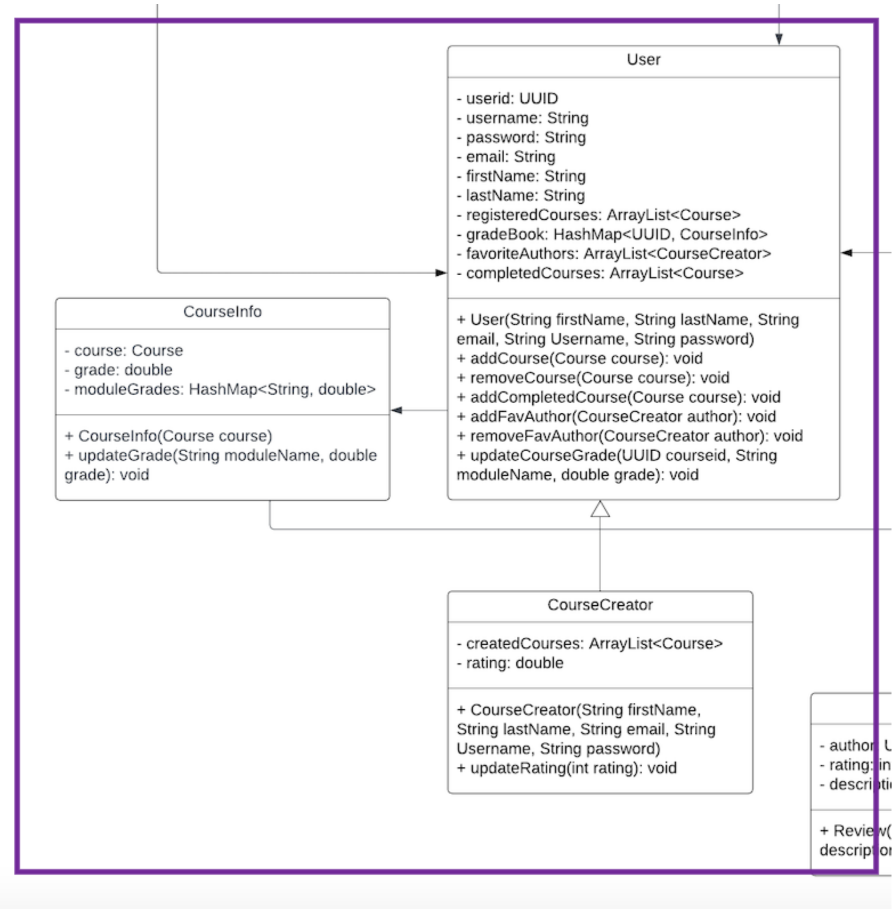


- createReview(): void
<<enumeration>>
Language
+ PYTHON
+ JAVA_SCRIPT
+ JAVA
+ C_SHARP
+ C
+ C_PLUS_PLUS
+ GO
+ R
+ SWIFT
+ PHP
<<enumeration>>
Difficulty
+ EASY
+ MEDIUM
+ HARD
Question
- question: String
- answerChoices: ArrayList<String>
- correctAnswer: int
+ Question(String question, String correct_answer)
+ evaluate(int userAnswer): boolean
+getQuestion(): String
Course
- courseid: UUID
- rating: double
- title: String
- author: User
- comments: ArrayList<Comment>
- reviews: ArrayList<Review>
- topic: Language
- description: String
- modules: ArrayList<Module>
- final: Assessment
- difficulty: Difficulty
- moduleCount: int
+ Course(String title, User author, Langauge Topic, String description, ArrayList<Module> modules, Assessment final, Difficulty difficulty)
+ addModule(Lesson lesson): void
+ addComment(Comment comment): void
+ addReview(Review review): void
+ getModule(int index): Lesson
+ getFinal(): Assessment
+ updateRating(int rating): void
+ createCertificate(): void
Assessment
- questions: ArrayList<Question>
- size: int
+ Assessment(int size)
+ addQuestion(Question question): void
+ getQuestions(): ArrayList<Question>
ng
oid
Module
- title: String
- quiz: Assessment
- comments: ArrayList<Comment>
- lessons: ArrayList<Lesson>
+ Module(String title, Assessment quiz)
+ addComment(Comment comment): void
+ addLesson(Lesson lesson): void
Review
- author: User
- rating: int
- description: String
+ Review(User author, int rating, description String)
Comment
- author: User
- commentid: UUID
- description: String
- replies: ArrayList<Comment>
+ Comment(User author, String description, replies ArrayList<Comment>)
+ addReply(Comment comment): void
Lesson
- title: String
- content: String
+Lesson(String title, String content, String example)

Description:

This part of the system handles the courses. Several classes such as Assessment, Modules, Questions, Lessons, Reviews, Comments, and enumerations are used in the Course class where all of the course details are defined. Users will be able create courses, add modules, comments, reviews, interact with these variables, and more in the Course class. Modules will have Lessons which contain a title and content. Comments will have IDs, authors, and descriptions which are used by courses and modules. Finally, the reviews have an author, rating for the course, and description. Reviews will be made by users for the course.

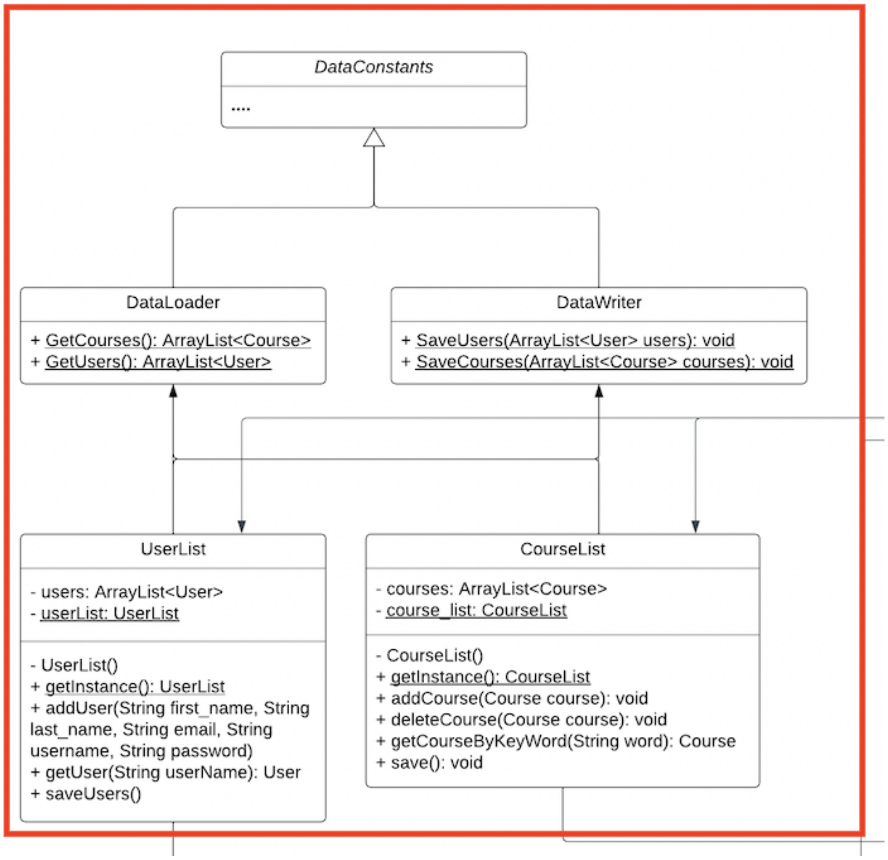
User Classes


User
- userid: UUID
- username: String
- password: String
- email: String
- firstName: String
- lastName: String
- registeredCourses: ArrayList<Course>
- gradeBook: HashMap<UUID, CourseInfo>
- favoriteAuthors: ArrayList<CourseCreator>
- completedCourses: ArrayList<Course>
+ User(String firstName, String lastName, String email, String Username, String password)
+ addCourse(Course course): void
+ removeCourse(Course course): void
+ addCompletedCourse(Course course): void
+ addFavAuthor(CourseCreator author): void
+ removeFavAuthor(CourseCreator author): void
+ updateCourseGrade(UUID courseid, String moduleName, double grade): void
CourseInfo
- course: Course
- grade: double
- moduleGrades: HashMap<String, double>
+ CourseInfo(Course course)
+ updateGrade(String moduleName, double grade): void
CourseCreator
- createdCourses: ArrayList<Course>
- rating: double
+ CourseCreator(String firstName, String lastName, String email, String Username, String password)
+ updateRating(int rating): void
- author
- rating:
- descri
+ Review(
descript

Description:

This part of the LMS covers the User, CourseCreator, and CourseInfo classes. The User stores information such as userid, username, password, email, and more. It can be used to update information in the CourseInfo class. The CourseInfo will contain a Course, a grade, and a HashMap of module names and their grades. Finally, the CourseCreator is a child class of the User and it stores the courses created as well as the CourseCreator's rating. These classes all interact and allow for User methods to run properly.

Writer and Loader

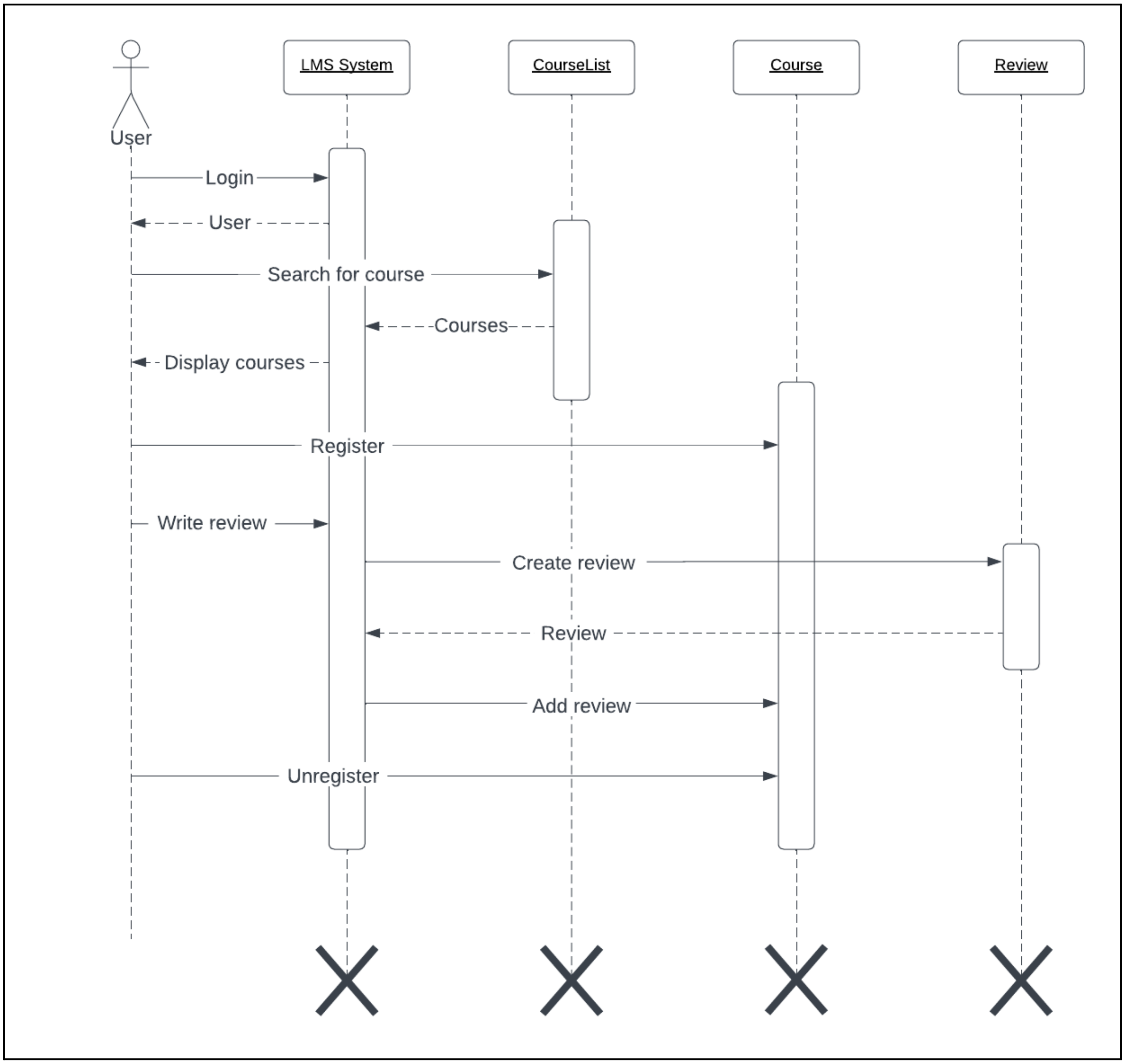

DataConstants
....
DataLoader
+ GetCourses(): ArrayList<Course>
+ GetUsers(): ArrayList<User>
DataWriter
+ SaveUsers(ArrayList<User> users): void
+ SaveCourses(ArrayList<Course> courses): void
UserList
- users: ArrayList<User>
- userList: UserList
- UserList()
+ getInstance(): UserList
+ addUser(String first_name, String last_name, String email, String username, String password)
+ getUser(String userName): User
+ saveUsers()
CourseList
- courses: ArrayList<Course>
- course_list: CourseList
- CourseList()
+ getInstance(): CourseList
+ addCourse(Course course): void
+ deleteCourse(Course course): void
+ getCourseByKeyWord(String word): Course
+ save(): void

Description:
This section of the system contains the lists of the Users and Courses, the loader and the writer, and the DataConstants class. The UserList and CourseList classes are singletons which hold lists of all the users and courses, respectively. These classes are important because they store the information for all users and courses in the system. These lists are used by the system to get a user's information when they sign in and to get courses that users search for. The DataWriter will write data from the users and courses into the relevant JSON files. The DataLoader will load the JSON file information into relevant objects when the system is started.

Sequence Diagrams

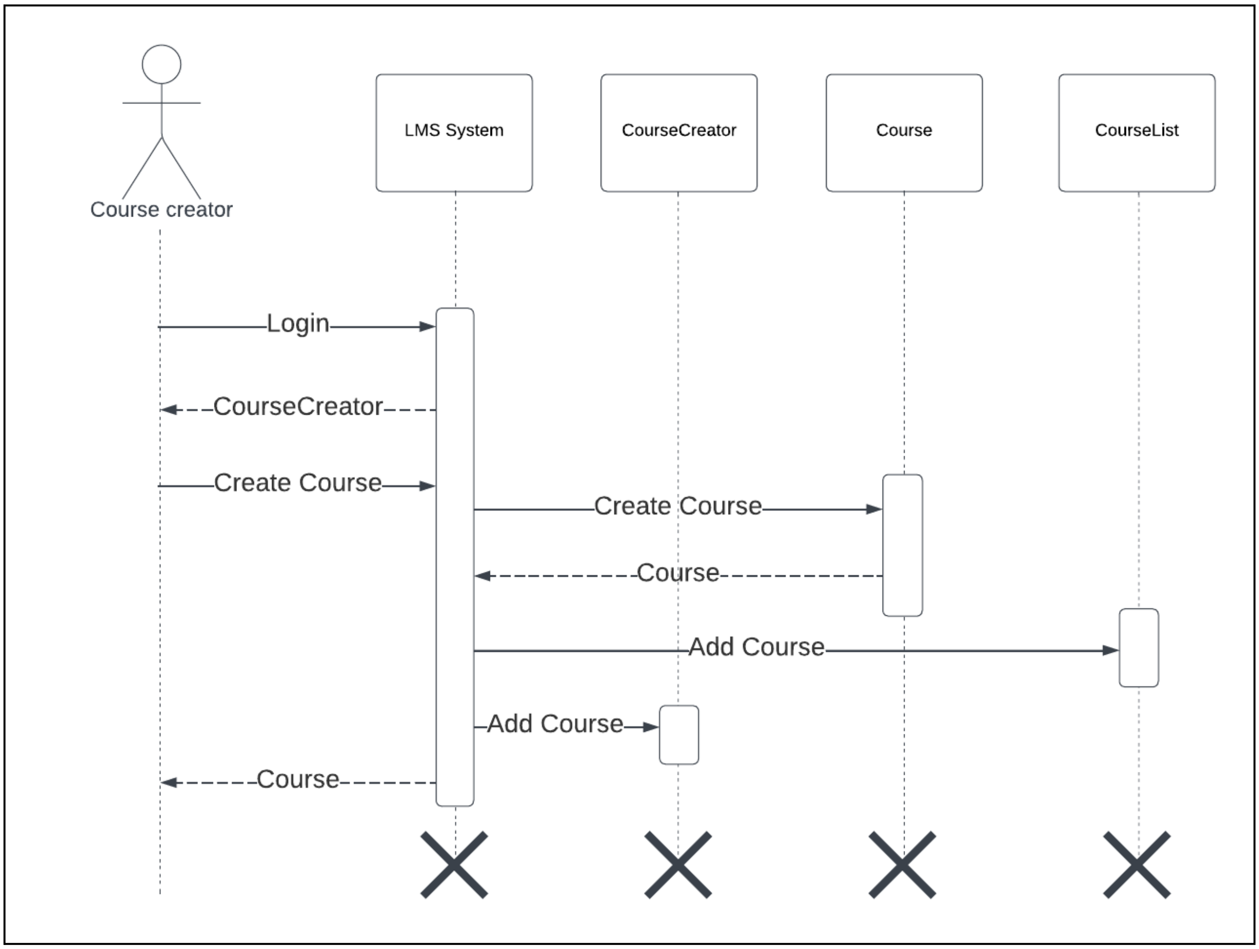
Scenario 1 - Leaving Review and Unregistering

The user will log into their account with the LMS. The user will search for a course and receive a list of relevant courses from the course list. These courses will be found with a simple keyword comparison between the searched word and course titles. The user will register for a course and leave a review on that course. The user will then unregister from the course. After unregistration, the course will be removed from the user's list of enrolled courses.



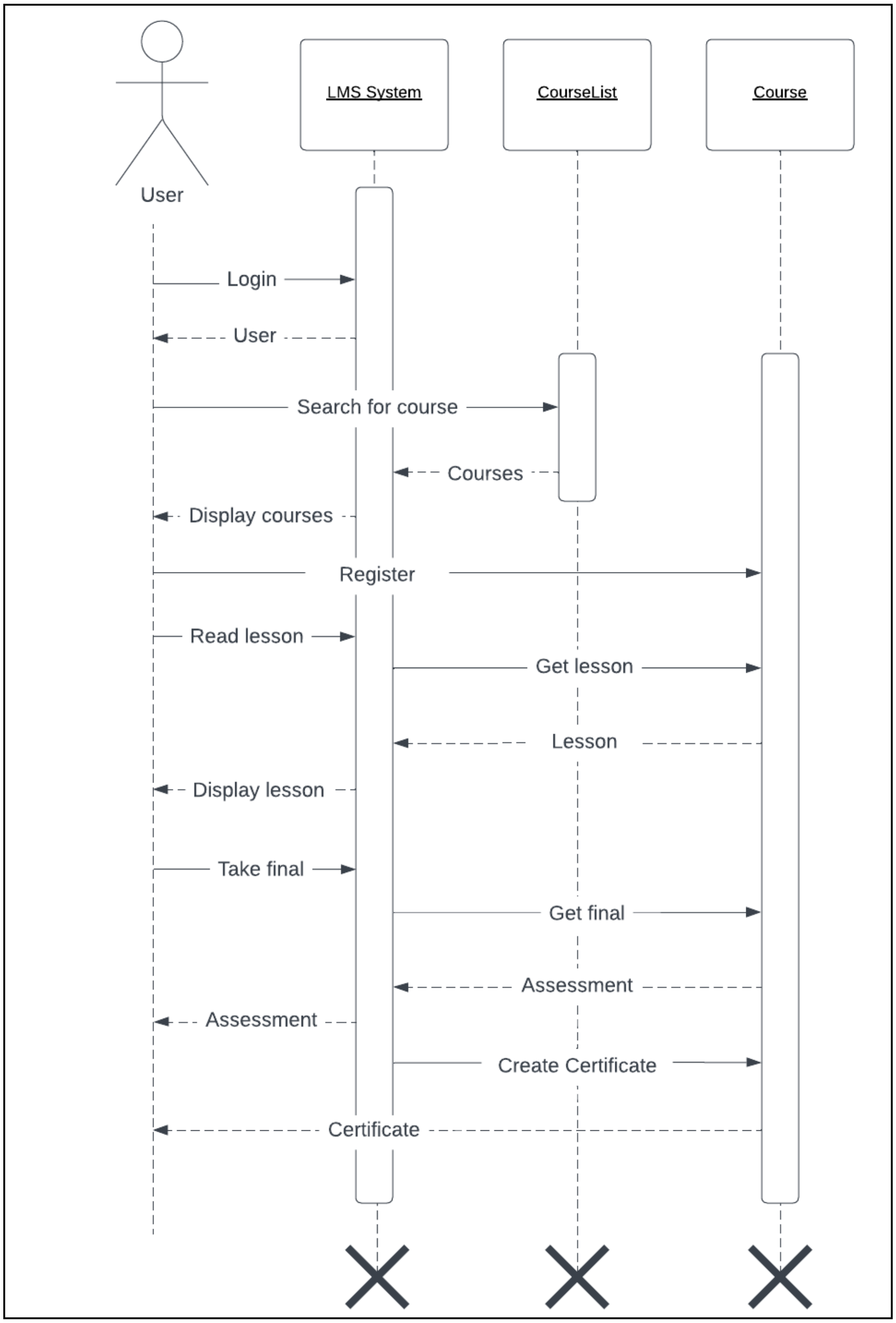
Scenario 2 - Creating Course

The user will log into their account with the LMS. The user will select Create Course from the LMS. This will allow the user to create a course by being prompted for their course's information and content. The LMS will then add the new course to the course list. The LMS will also add the course to the creator's list of created courses.



Scenario 3 - Completing a Course

The user will log into their account with the LMS. The user will search for a course and receive a list of relevant courses back from the Course List. The user will register for a course. The user will select "read lesson" in the LMS which will return it from the course. The system will then display the lesson's content. The user will select "take final" in the LMS which will return it from the course. The system will prompt the user with questions from the final. After the final is completed, the LMS will return a certificate upon completion of the course.


User
LMS System
CourseList
Course
Login
User
Search for course
Courses
Display courses
Register
Read lesson
Get lesson
Lesson
Display lesson
Take final
Get final
Assessment
Assessment
Create Certificate
Certificate

References

Our team’s software design document