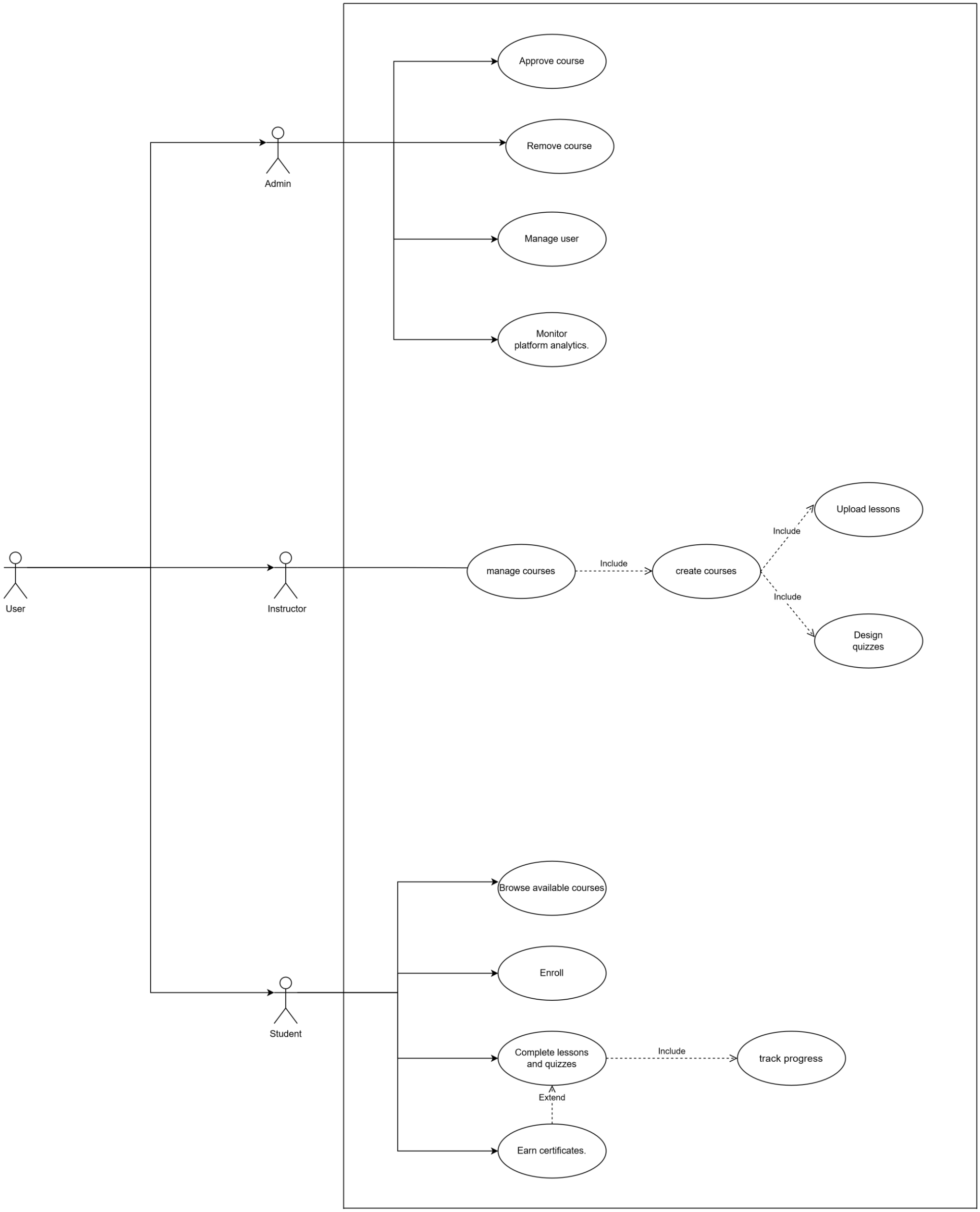
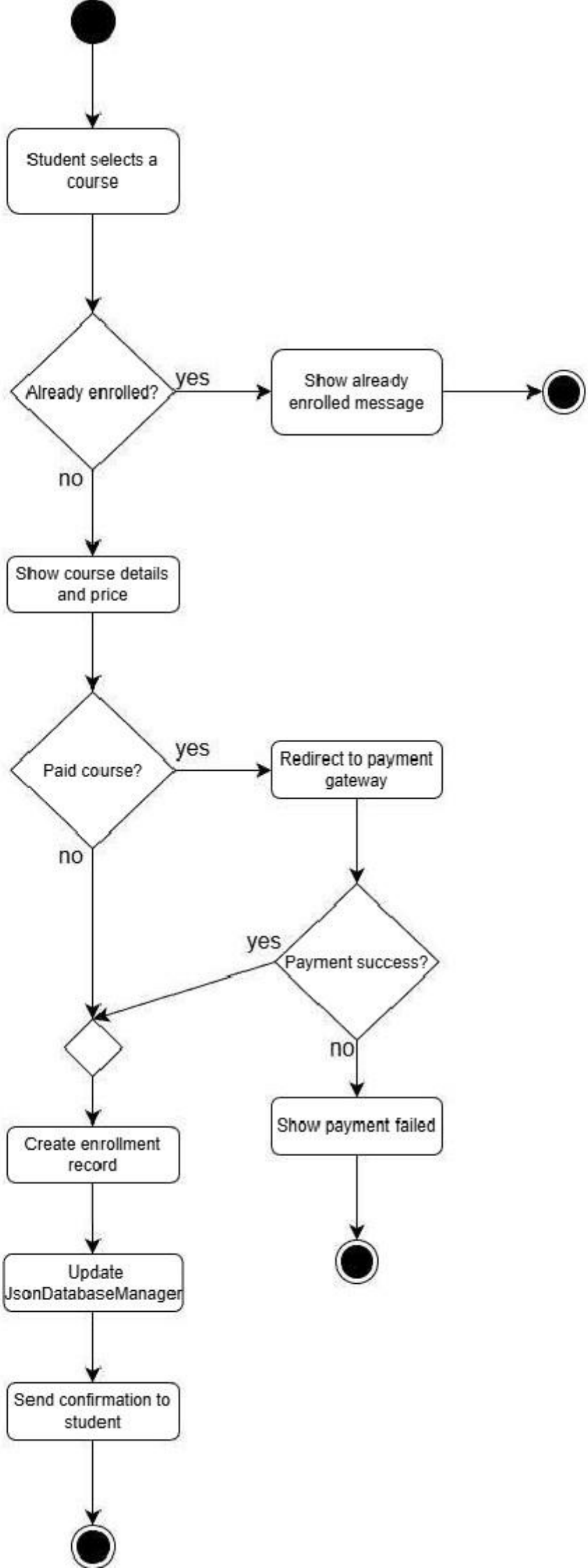

Admin
Approve course
Remove course
Manage user
Monitor
platform analytics.
User
Instructor
manage courses
Include
create courses
Include
Upload lessons
Include
Design
quizzes
Student
Browse available courses
Enroll
Complete lessons
and quizzes
Include
track progress
Extend
Earn certificates.

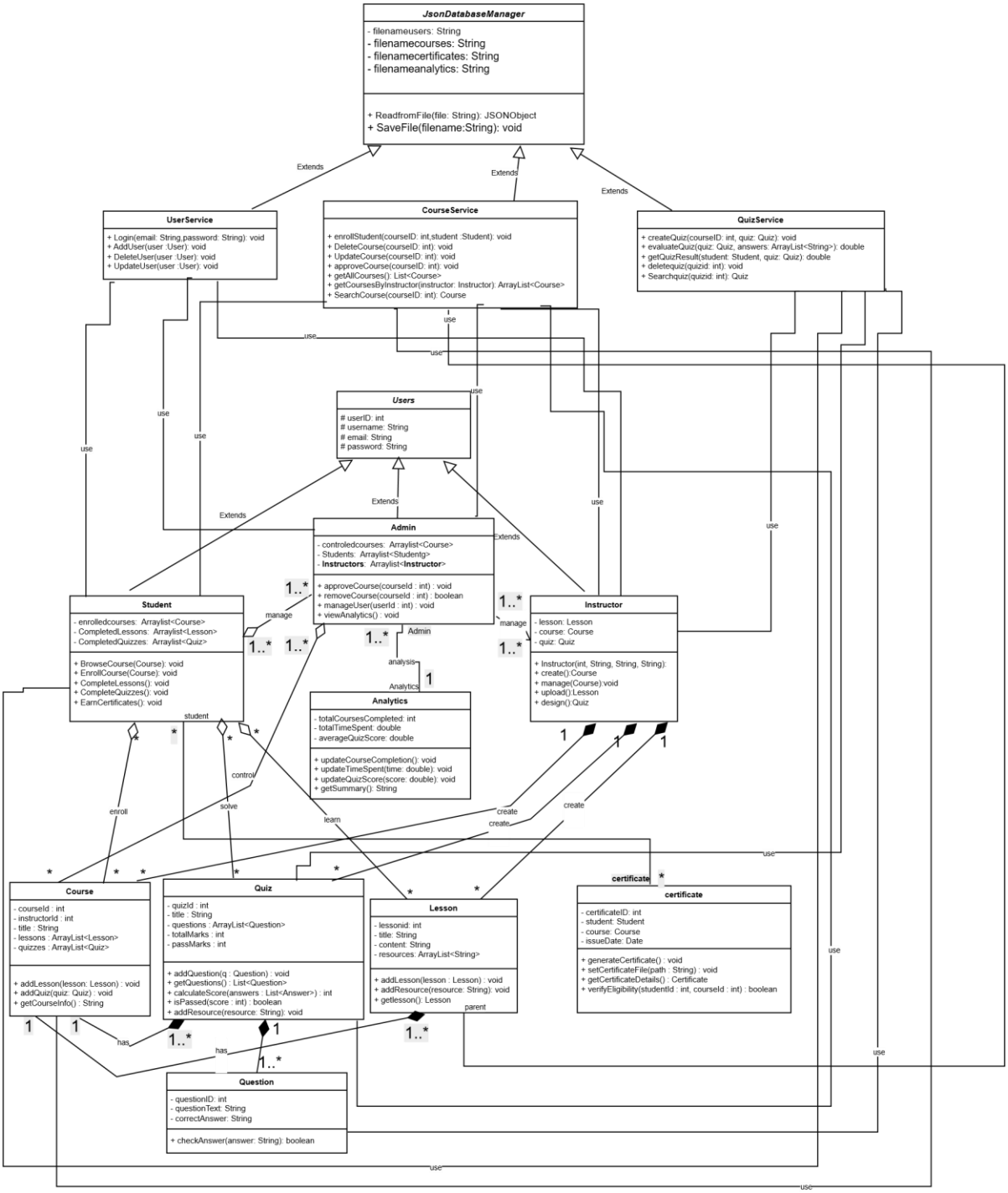
This use case diagram represents an online learning platform involving three primary actors: Admin, Instructor, and Student. The admin manages the overall system by approving or removing courses, handling user accounts, and monitoring platform analytics. The instructor is responsible for creating and managing courses, including uploading lessons and designing quizzes to support the learning process. Meanwhile, the Student can browse available courses, enroll, complete lessons and quizzes, track their progress, and earn certificates upon completion. Overall, the diagram demonstrates how each user interacts with the system to create an efficient and complete e-learning environment.


Student selects a course
Already enrolled?
yes
Show already enrolled message
no
Show course details and price
Paid course?
yes
Redirect to payment gateway
no
yes
Payment success?
no
Show payment failed
Create enrollment record
Update JsonDatabaseManager
Send confirmation to student

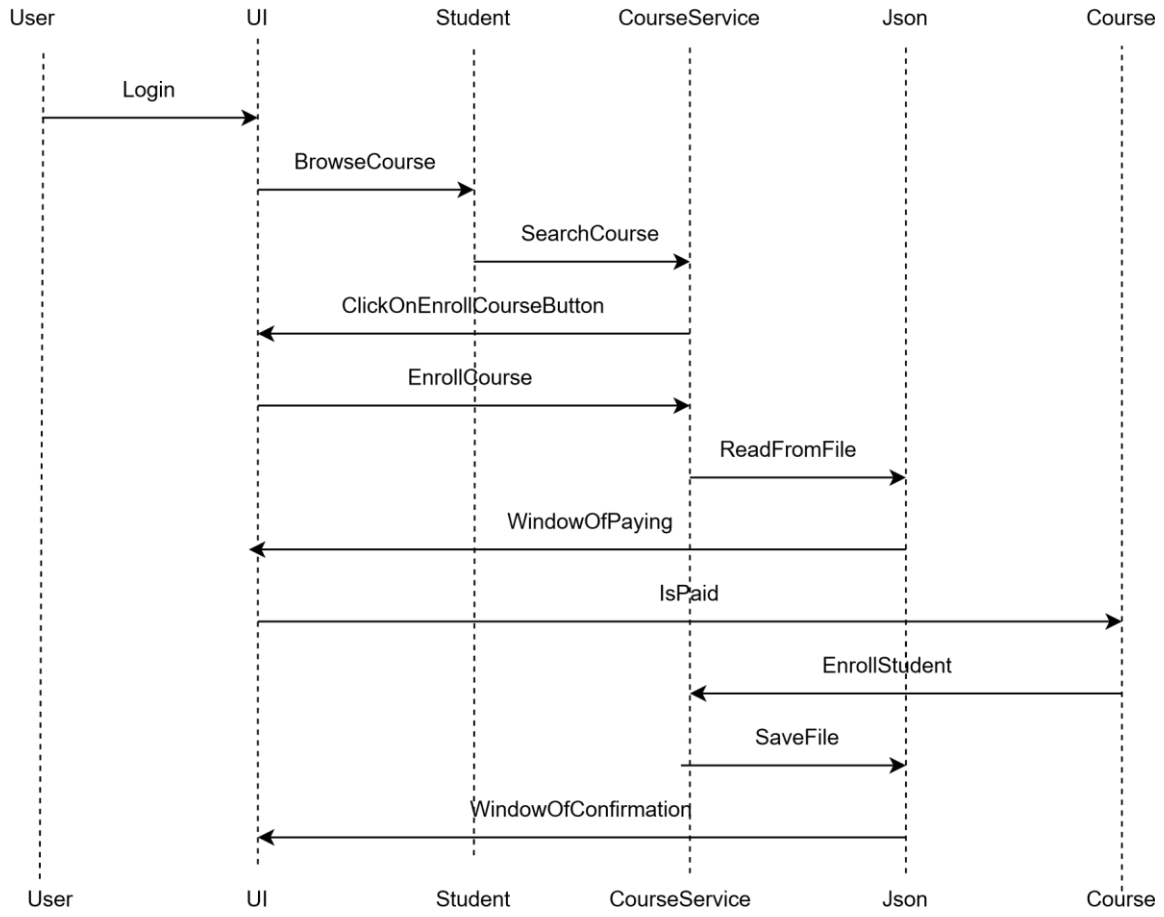
This Activity Diagram illustrates the workflow of the “Enroll in a Course” use case in the SkillForge platform.

It shows how a student selects a course, the system checks enrollment status, handles paid and free courses, and processes payments.

The diagram also demonstrates how successful enrollments update the database and send confirmation to the student, while invalid or failed cases terminate early.


JsonDatabaseManager
- filenameusers: String
- filenamecourses: String
- filenamecertificates: String
- filenameanalytics: String
+ ReadfromFile(file: String): JSONObject
+ SaveFile(filename:String): void
Extends
Extends
Extends
UserService
+ Login(email: String,password: String): void
+ AddUser(user :User): void
+ DeleteUser(user :User): void
+ UpdateUser(user :User): void
CourseService
+ enrollStudent(courseID: int,student :Student): void
+ DeleteCourse(courseID: int): void
+ UpdateCourse(courseID: int): void
+ approveCourse(courseID: int): void
+ getAllCourses(): List<Course>
+ getCoursesByInstructor(instructor: Instructor): ArrayList<Course>
+ SearchCourse(courseID: int): Course
QuizService
+ createQuiz(courseID: int, quiz: Quiz): void
+ evaluateQuiz(quiz: Quiz, answers: ArrayList<String>): double
+ getQuizResult(student: Student, quiz: Quiz): double
+ deletequiz(quizid: int): void
+ Searchquiz(quizid: int): Quiz
use
Users
userID: int
username: String
email: String
password: String
Extends
Admin
- controledcourses: Arraylist<Course>
- Students: Arraylist<Studentg>
- Instructors: Arraylist<Instructor>
+ approveCourse(courseId : int) : void
+ removeCourse(courseId : int) : boolean
+ manageUser(userId : int) : void
+ viewAnalytics() : void
Student
- enrolledcourses: Arraylist<Course>
- CompletedLessons: Arraylist<Lesson>
- CompletedQuizzes: Arraylist<Quiz>
+ BrowseCourse(Course): void
+ EnrollCourse(Course): void
+ CompleteLessons(): void
+ CompleteQuizzes(): void
+ EarnCertificates(): void
Instructor
- lesson: Lesson
- course: Course
- quiz: Quiz
+ Instructor(int, String, String, String):
+ create():Course
+ manage(Course):void
+ upload():Lesson
+ design():Quiz
manage
Admin
analysis
Analytics
Analytics
- totalCoursesCompleted: int
- totalTimeSpent: double
- averageQuizScore: double
+ updateCourseCompletion(): void
+ updateTimeSpent(time: double): void
+ updateQuizScore(score: double): void
+ getSummary(): String
student
control
enroll
solve
learn
create
create
create
Course
- courseId : int
- instructorId : int
- title : String
- lessons : ArrayList<Lesson>
- quizzes : ArrayList<Quiz>
+ addLesson(lesson: Lesson) : void
+ addQuiz(quiz: Quiz) : void
+ getCourseInfo() : String
Quiz
- quizId : int
- title : String
- questions : ArrayList<Question>
- totalMarks : int
- passMarks : int
+ addQuestion(q : Question) : void
+ getQuestions() : List<Question>
+ calculateScore(answers : List<Answer>) : int
+ isPassed(score : int) : boolean
+ addResource(resource: String): void
Lesson
- lessonid: int
- title: String
- content: String
- resources: ArrayList<String>
+ addLesson(lesson : Lesson) : void
+ addResource(resource: String): void
+ getlesson(): Lesson
parent
certificate
certificate
- certificateID: int
- student: Student
- course: Course
- issueDate: Date
+ generateCertificate() : void
+ setCertificateFile(path : String) : void
+ getCertificateDetails() : Certificate
+ verifyEligibility(studentId : int, courseId : int) : boolean
has
has
Question
- questionID: int
- questionText: String
- correctAnswer: String
+ checkAnswer(answer: String): boolean

This UML Class Diagram shows the object-oriented design of the SkillForge platform, including key components such as Users, Courses, Lessons, Quizzes, and Certificates. It illustrates how different user roles interact with these elements and how system modules connect, serving as a blueprint for development and ensuring a clear, organized design.



This diagram illustrates the sequence of interactions between system components in a course enrollment process. It shows how users, the UI, and backend services (like CourseService and Json) coordinate to handle actions such as browsing courses, enrolling, confirming payment, and saving enrollment data. The flow highlights key steps including SearchCourse, EnrollCourse, and EnrollStudent, demonstrating how the system manages both paid and unpaid course workflows.