HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY

GROUP 7

TUTOR SUPPORT SYSTEM

GROUP PROJECT

GROUP MEMBERS

Stu No.	Name	Contribution
2353057	Võ Hoàng Sơn	100%
2352254	Hoàng Hải Đăng	100%
2352991	Lâm Minh Tùng Quân	100%
2353053	Phan Quốc Đại Sơn	100%
2352787	Trương Gia Kỳ Nam	100%
2352938	Nguyễn Hữu Phúc	100%
2352856	Châu Anh Nhật	100%

Table of contents

Table of contents

Introduction

Tutor support HCMUT

Efficient Resource &
Cost Management

Administration &
Monitoring

Promote Accessibility &
Security

Context & Proposed Solution

Context (The Problem):

- **HCMUT offers a diverse but challenging curriculum, from foundation courses to specialized modules.**
- **Students often struggle due to the gap in teaching methods (freshmen) or subject complexity (seniors)**
-

Solution: Tutor Support System.

Key Objectives:

- **Connection: A digital platform linking students with suitable tutors.**
- **Support: Facilitates progress tracking and personalized academic guidance.**
- **Management: Provides analytical tools for coordinators to ensure quality and efficiency**

System Stakeholders & Key Roles

Student:

- Registers for tutoring & mentoring.
- Attends sessions (Online/Offline) and accesses HCMUT_LIBRARY resources .
- Evaluates session quality & tutor performance.

Tutor:

- Sets availability & organizes advising sessions.
- Shares materials and tracks student learning progress .
- Note: Roles are context-dependent (A user can be a Tutor in one course and a Student in another) .

Coordinator:

- Manages registration flows & oversees AI/Manual matching.
- Monitors quality via feedback & generates operational reports .

Project Scope & Boundaries

Core Functionality (In-Scope):

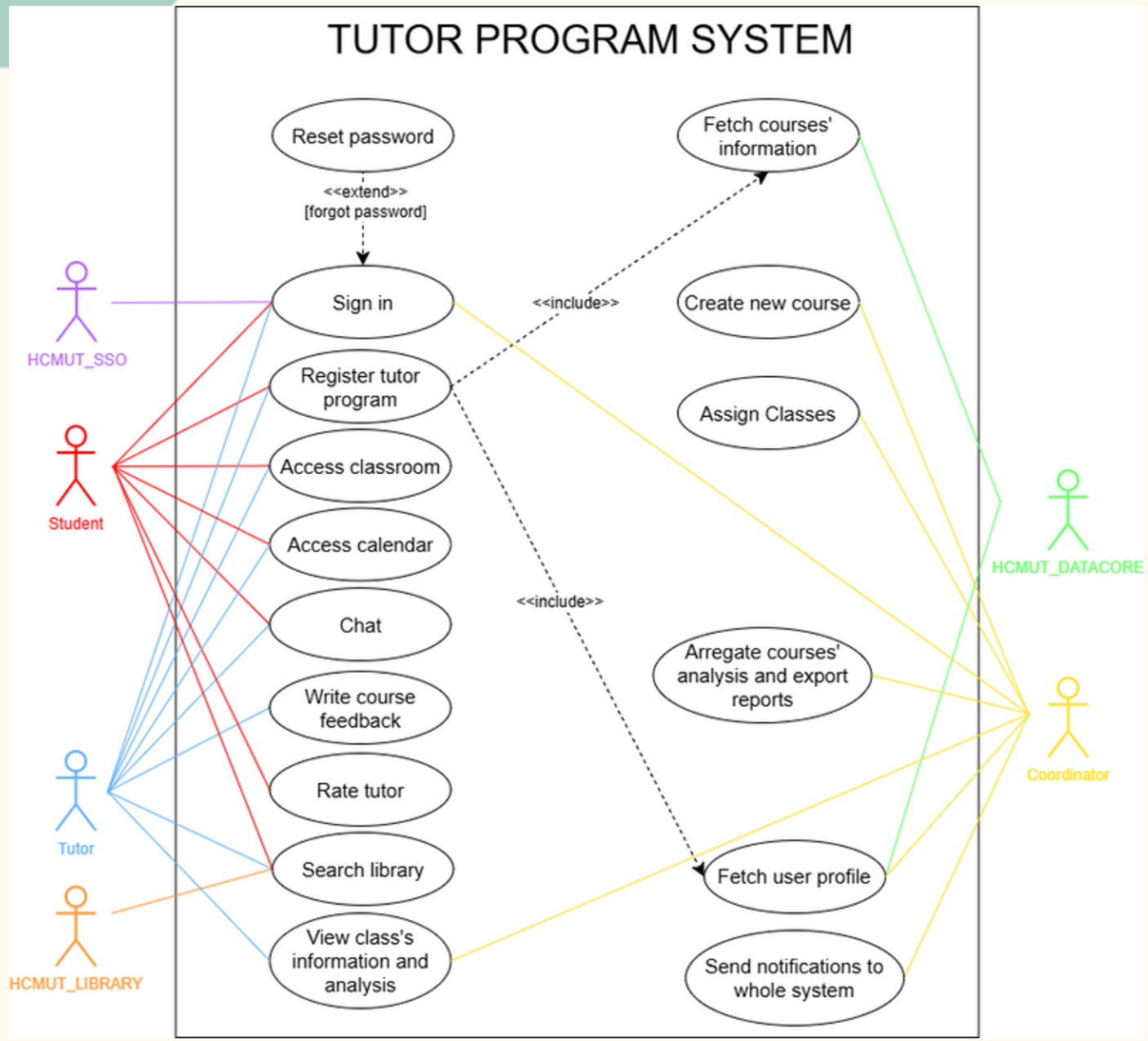
- **Management: Profile management for Tutors/Students and AI/Manual matching mechanisms.**
- **Operations: Scheduling (Online/In-class), attendance tracking, and session notifications.**
- **Academic Support: Virtual classrooms for content delivery, exercises, and non-academic mentoring.**
- **Quality Assurance: Feedback collection and analytical reporting for Coordinators.**
- **Integration: Secure HCMUT_SSO, DATACORE synchronization, and HCMUT_LIBRARY access.**

Exclusions (Out-of-Scope):

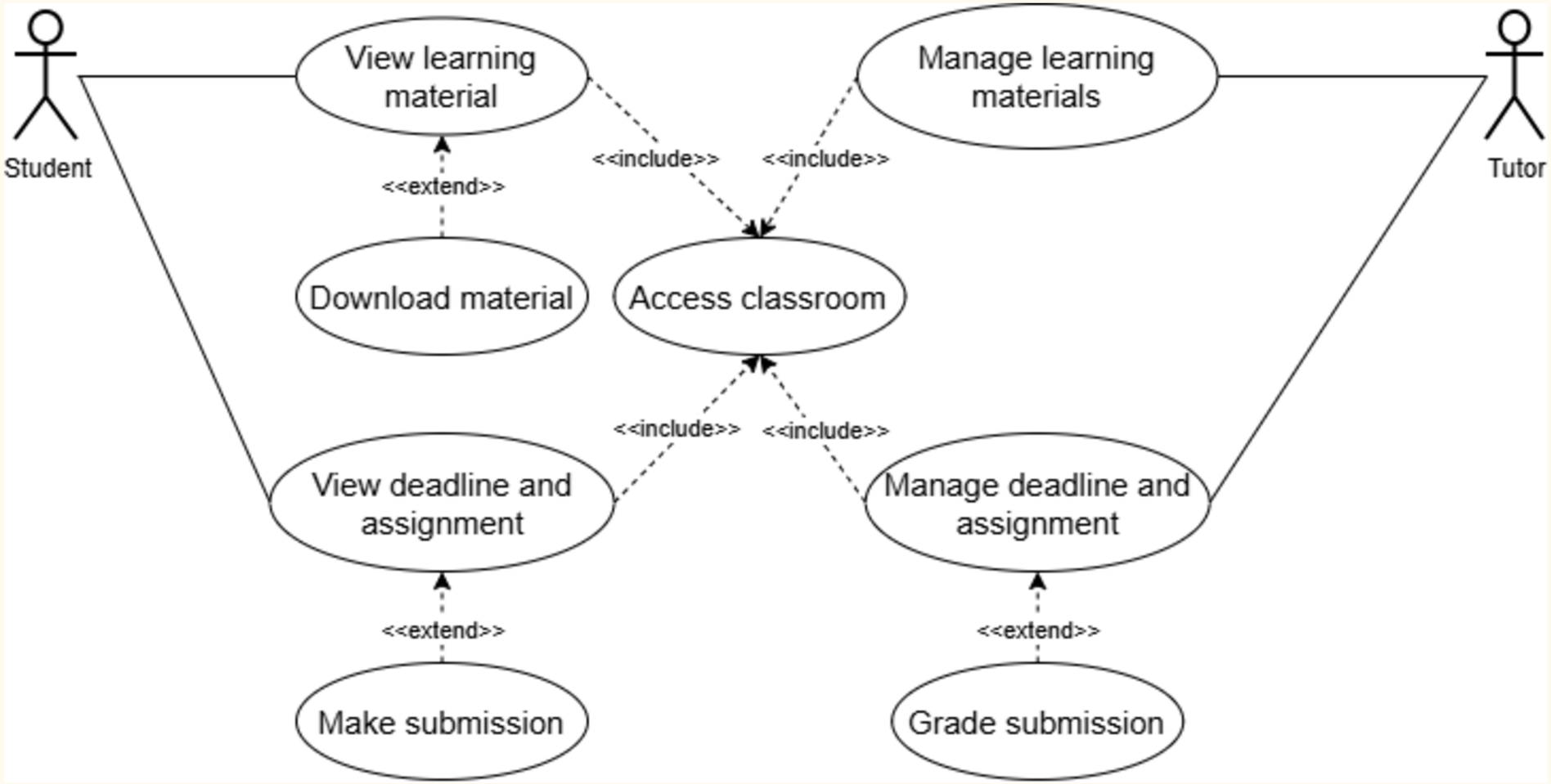
- **AI-driven personalized learning paths.**
- **Community features (e.g., tutor discussion boards).**

Table of contents

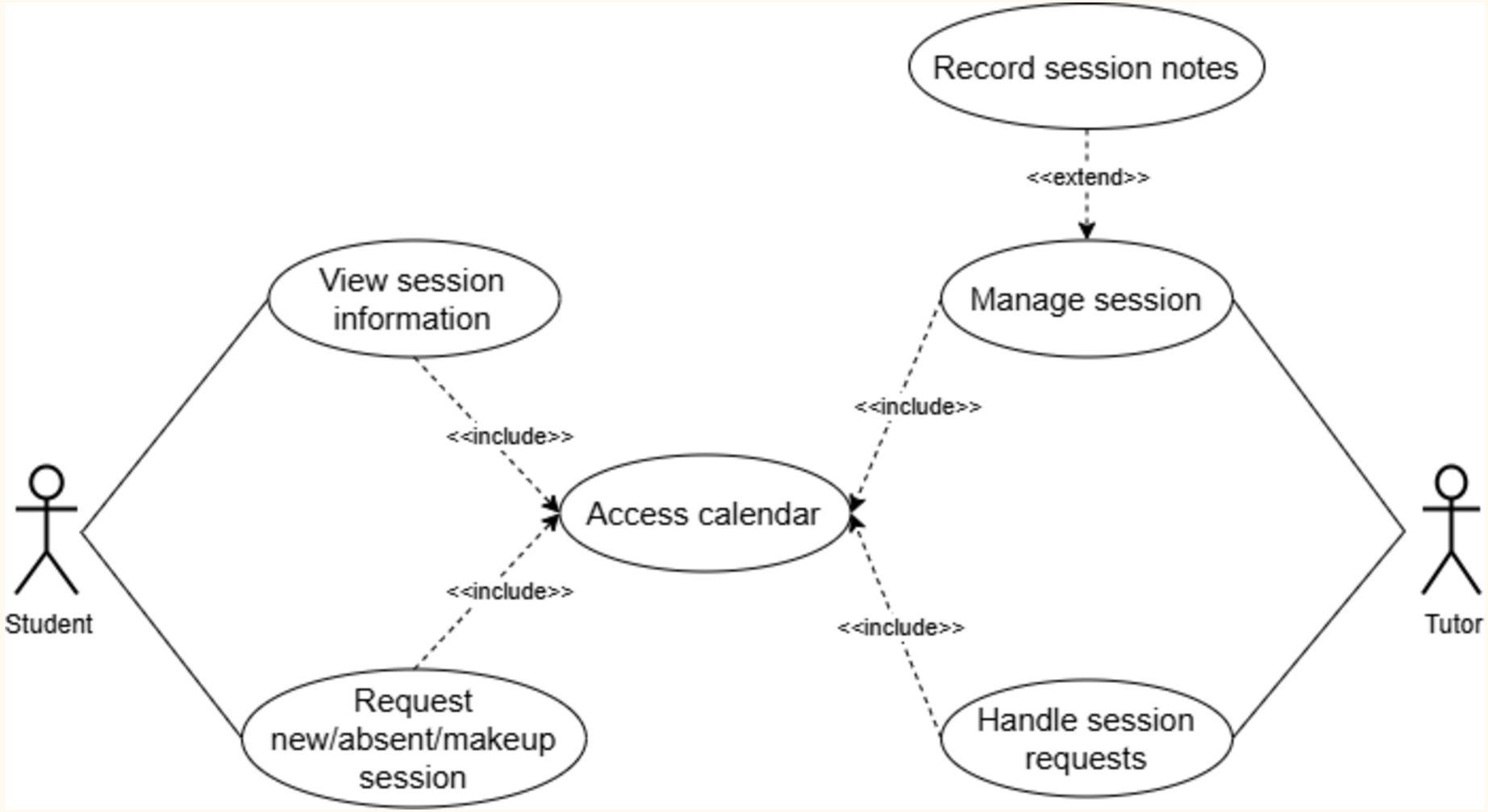
Use case diagram



Whole System UC diagram



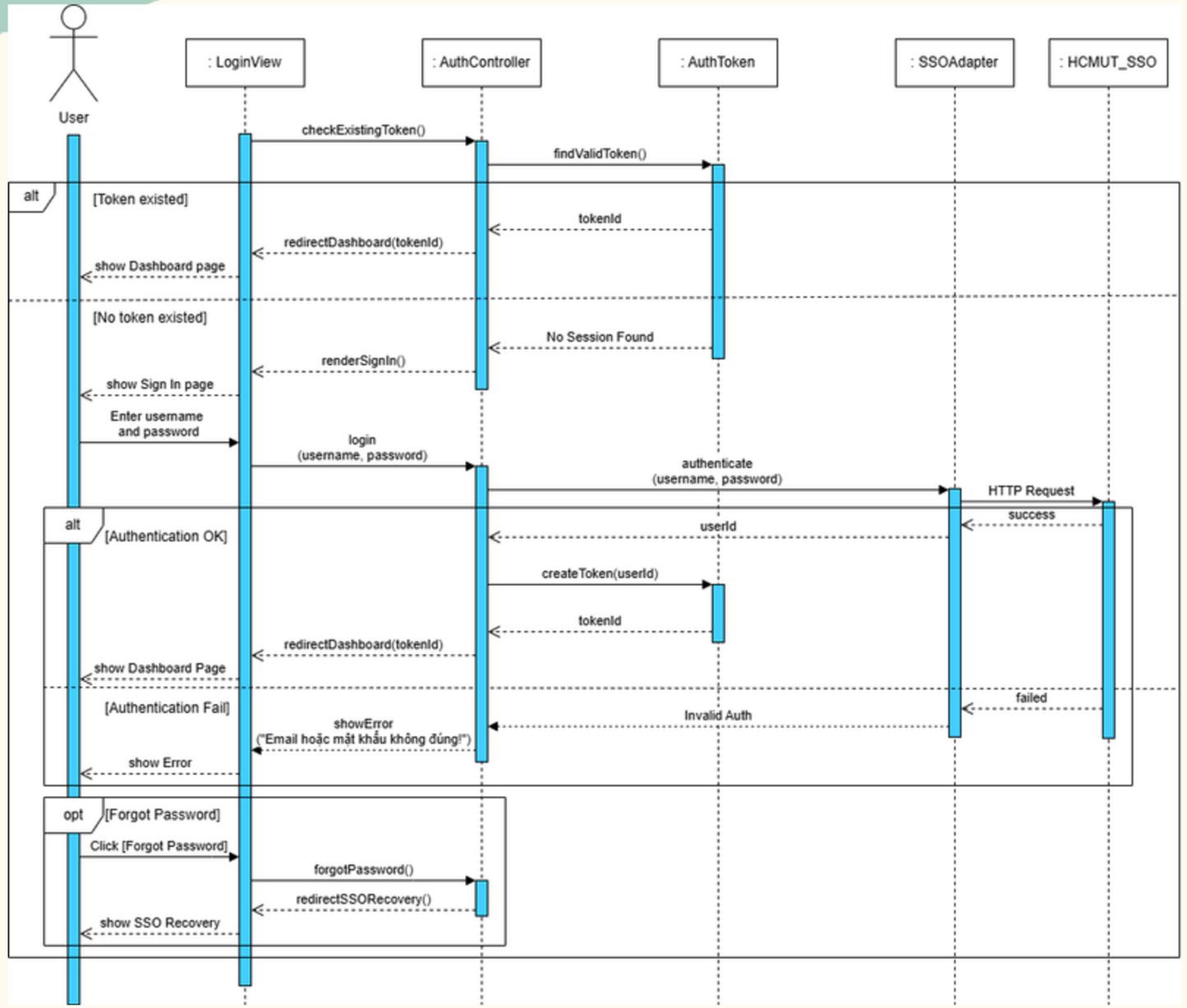
Material UC diagram



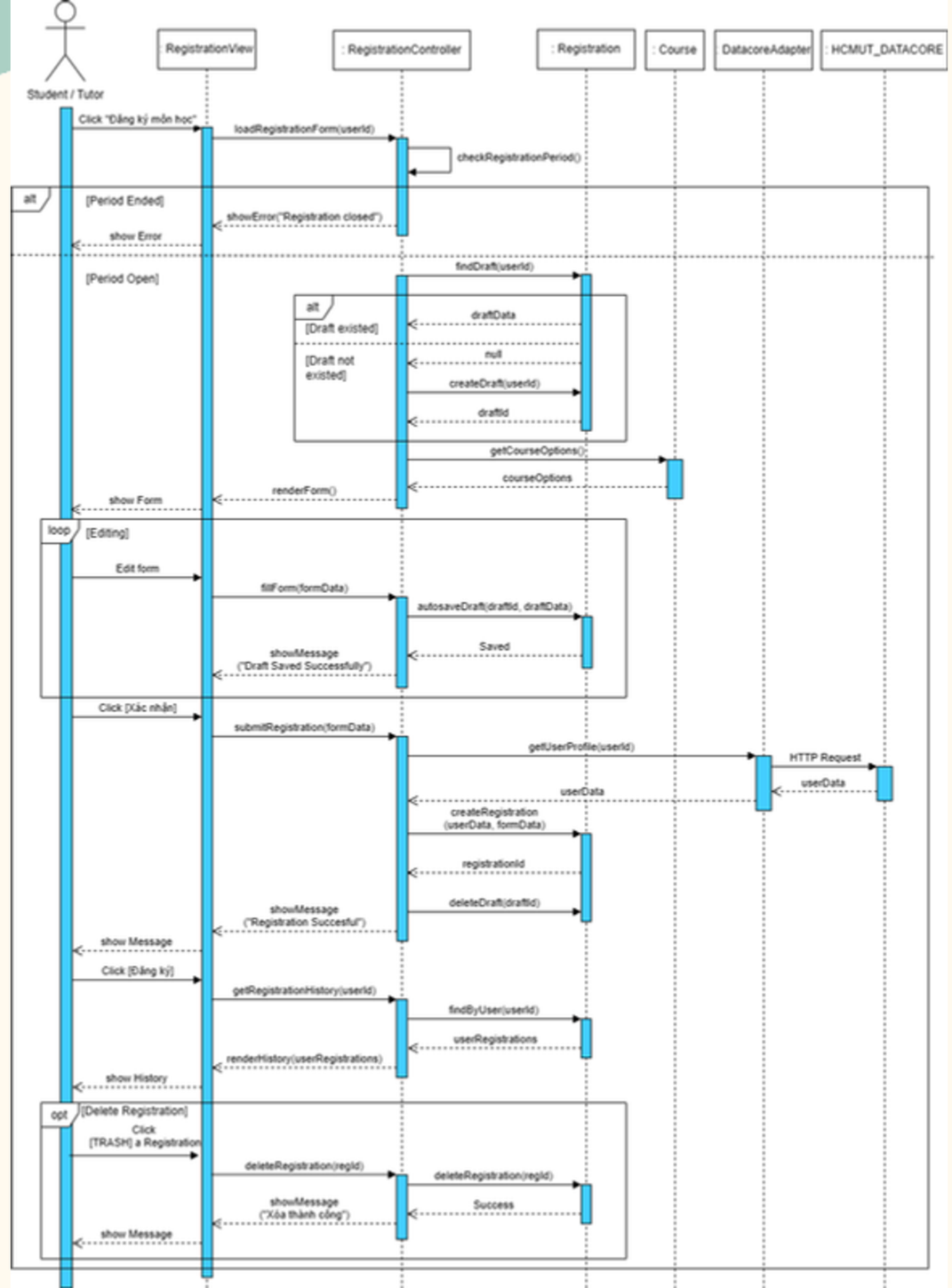
Session UC diagram

Table of contents

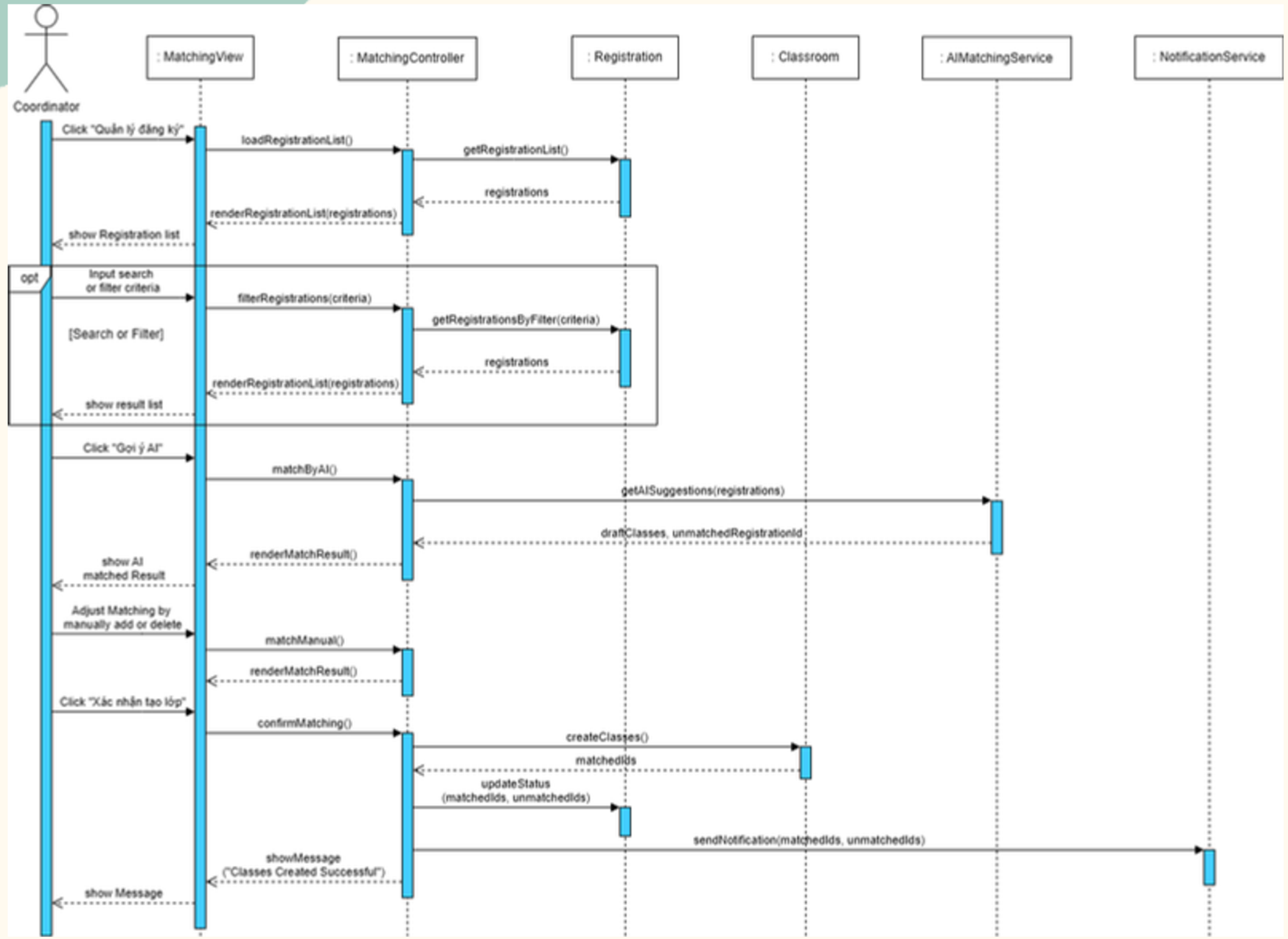
Sequence diagram



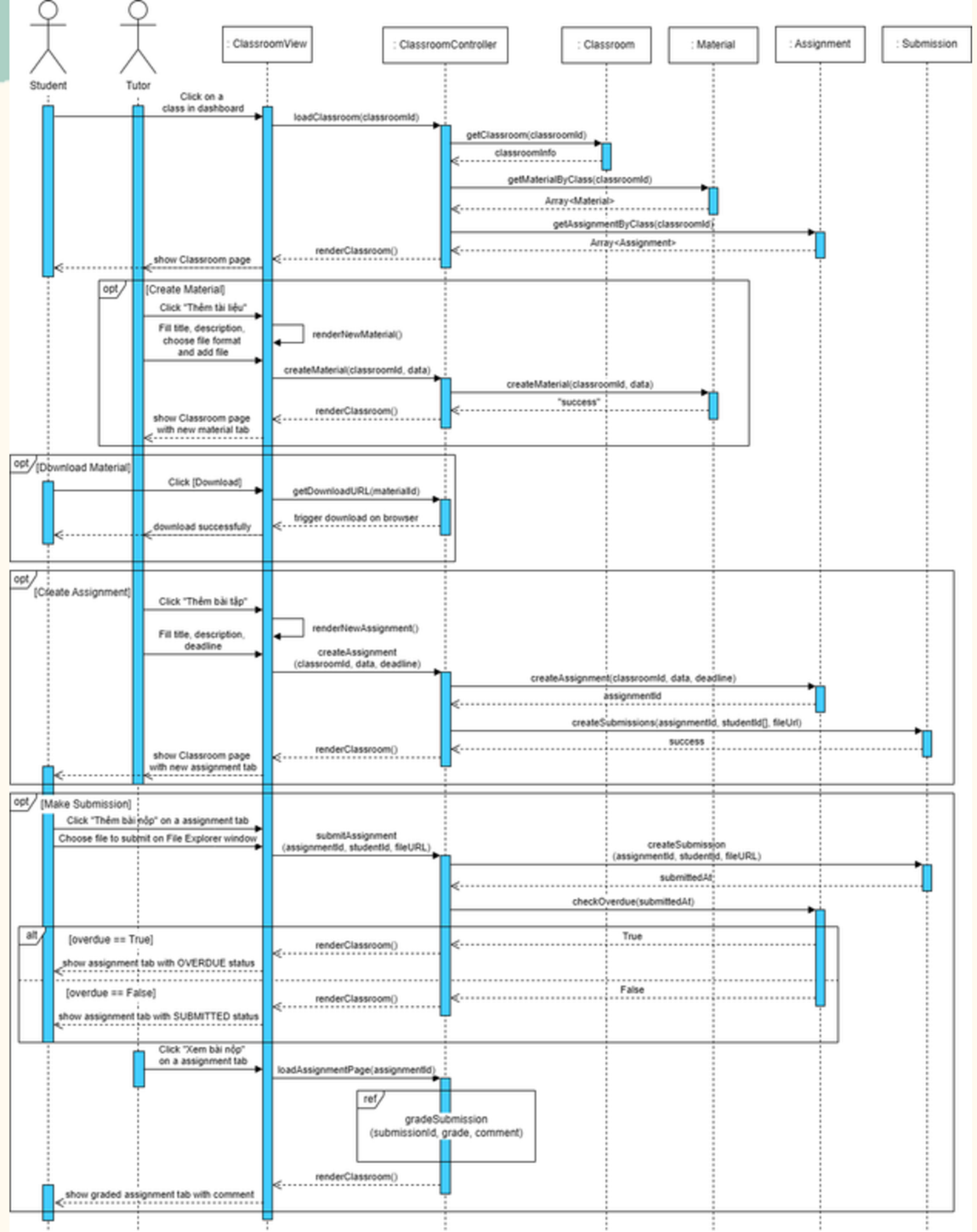
Login sequence diagram



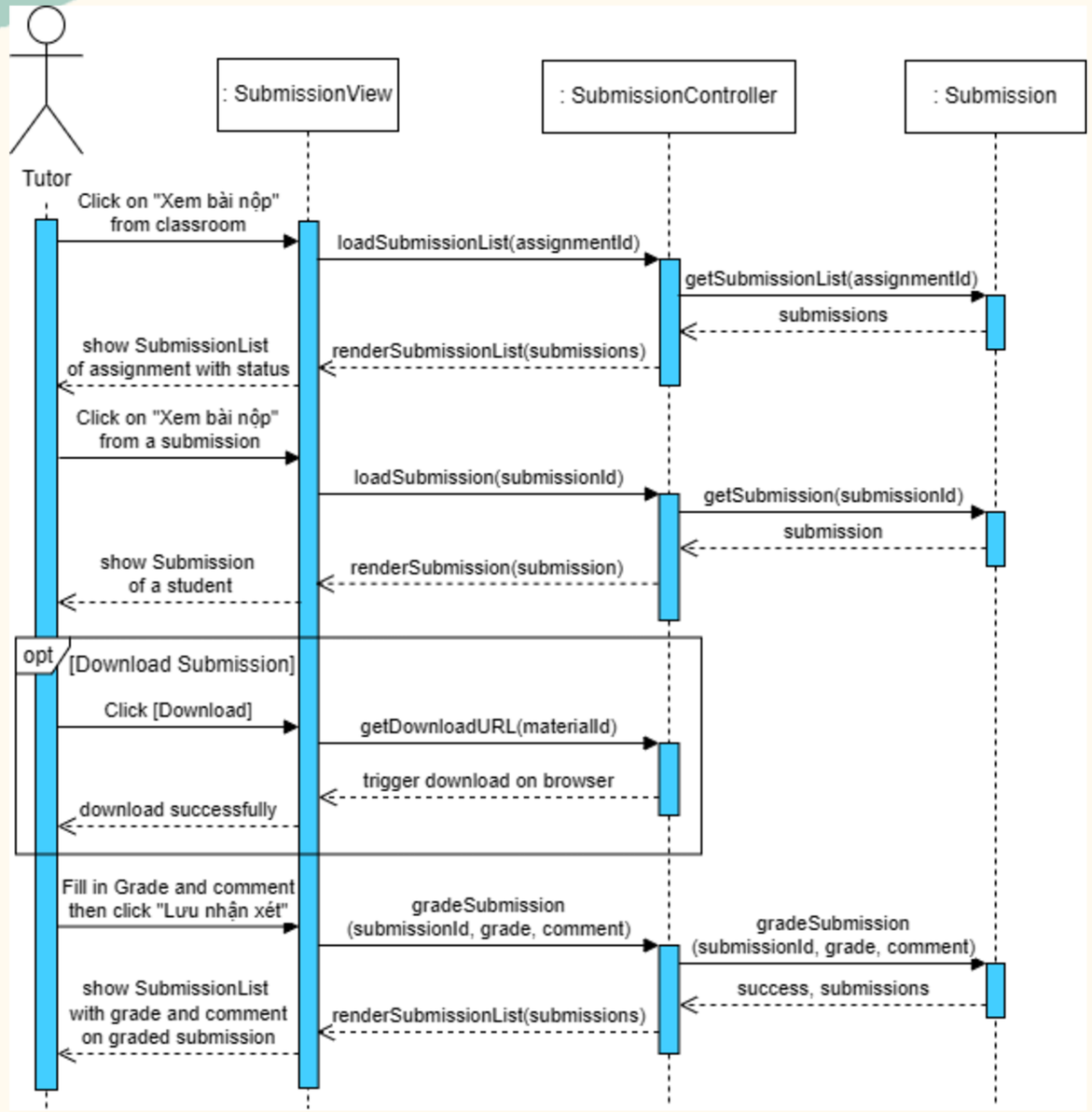
Registration sequence diagram



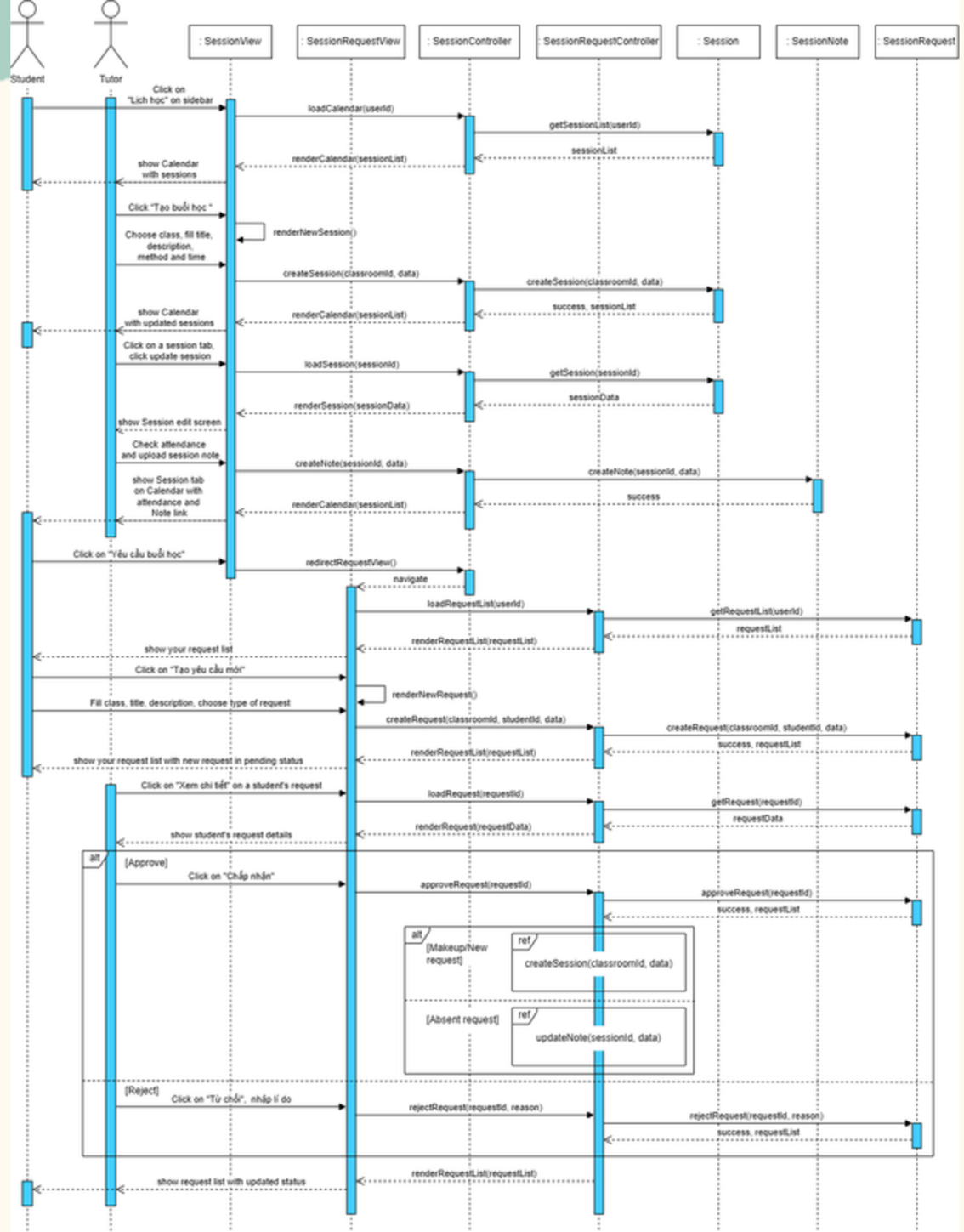
Assign class sequence diagram



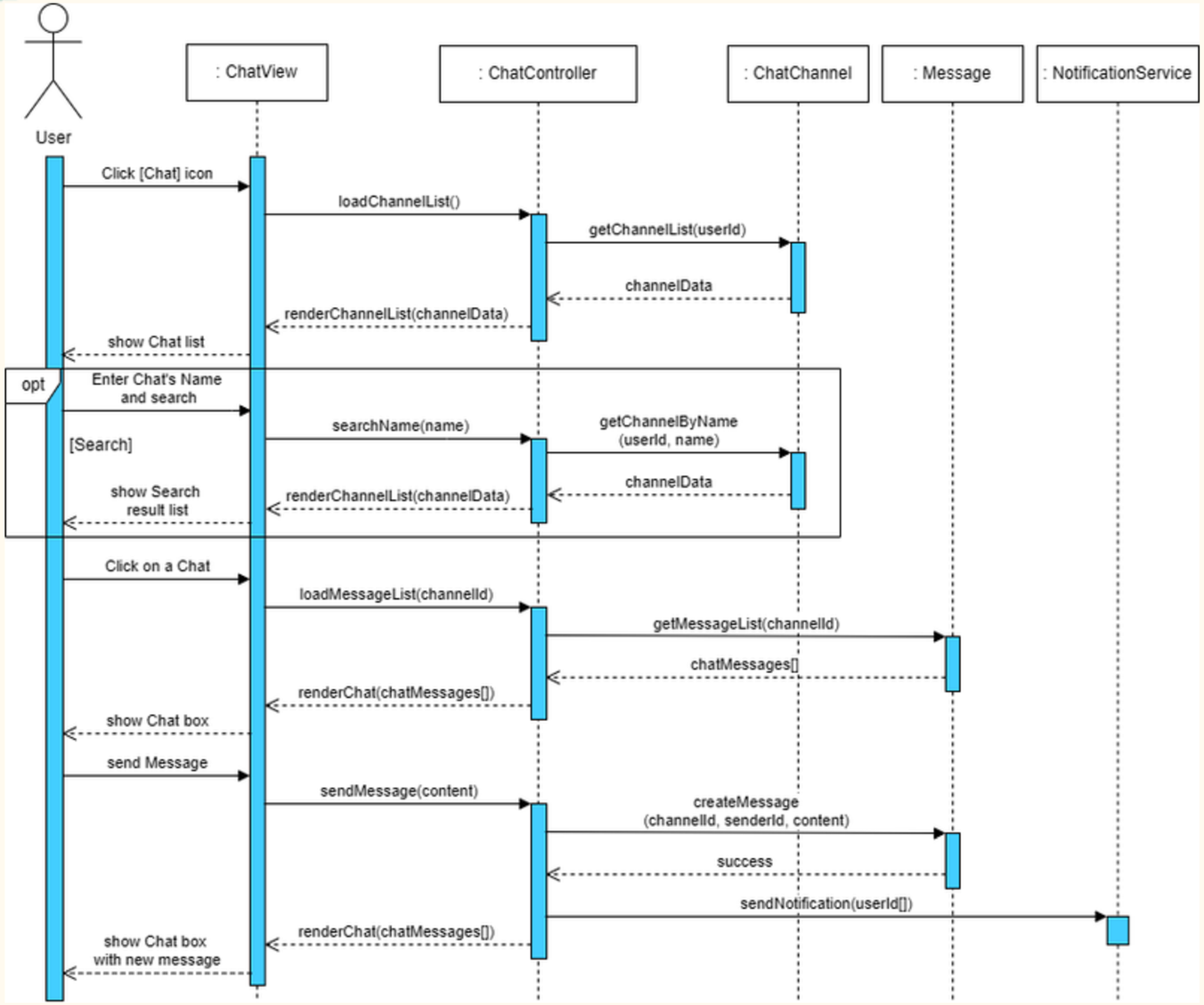
Material sequence diagram



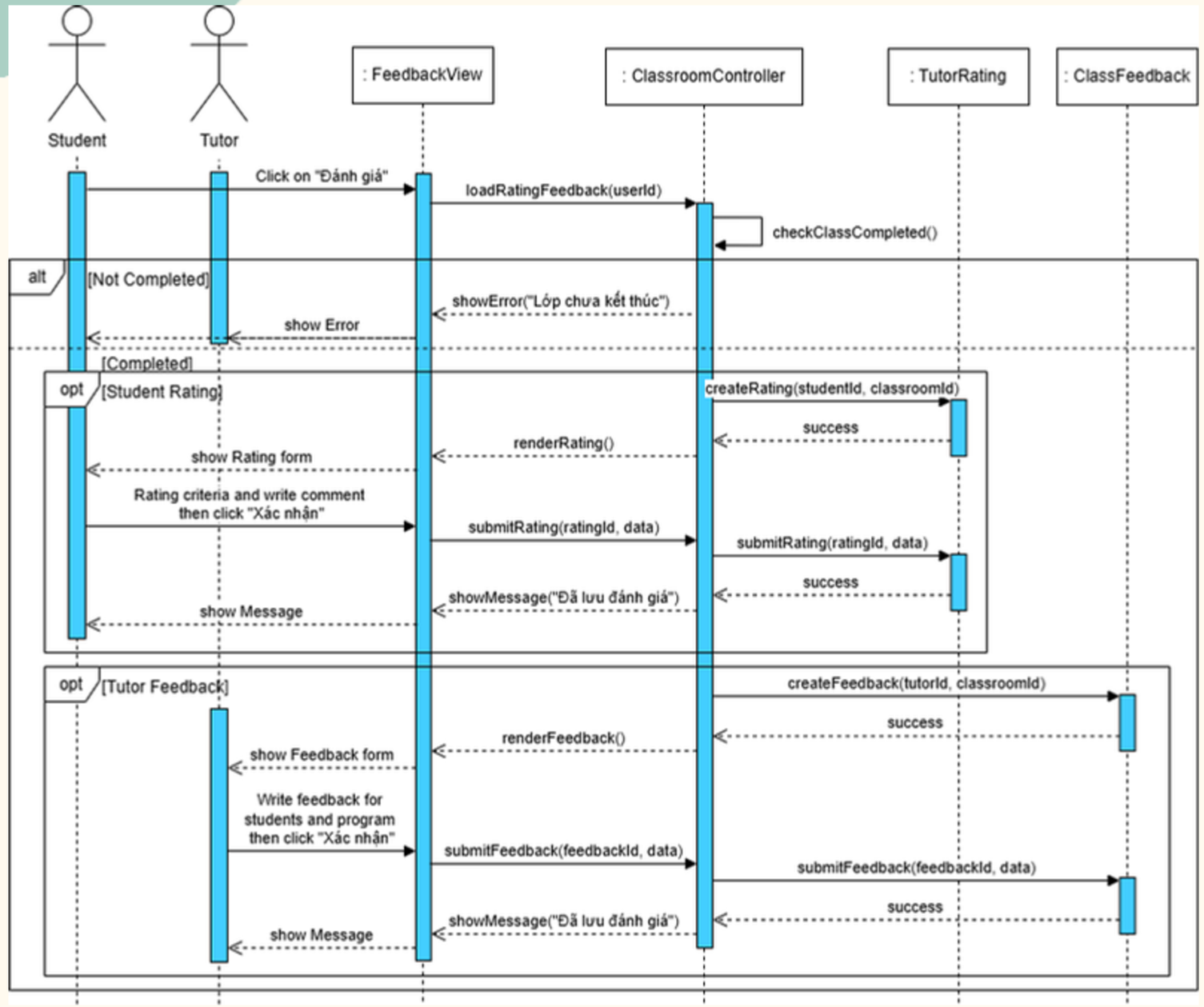
Grading submission sequence diagram



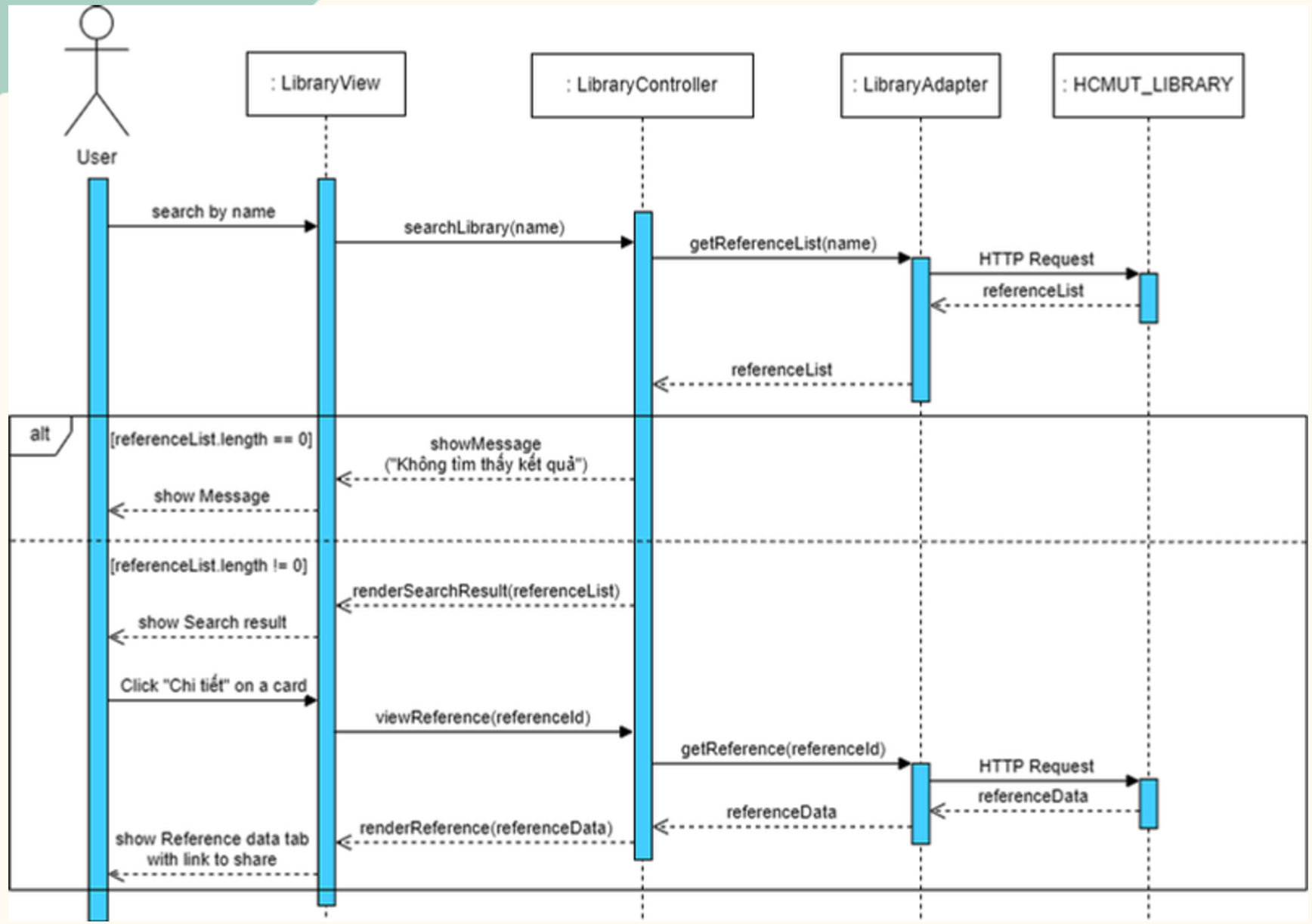
Session sequence diagram



Chat sequence diagram



Rating & Feedback sequence diagram



Search library sequence diagram

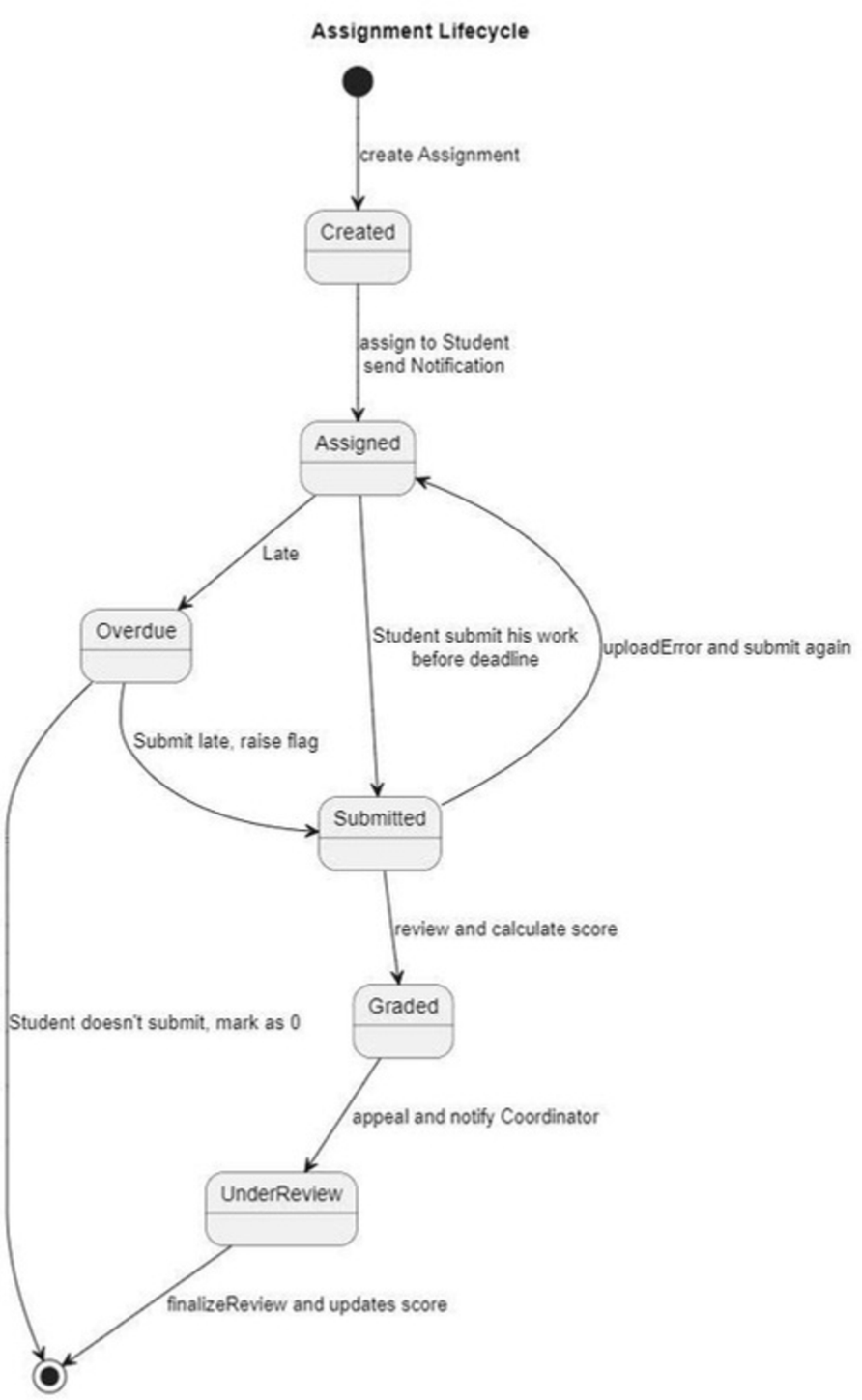
STATE CHART

ASSIGNMENT

- **Purpose**

  Models the full workflow of an assignment from creation to grading and review, ensuring clarity and traceability in the system.

- **Main States**
  - Created – Tutor creates assignment and sets metadata.
  - Assigned – Students receive the assignment with notifications.
  - Overdue – Deadline passed; late submissions are still allowed but flagged.
  - Submitted – Student submits work; upload errors allow resubmission.
  - Graded – Tutor evaluates and assigns a score; non-submission results in an automatic zero.
  - UnderReview – Students may appeal; coordinator/tutor revises scoring if needed.

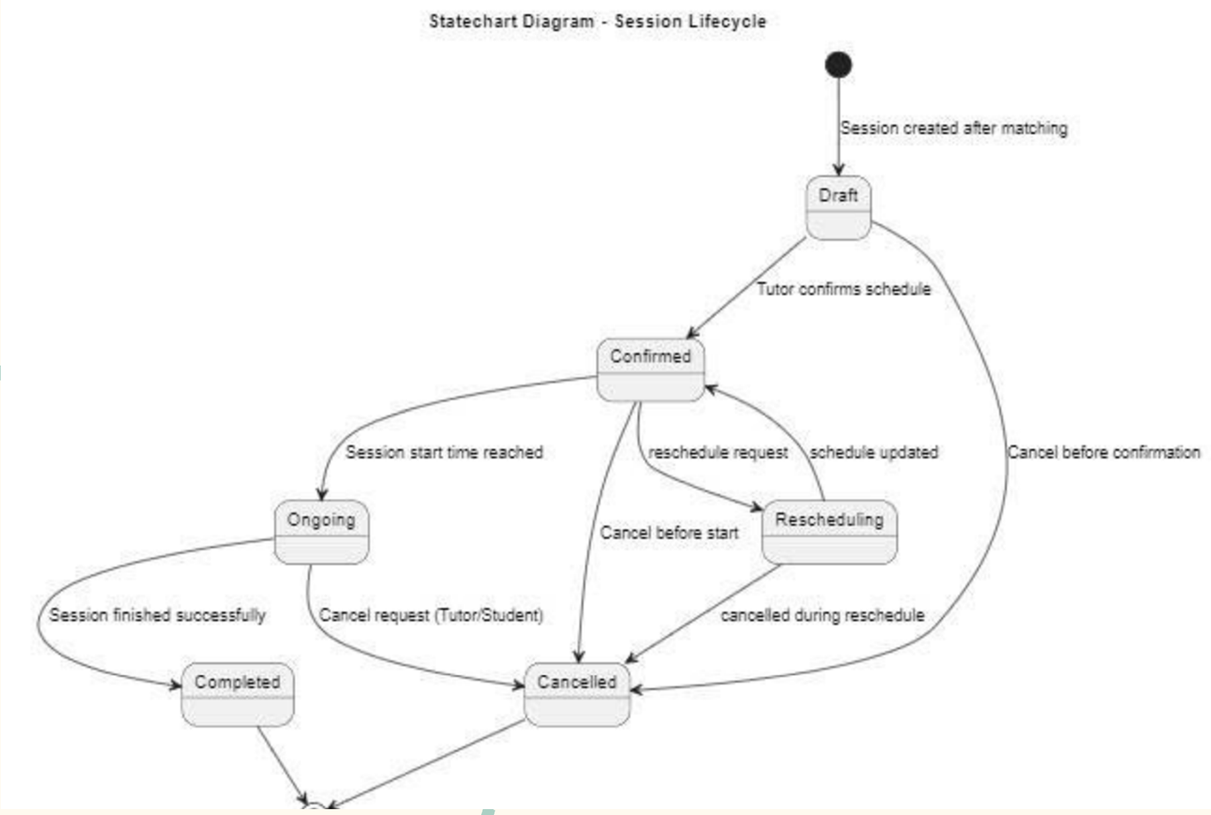


SESSION

- **Purpose**

Describes the full lifecycle of a tutoring session, from initial scheduling to completion or cancellation, supporting realistic tutor–student interactions.

- **Main States**
  - Draft: Session is created after matching. Tutor can edit schedule details.
  - Confirmed: Tutor confirms schedule. Both parties receive notifications.
  - Rescheduling: Triggered when tutor or student requests a schedule change.
  - Ongoing: Session automatically starts when scheduled time is reached.
  - Completed: Session finishes successfully. Attendance is recorded.
  - Cancelled: Session is cancelled at any point (before start, during reschedule, or during the session).

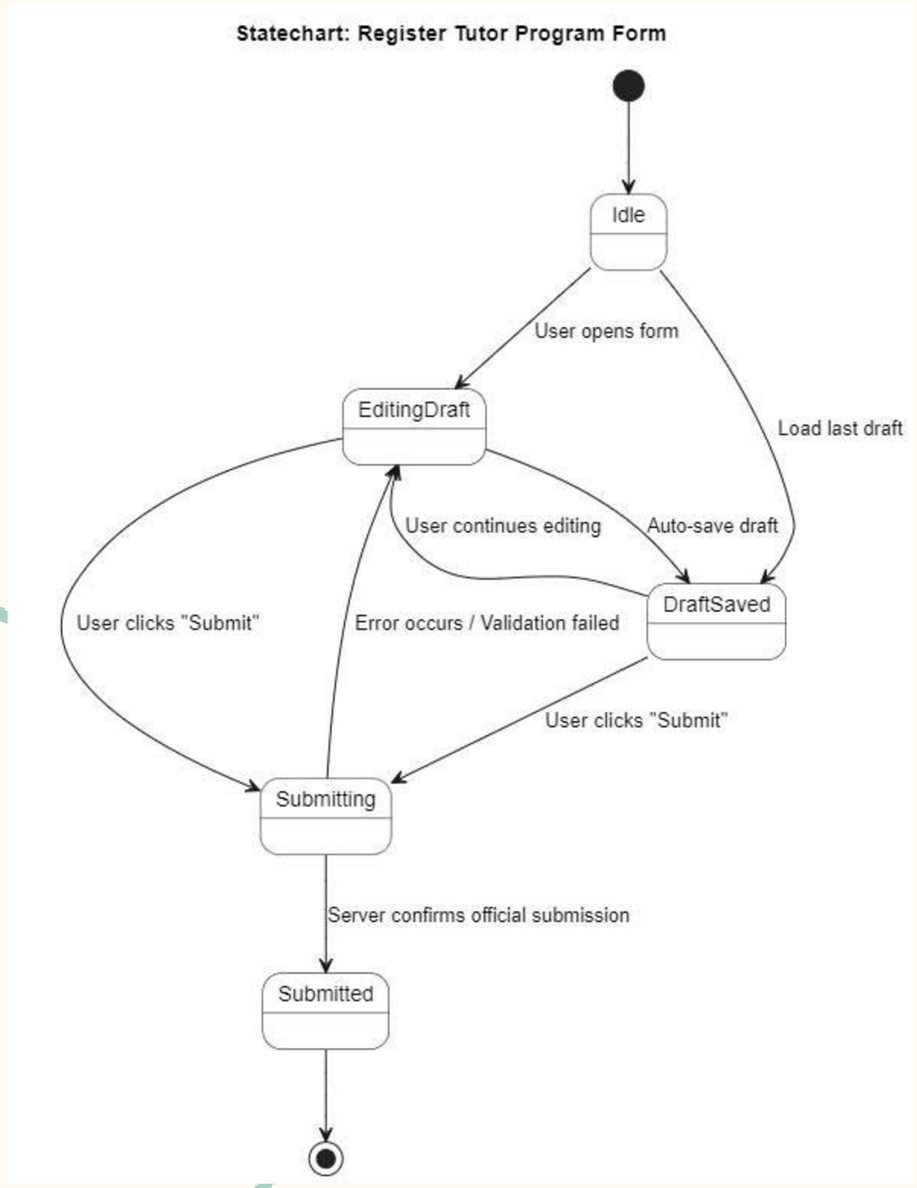


REGISTRATION

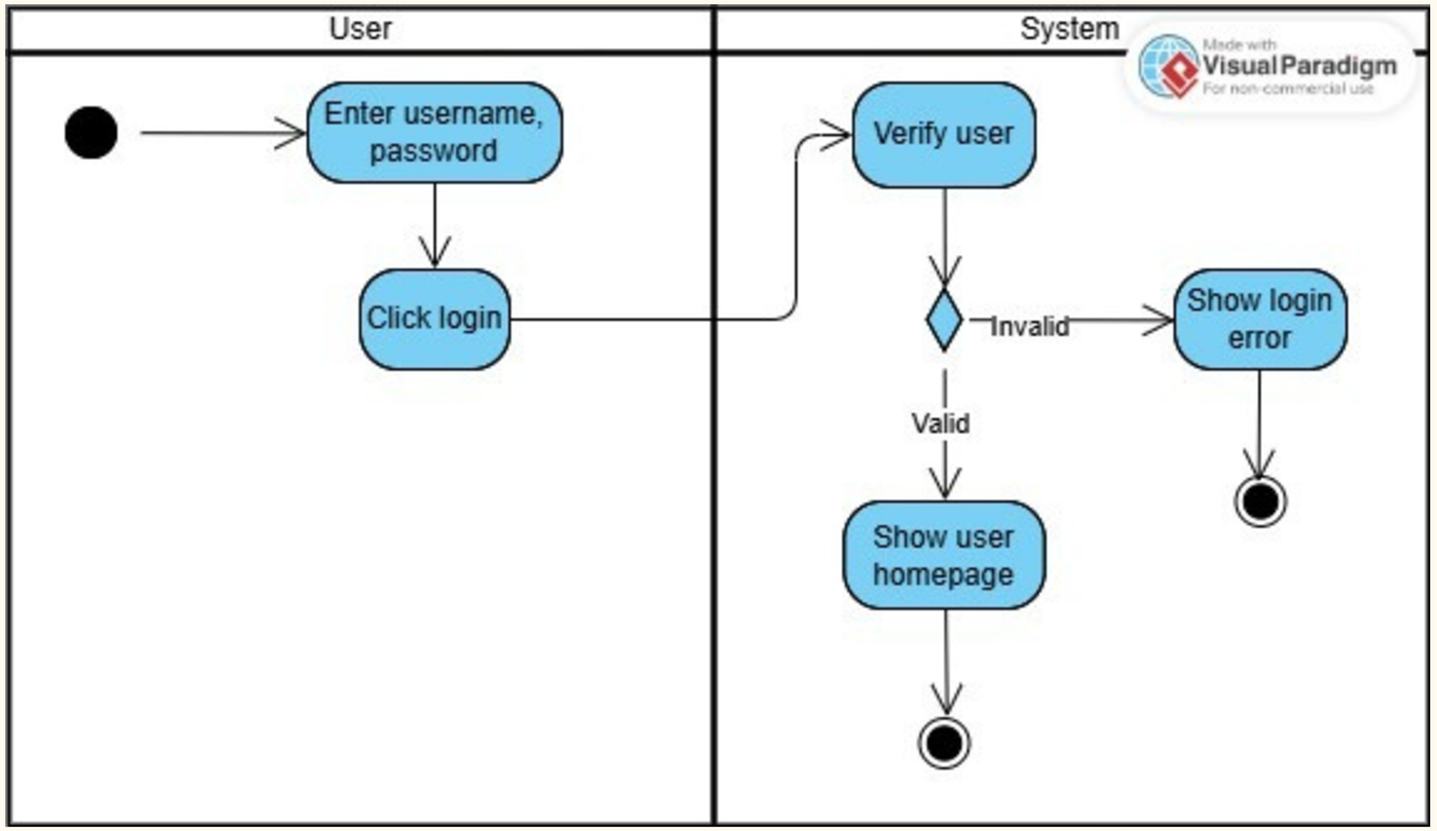
- **Purpose**

Models the user workflow of filling, saving, editing, and submitting the tutor registration form.Ensures users never lose progress and can recover drafts at any time.

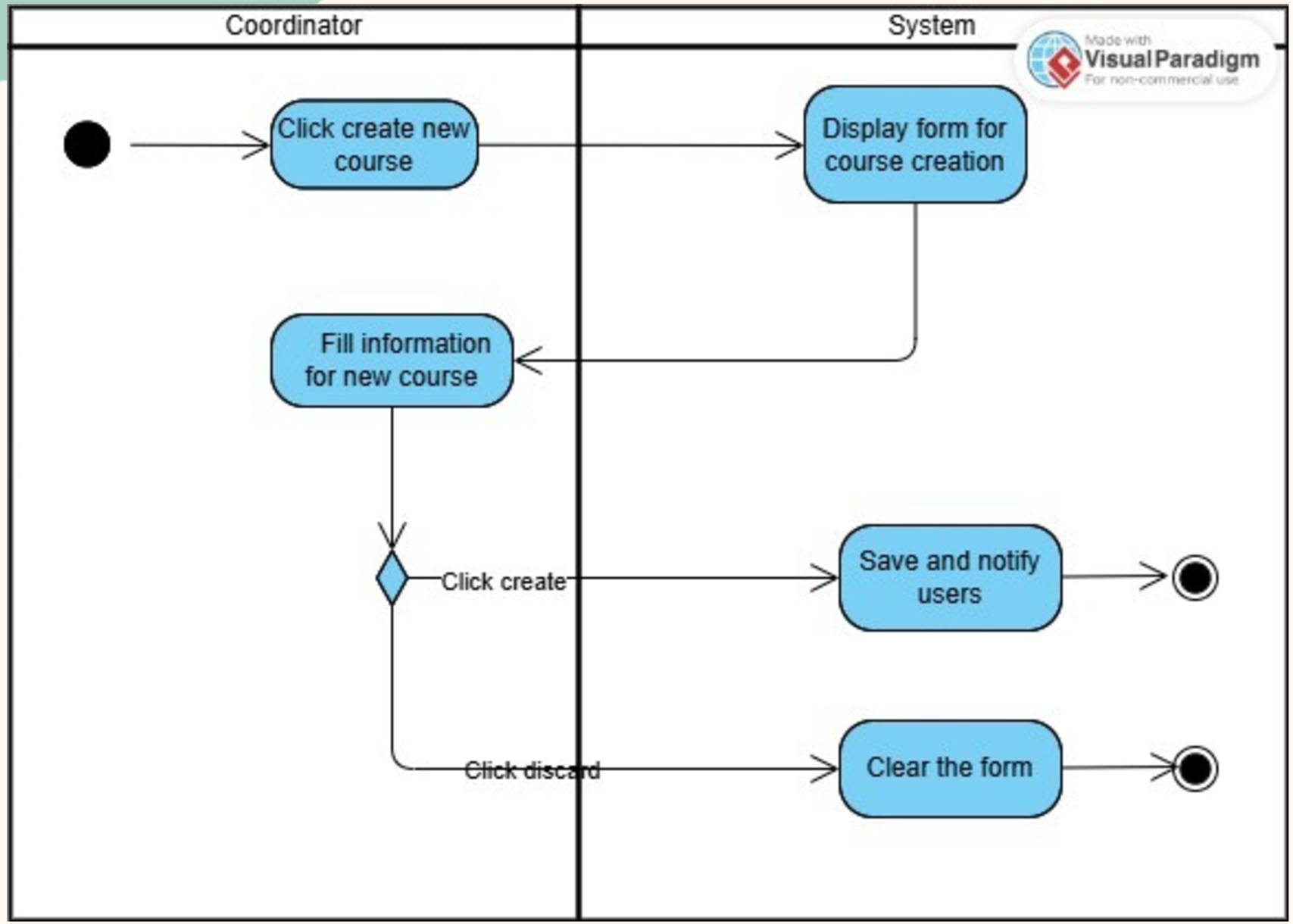
- **Main States**
  - Idle: User has not opened the form yet. May load the last saved draft automatically.
  - EditingDraft: User is actively editing the form. Auto-save may trigger at intervals, and user may continue editing freely.
  - DraftSaved: The form has been saved but not submitted.
  - User may resume editing or submit the form.
  - Submitting: The system validates the input and processes the submission. Errors or validation failures return the user to EditingDraft.
  - Submitted: Server confirms official submission. Form becomes read-only and final.



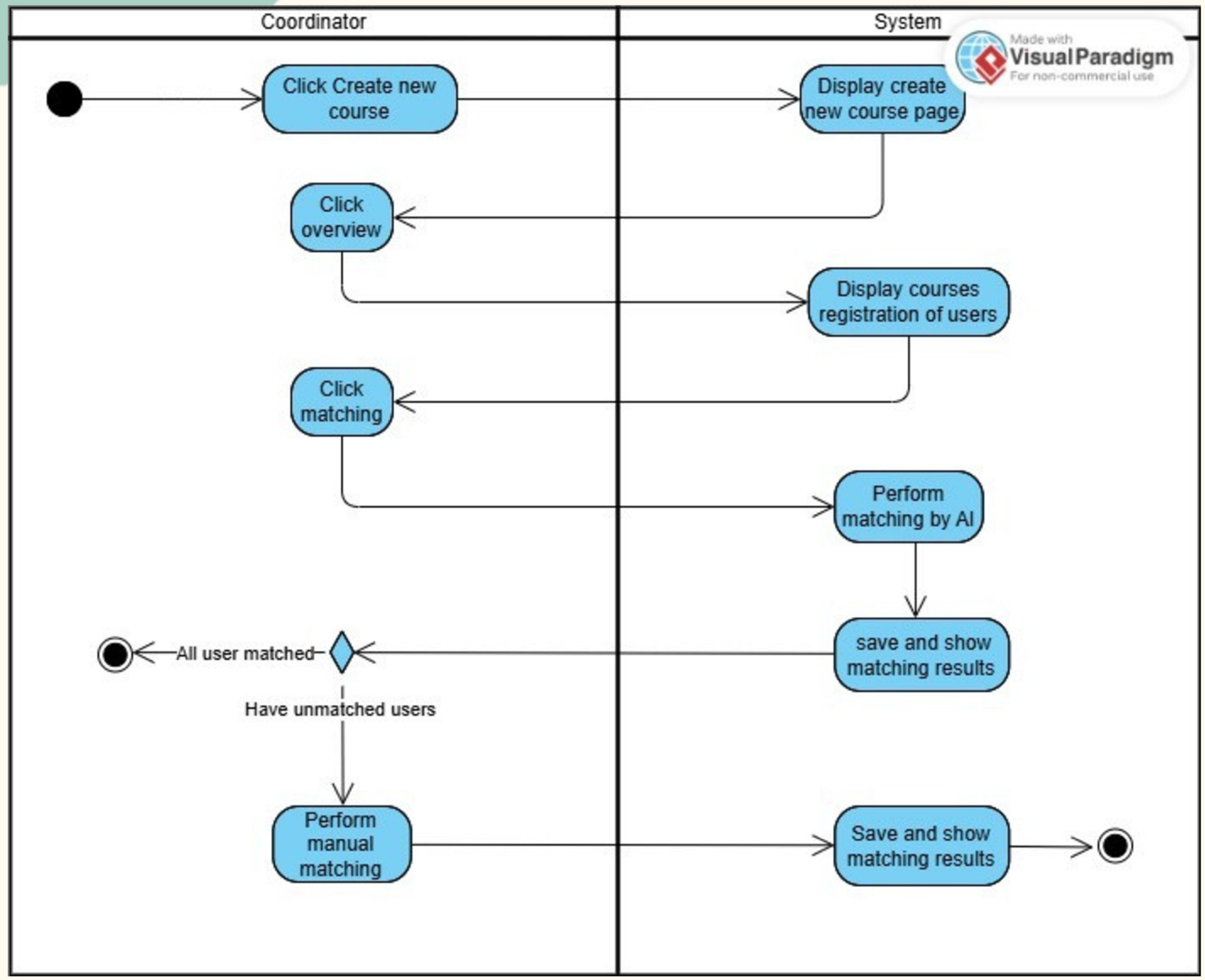
Activity diagram



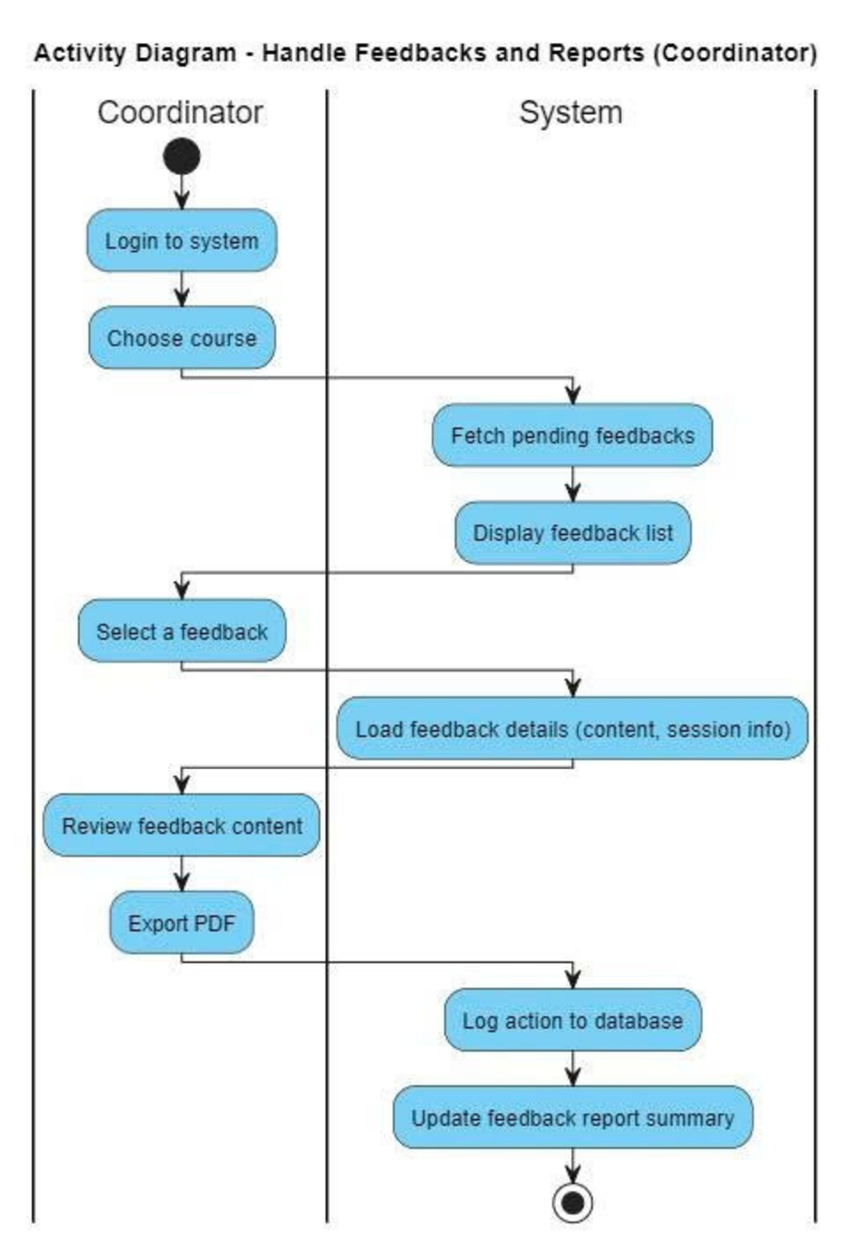
User - login process



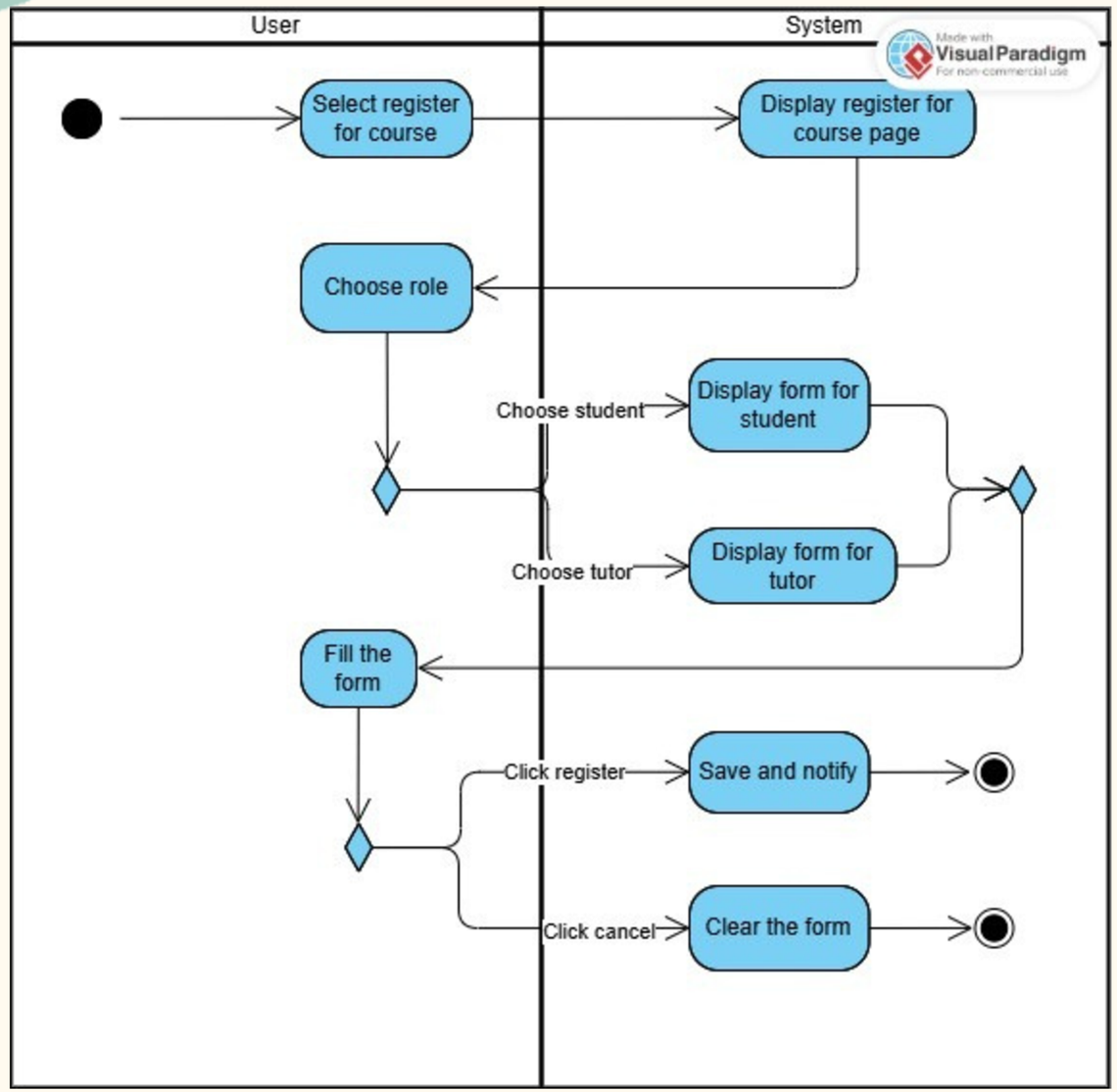
Coordinator - create new course



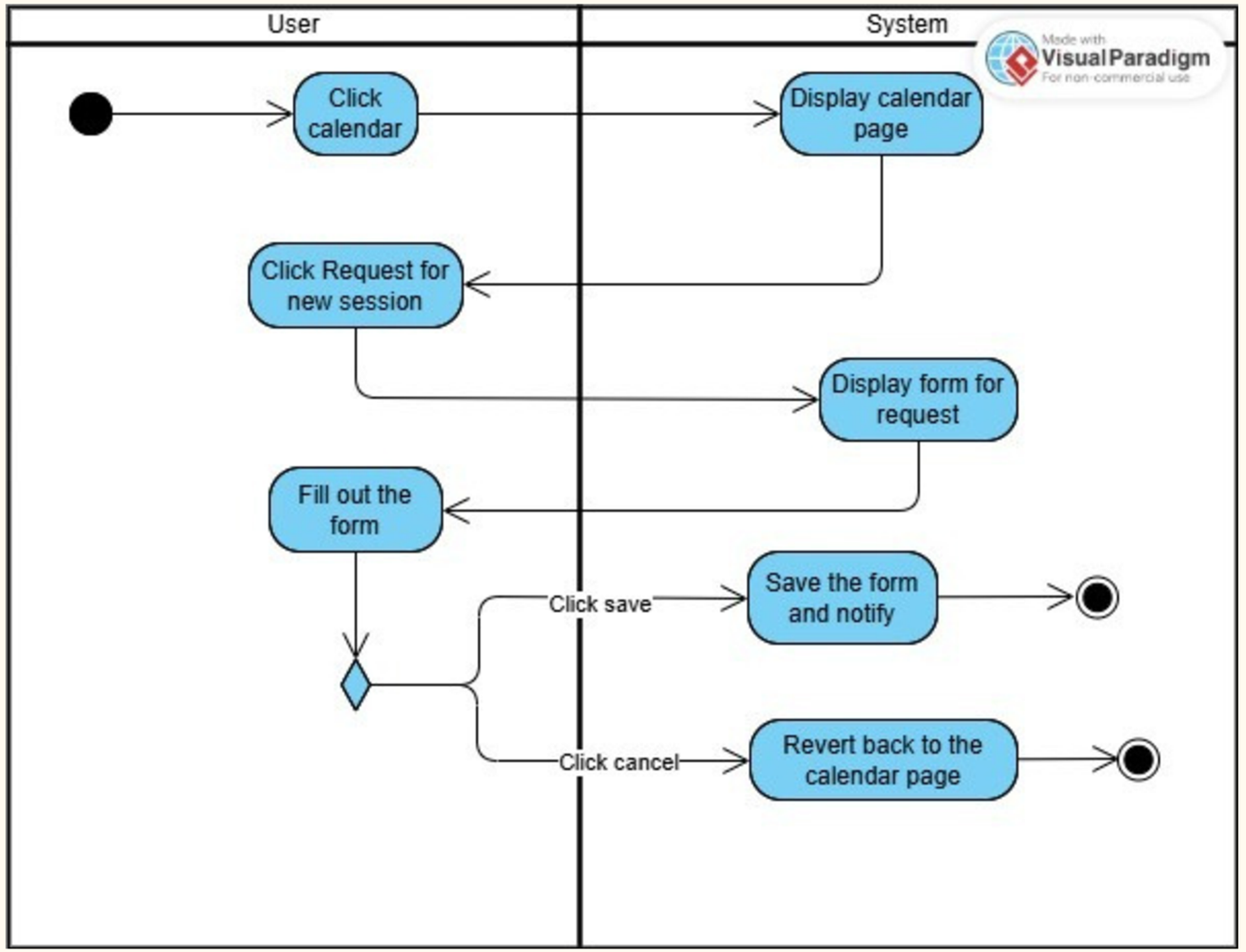
Coordinator - handle matching process



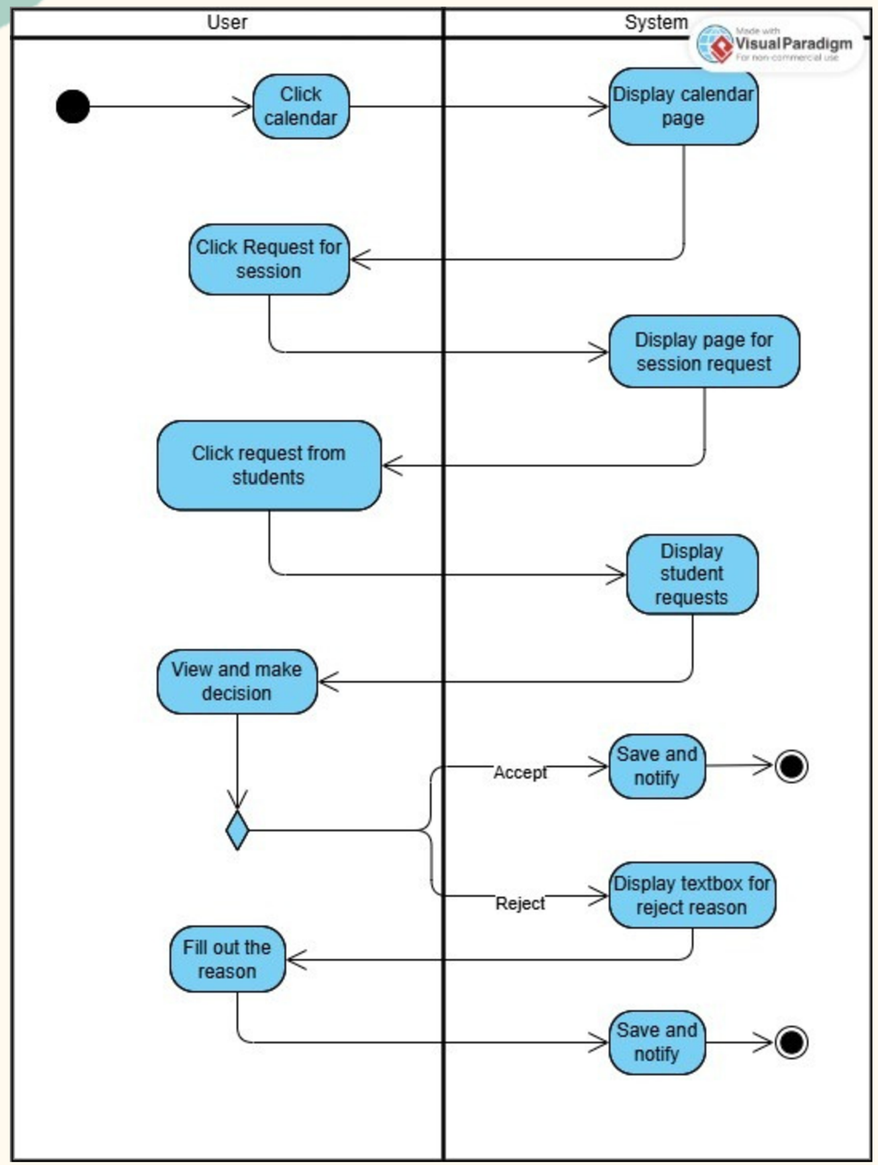
Coordinator - handle feedback and report



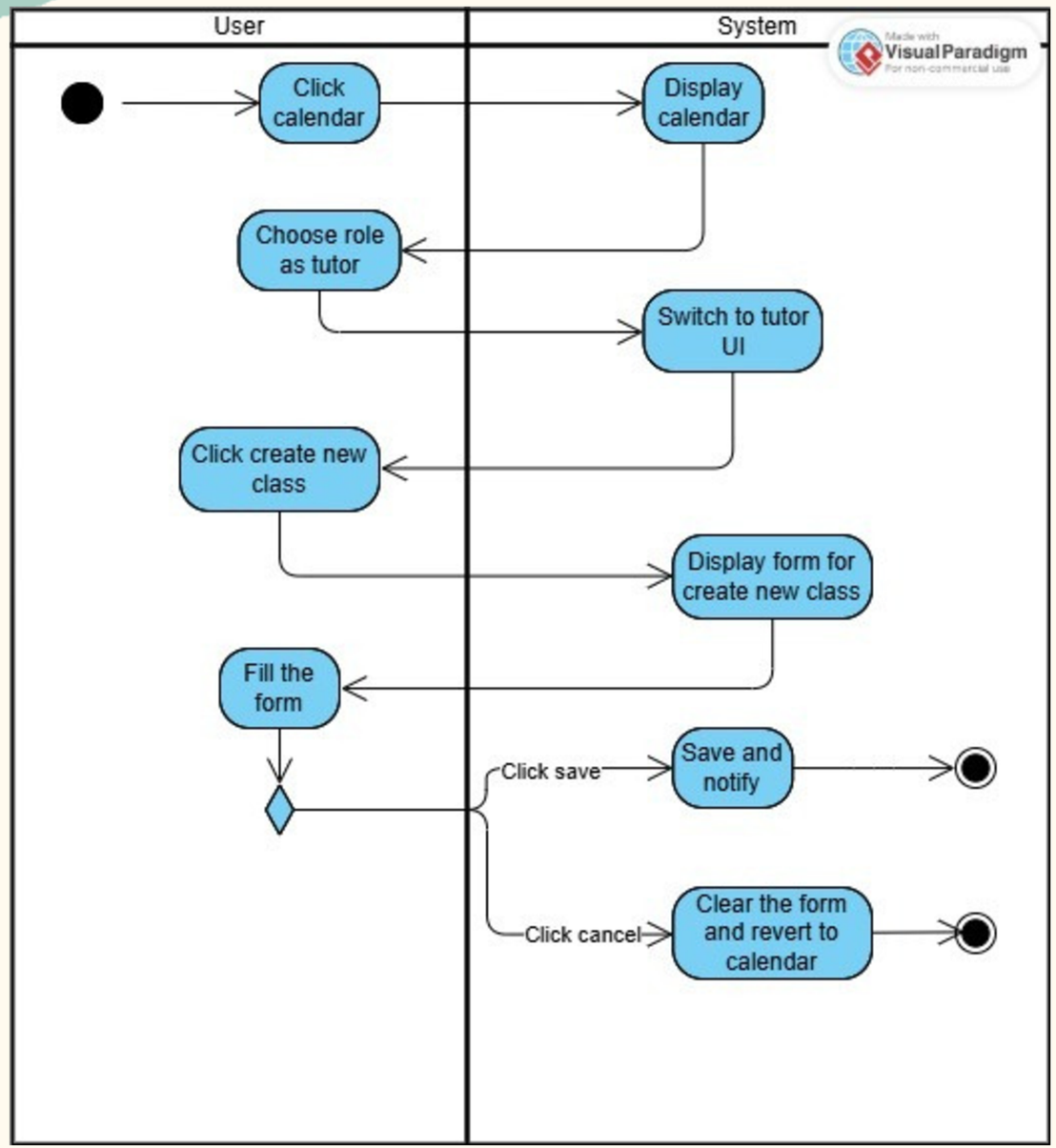
User - register for the tutor program



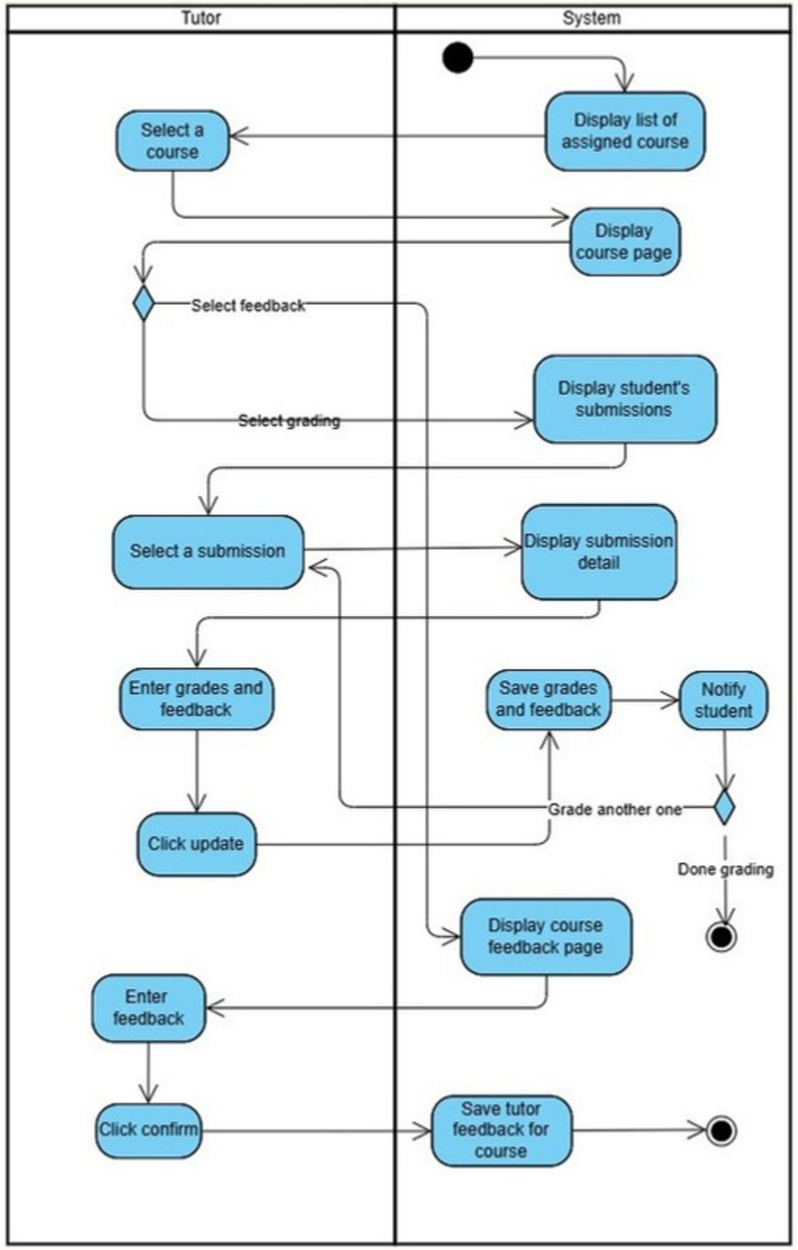
Students - make request for new session



Tutor - handle session requests from students



Tutor - create new session



Tutor - Grading and give feedback

Class diagram

MVC Architecture

Why We Chose the MVC Architecture ?

MVC divides the system into three components:

Model: Handles application data and business logic.

View: Represents the user interface and presentation layer.

Controller: Manages user inputs and interactions between the Model and View.

Impact on the Class Diagram

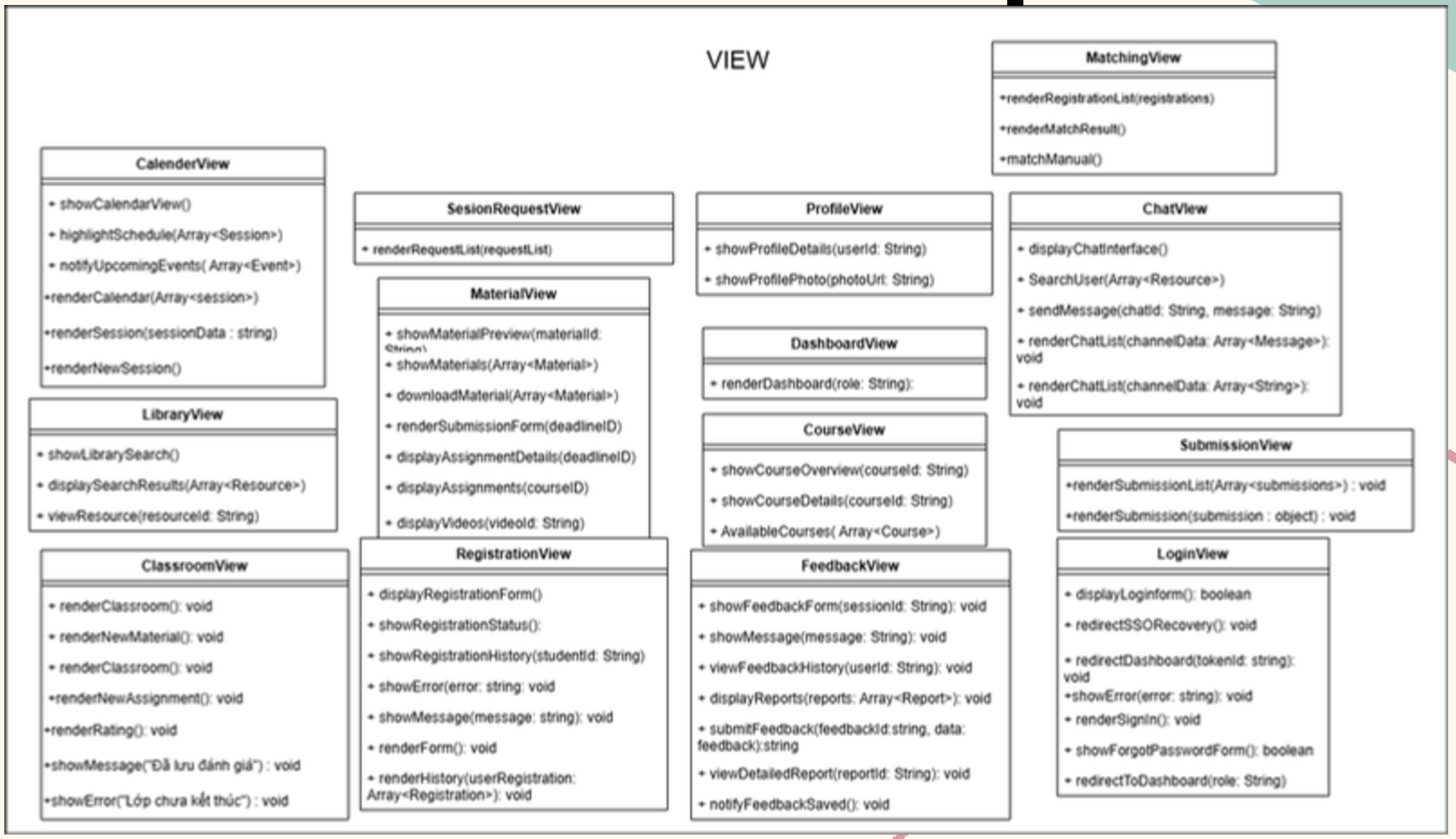
The class diagram was designed to reflect the MVC architecture and illustrate the relationships between the Model, View, and Controller components.

It provides a visual representation of how the system's components interact and share responsibilities.

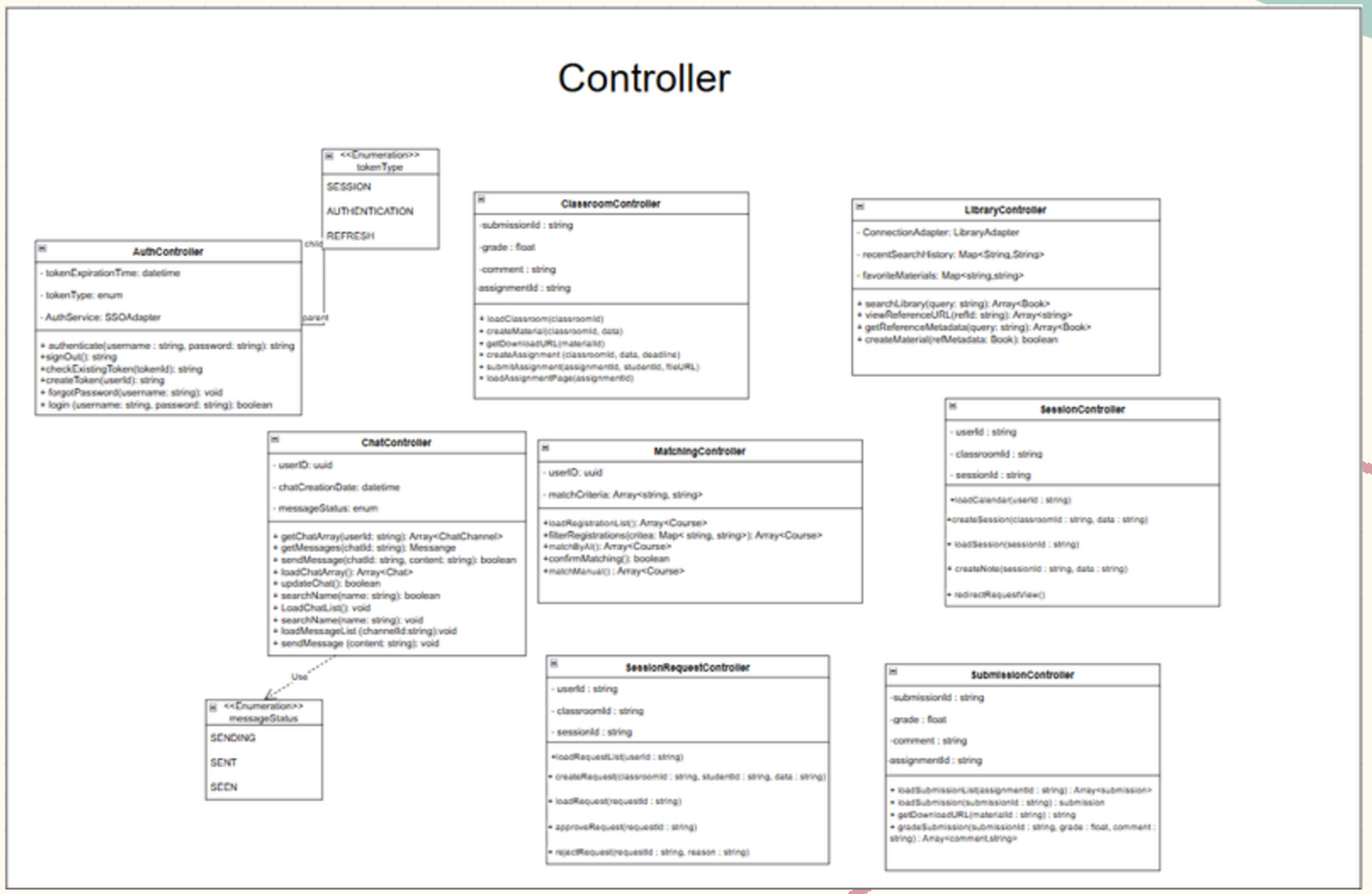
Model Group


AuthToken
- tokenId: string
- userId: string
- expiresAt: Date
+ createToken(userId: string): tokenId
+ getTokenByUser(userId: string): AuthToken
+ refreshToken(tokenId: string): void
+ revokeToken(userId: string): void
+ deleteToken(userId: string): void
+ checkExistingToken(tokenId): boolean
+ verifyToken(token: string): boolean
+ cleanupExpiredTokens(): void
Registration
- registrationId: string
- userId: string
- courseId: string
- role: "STUDENT" | "TUTOR"
- isDraft: boolean
- status: "PENDING" | "APPROVED" | "REJECTED"
- language: "VIETNAMESE" | "ENGLISH"
- method: "ONLINE" | "ON_OFFLINE"
- location?: string
- records?: string
- aim?: string
- specialRequirements: string
+ checkAllJsonForm(): boolean
+ getRegistrations(field: string = 'None', value: string = 'None'): Array<Registration>
+ submitRegistration(registrationId: string): void
+ deleteRegistration(registrationId: string): void
+ approveRegistration(registrationId: string): void
+ declineRegistration(registrationId: string): void
+ getDraft (userId: string): Registration
+ createDraft (userId: string): void
+ updateDraft (draftId: string, formData: Registration): void
+ deleteDraft (draftId: string): void
Course
- courseId: string
- courseName: string
- department: string
- description: string
+ createCourse(courseName: Course): void
+ fetchCoursesData(arg): Array<Course>
+ getCourse(field: string, value: string): void
+ upadteCourse(courseId: string, courseData: Course): void
+ deleteCourse(courseId: string): void
+ getStatistics(courseId: string): string
Notification
- notificationId: string
- senderId: string
- recipientId: string
- title: string
- content: string
- isRead: boolean
- sentAt: Date
+ createNotification(title: string, content: string, recipientId: Array<string>): void
+ getAllNotification(recipientId: string): Array<Notification>
+ findNotification(title: string) : Array<Notification>
+ deleteNotification(notificationId: string): void
+ sendNotification(userIds: Array<string>): void
+ markAsRead (notificationId: string): void
User
- userId: string
- fullName: string
- DoB: Date
- email: string
- pictureURL: string
- isStudent: boolean
- isTutor: boolean
- isCoordinator: boolean
- isAdmin: boolean
+ signin(username: string, password: string): AuthToken
+ getUserProfile(userId: string): User
+ addRole(userId string, role: string): void
+ removeRole(userId string, role: string): void
+ hasRole(role: string): boolean
+ findById(userId: string): User
+ findByName(name: string): Array<User>
+ getStatistic(userId: string): double
ChatChannel
- channelId:string
- channelName:string
- participants: Array<string>
+ getAllListMessages(userId: string): Map<string, Message>
+ getChannelMessages(channel: string): Array<Message>
+ create(data: CreateChatChannelDto): ChatChannel
+ findByIdChannel(id: string): ChatChannel
+ findByUserId(userId: string): Array<ChatChannel>
+ addParticipant(channelId: string, userId string): boolean
+ removeParticipant(channelId: string, userId: string): boolean
+ archive(channelId: string): ChatChannel
+ getMessages(channelId: string, limit number, before?: Date): Array<Message>
+ canEdit(userId: string): boolean
+ isParticipant(userId: string): boolean
getChannelList(userId:string): Map<string, Message>
Message
- messageId: string
- channelId: string
- senderId: string
- content: string
- sentAt: Date
- attachmentURL? string
+ create(Message):channelId: string, content: string, attachmentURL? string): void
+ getMessageList(channelId: string): Array<Message>
+ downloadAttachment(messageId: string): void
SessionRequest
- requestId: string
- sessionId: string
- studentId: string
- typeOfRequest: "NEW" | "ABSENT"
- status: "PENDING" | "APPROVE" | "REJECT"
- title: string
- reason? string
+ Create(sessionId: string, studentId: string, title: string, typeOfRequest: string, reason: string): void
+ filterSessionRequest(field: string, value: string): Array<sessionRequest>
+ findPending(): Array<SessionRequest>
+ approveRequest(requestId: string): void
+ rejectRequest(requestId: string, reason: string): void
Session
- sessionId: string
- classroomId: string
- title: string
- description: string
- startTime: Date Time
- endTime: Date Time
- sessionType: "ONLINE" | "OFFLINE"
- linkURL? string
- location? string
- cancelReason? string
+ createSession(title: string, description: string, startTime: Date, endTime: Date, sessionType: "ONLINE" | "OFFLINE", linkURL? string, location? string): void
+ getSessionsByUser(userId string): Array<Session>
+ getSessionsByClass(classroomId string): Array<Session>
+ updateSession(sessionId: string, updatedSession: Session): void
+ cancelSession(sessionId: string, reason: string): void
Classroom
- classroomId: string
- courseId: string
- language: "VIETNAMESE" | "ENGLISH"
- tutorId string
- studentIds: Array<string>
- status: "OPENED" | "CLOSED"
+ createClasses(classrooms: Array<Classroom>): void
+ createClass(classroom: Classroom): void
+ getClassesByUser(userId string): Array<Classroom>
+ getClassByCourse(courseId string): Array<Classroom>
+ updateClass(classroomId string, updatedClassroom: Classroom): void
+ deleteClass(classroomId string): void
+ closeClass(classroomId string): void
+ enrolTutor(classroomId string, tutorId string): void
+ removeTutor(classroomId string, tutorId string): void
+ enrolStudent(classroomId string, studentId string): boolean
+ removeStudent(classroomId string, studentId string): boolean
+ getClassroom(classRoomId string): Classroom
Assignment
- assignmentId string
- classroomId string
- title string
- description? string
- deadline Date
+ getAllAssignmentByUser(userId string): Array<Assignment>
+ getAllAssignmentByClass(classroomId string): Array<Assignment>
+ createAssignment(classroomId string, title string, description? string, deadline Date): assignmentId string
+ updateAssignment(assignmentId string, updatedAss: Assignment): void
+ deleteAssignment(assignmentId): void
+ checkOverdue(submittedAt Date): boolean
Submission
- submissionId string
- assignmentId string
- studentId string
- fileURL? string
- submittedAt? Date
- grade? double
- tutorComment? string
- status: "SUBMITTED" | "NOT_SUBMITTED" | "GRADED"
+ createSubmission(assignmentId string, studentId string, fileURL string): Date
+ getSubmission(userId string): void
+ getAllSubmission(assignmentId string): void
+ updateSubmission(assignmentId string, studentId string, fileURL string): submissionId
+ deleteSubmission(submissionId string): void
+ grade(submissionId string, grade double, tutorComment? string): void
+ downloadSubmission(submissionId string): void
SessionNote
- noteId: string
- sessionId: string
- attendace: Map<string, boolean>
- meetingMinute string
+ createNote(sessionId: string, Attendance: Map<string, boolean>, meetingMinute string): void
+ getNote(sessionId string): SessionNote
+ updateNote(noteId: string, updatedNote SessionNote): void
+ deleteNote(noteId: string): void
TutorRating
- ratingId string
- studentId string
- classroomId string
- criteriaRating Map<string, int>
- comment string
+ createRating(studentId string, classroomId string, criteriaRating Map<string, int>, comment string): void
+ findByTutorId(tutorId string): Array<TutorRating>
+ findByClassroomId(classroomId string): Array<TutorRating>
+ updateRating(ratingId string, updatedRating TutorRating): void
+ deleteRating(ratingId string): void
- studentId string
Material
- materialId string
- classroomId string
- title string
- description? string
- type ClassMaterial
- fileURL? string
+ createMaterial(classroomId string, title string, description? string, type ClassMaterial, fileURL? string): string
+ getAllMaterial(classroomId string): Array<Material>
+ updateMaterial(materialId string, updatedMaterial Material): void
+ deleteMaterial (materialId string): void
+ downloadMaterial (materialId string): void
+ getMaterialByClass(classId string): Array<Material>
<<Enumeration>>
ClassMaterial
- FILE: int = 0
- MOVE: int = 1
- REFERENCE: int = 2
- NOTE: int = 3
ClassFeedback
- feedbackId string
- classroomId string
- tutorId string
- studentFeedacks: Map<string, string>
- comment string
+ createFeedback(tutorId string, studentFeedacks: Map<string, string>, comment string): void
+ createFeedback(tutorId string, studentFeedacks: Map<string, string>, comment string): void
+ submitFeedback(ratingId string, data Map<string, string>): void
+ findByClassroomId(classroomId string): Array<ClassFeedback>
+ findByTutorId(tutorId string): Array<ClassFeedback>
+ updateFeedback(feedbackId string, updatedFeedback ClassFeedback): void

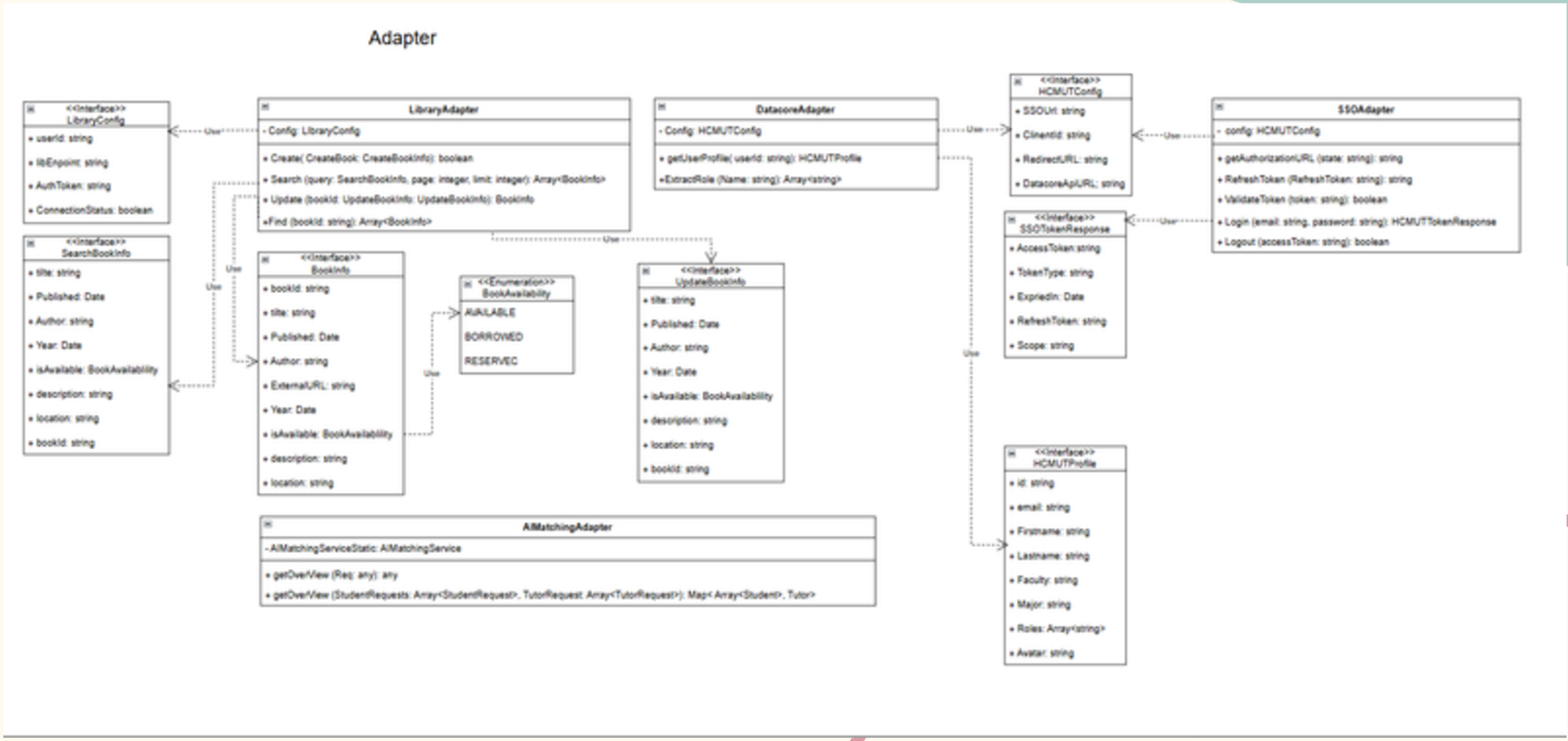
View Group


VIEW
MatchingView
+renderRegistrationList(registrations)
+renderMatchResult()
+matchManual()
CalenderView
+ showCalendarView()
+ highlightSchedule(Array<Session>)
+ notifyUpcomingEvents(Array<Event>)
+renderCalendar(Array<session>)
+renderSession(sessionData : string)
+renderNewSession()
SesionRequestView
+ renderRequestList(requestList)
MaterialView
+ showMaterialPreview(materialId: String)
+ showMaterials(Array<Material>)
+ downloadMaterial(Array<Material>)
+ renderSubmissionForm(deadlineID)
+ displayAssignmentDetails(deadlineID)
+ displayAssignments(courseID)
+ displayVideos(videoId: String)
ProfileView
+ showProfileDetails(userId: String)
+ showProfilePhoto(photoUrl: String)
DashboardView
+ renderDashboard(role: String):
ChatView
+ displayChatInterface()
+ SearchUser(Array<Resource>)
+ sendMessage(chatId: String, message: String)
+ renderChatList(channelData: Array<Message>): void
+ renderChatList(channelData: Array<String>): void
LibraryView
+ showLibrarySearch()
+ displaySearchResults(Array<Resource>)
+ viewResource(resourceId: String)
CourseView
+ showCourseOverview(courseId: String)
+ showCourseDetails(courseId: String)
+ AvailableCourses(Array<Course>)
SubmissionView
+renderSubmissionList(Array<submissions>) : void
+renderSubmission(submission : object) : void
ClassroomView
+ renderClassroom(): void
+ renderNewMaterial(): void
+ renderClassroom(): void
+renderNewAssignment(): void
+renderRating(): void
+showMessage("Đã lưu đánh giá") : void
+showError("Lớp chưa kết thúc") : void
RegistrationView
+ displayRegistrationForm()
+ showRegistrationStatus():
+ showRegistrationHistory(studentId: String)
+ showError(error: string: void
+ showMessage(message: string): void
+ renderForm(): void
+ renderHistory(userRegistration: Array<Registration>): void
FeedbackView
+ showFeedbackForm(sessionId: String): void
+ showMessage(message: String): void
+ viewFeedbackHistory(userId: String): void
+ displayReports(reports: Array<Report>): void
+ submitFeedback(feedbackId:string, data: feedback):string
+ viewDetailedReport(reportId: String): void
+ notifyFeedbackSaved(): void
LoginView
+ displayLoginform(): boolean
+ redirectSSORecovery(): void
+ redirectDashboard(tokenId: string): void
+showError(error: string): void
+ renderSignIn(): void
+ showForgotPasswordForm(): boolean
+ redirectToDashboard(role: String)

Controller Group



Adapter


Adapter
<<Interface>>
LibraryConfig
+ userId: string
+ libEnpoint: string
+ AuthToken: string
+ ConnectionStatus: boolean
LibraryAdapter
- Config: LibraryConfig
+ Create(CreaneBook: CreaneBookInfo): boolean
+ Search (query: SearchBookInfo, page: integer, limit: integer): Array<BookInfo>
+ Update (bookId: UpdateBookInfo: UpdateBookInfo): BookInfo
+Find (bookId: string): Array<BookInfo>
DatacoreAdapter
- Config: HCMUTConfig
+ getUserProfile(userId: string): HCMUTProfile
+ExtractRole (Name: string): Array<string>
<<Interface>>
HCMUTConfig
+ SSOUrl: string
+ ClientId: string
+ RedirectURL: string
+ DatacoreApiURL: string
SSOAdapter
- config: HCMUTConfig
+ getAuthorizationURL (state: string): string
+ RefreshToken (RefreshToken: string): string
+ ValidateToken (token: string): boolean
+ Login (email: string, password: string): HCMUTTokenResponse
+ Logout (accessToken: string): boolean
<<Interface>>
SSOTokenResponse
+ AccessToken:string
+ TokenType: string
+ ExpriedIn: Date
+ RefreshToken: string
+ Scope: string
<<Interface>>
SearchBookInfo
+ title: string
+ Published: Date
+ Author: string
+ Year: Date
+ isAvailable: BookAvailability
+ description: string
+ location: string
+ bookId: string
<<Interface>>
BookInfo
+ bookId: string
+ title: string
+ Published: Date
+ Author: string
+ ExternalURL: string
+ Year: Date
+ isAvailable: BookAvailability
+ description: string
+ location: string
<<Enumeration>>
BookAvailability
AVAILABLE
BORROWED
RESERVEC
<<Interface>>
UpdateBookInfo
+ title: string
+ Published: Date
+ Author: string
+ Year: Date
+ isAvailable: BookAvailability
+ description: string
+ location: string
+ bookId: string
<<Interface>>
HCMUTProfile
+ id: string
+ email: string
+ Firstname: string
+ Lastname: string
+ Faculty: string
+ Major: string
+ Roles: Array<string>
+ Avatar: string
AIMatchingAdapter
-AIMatchingServiceStatic: AIMatchingService
+ getOverView (Req: any): any
+ getOverView (StudentRequests: Array<StudentRequest>, TutorRequest: Array<TutorRequest>): Map< Array<Student>, Tutor>
Use

Login - MVC

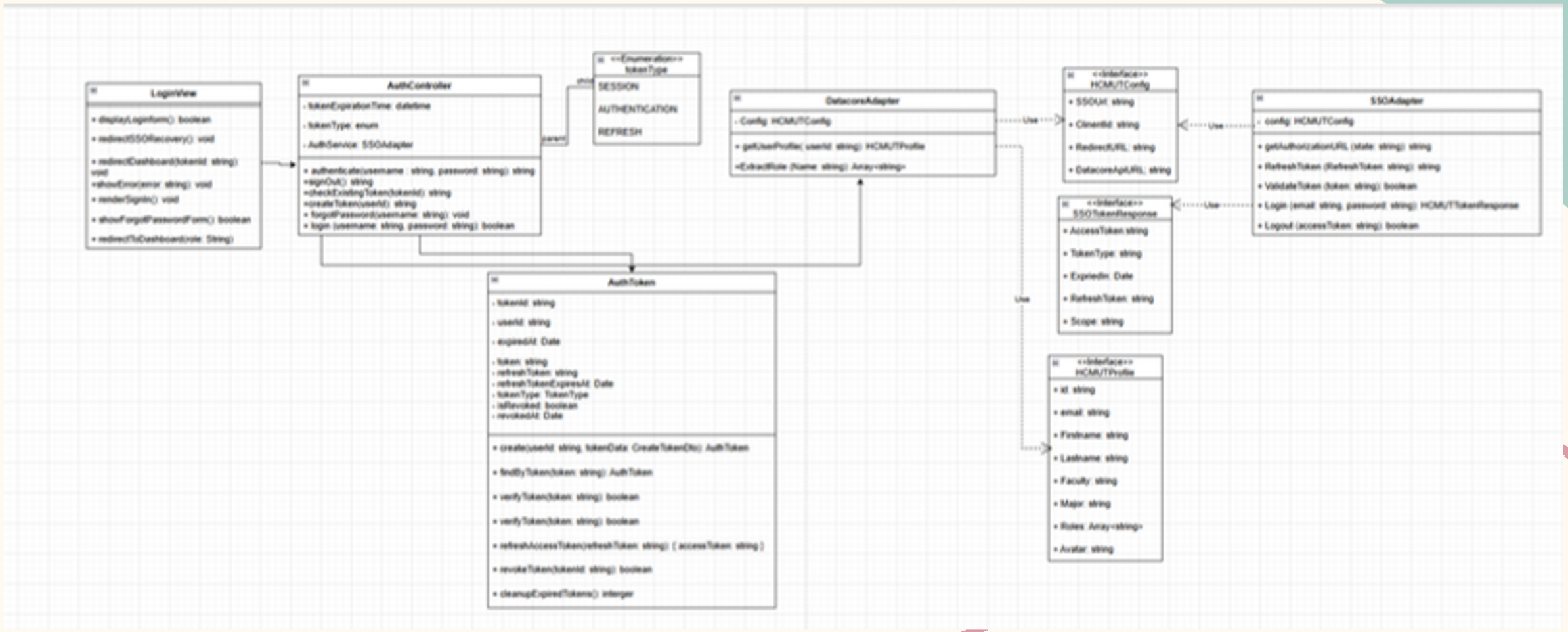
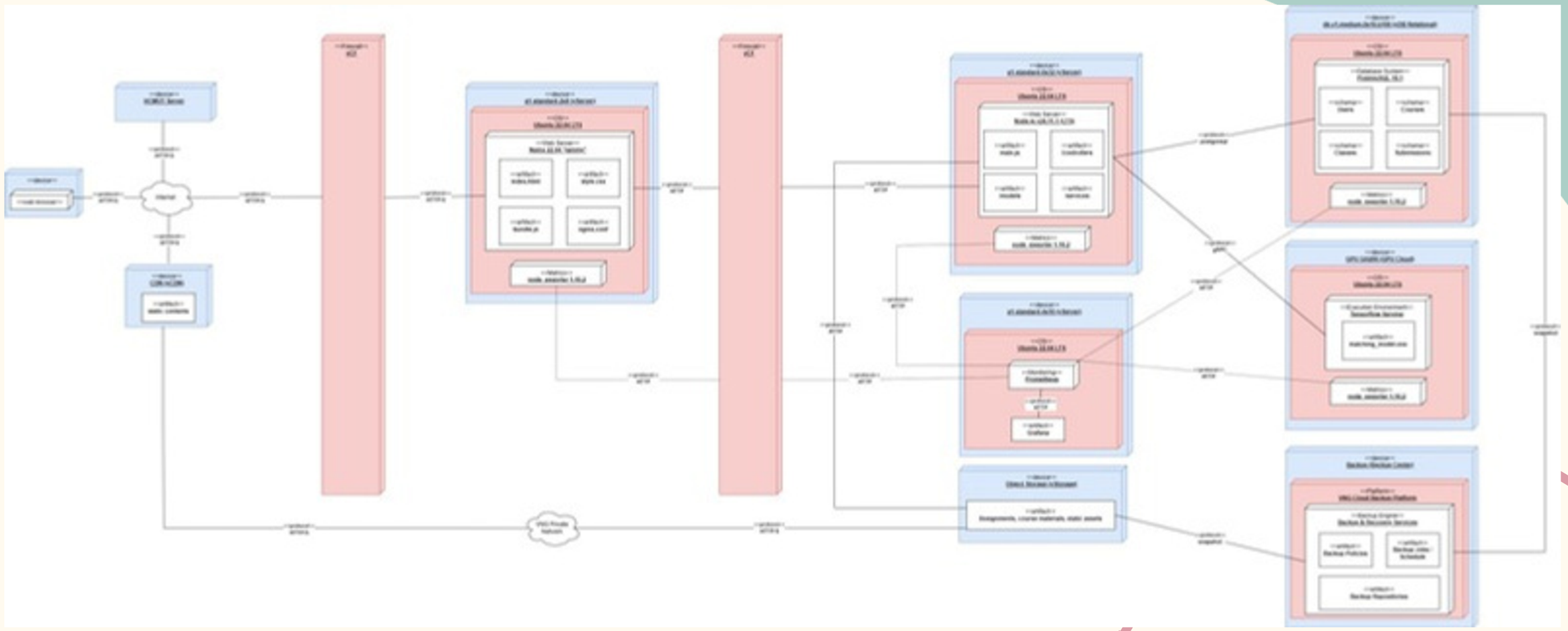

LoginView
+ displayLoginform(): boolean
+ redirectSSORecovery(): void
+ redirectDashboard(tokenId: string): void
+showError(error: string): void
+ renderSignin(): void
+ showForgotPasswordForm(): boolean
+ redirectToDashboard(role: String)
AuthController
- tokenExpirationTime: datetime
- tokenType: enum
- AuthService: SSOAdapter
+ authenticate(username : string, password: string): string
+signOut(): string
+checkExistingToken(tokenId): string
+createToken(userId): string
+ forgotPassword(username: string): void
+ login (username: string, password: string): boolean
<<Enumeration>>
tokenType
SESSION
AUTHENTICATION
REFRESH
create
parent
DatacoreAdapter
- Config: HCMUTConfig
+ getUserProfile(userId: string): HCMUTProfile
+ExtractRole (Name: string): Array<string>
AuthToken
- tokenId: string
- userId: string
- expiresAt: Date
- token: string
- refreshToken: string
- refreshTokenExpiresAt: Date
- tokenType: TokenType
- isRevoked: boolean
- revokedAt: Date
+ create(userId: string, tokenData: CreateTokenDto): AuthToken
+ findByToken(token: string): AuthToken
+ verifyToken(token: string): boolean
+ verifyToken(token: string): boolean
+ refreshAccessToken(refreshToken: string): { accessToken: string }
+ revokeToken(tokenId: string): boolean
+ cleanupExpiredTokens(): interger
<<Interface>>
HCMUTConfig
+ SSOUrl: string
+ ClinentId: string
+ RedirectURL: string
+ DatacoreApiURL: string
<<Interface>>
SSOTokenResponse
+ AccessToken string
+ TokenType: string
+ ExpiresIn: Date
+ RefreshToken: string
+ Scope: string
<<Interface>>
HCMUTProfile
+ id: string
+ email: string
+ Firstname: string
+ Lastname: string
+ Faculty: string
+ Major: string
+ Roles: Array<string>
+ Avatar: string
SSOAdapter
- config: HCMUTConfig
+ getAuthorizationURL (state: string): string
+ RefreshToken (RefreshToken: string): string
+ ValidateToken (token: string): boolean
+ Login (email: string, password: string): HCMUTTokenResponse
+ Logout (accessToken: string): boolean
Use
Use
Use
Use
Use

Table of contents

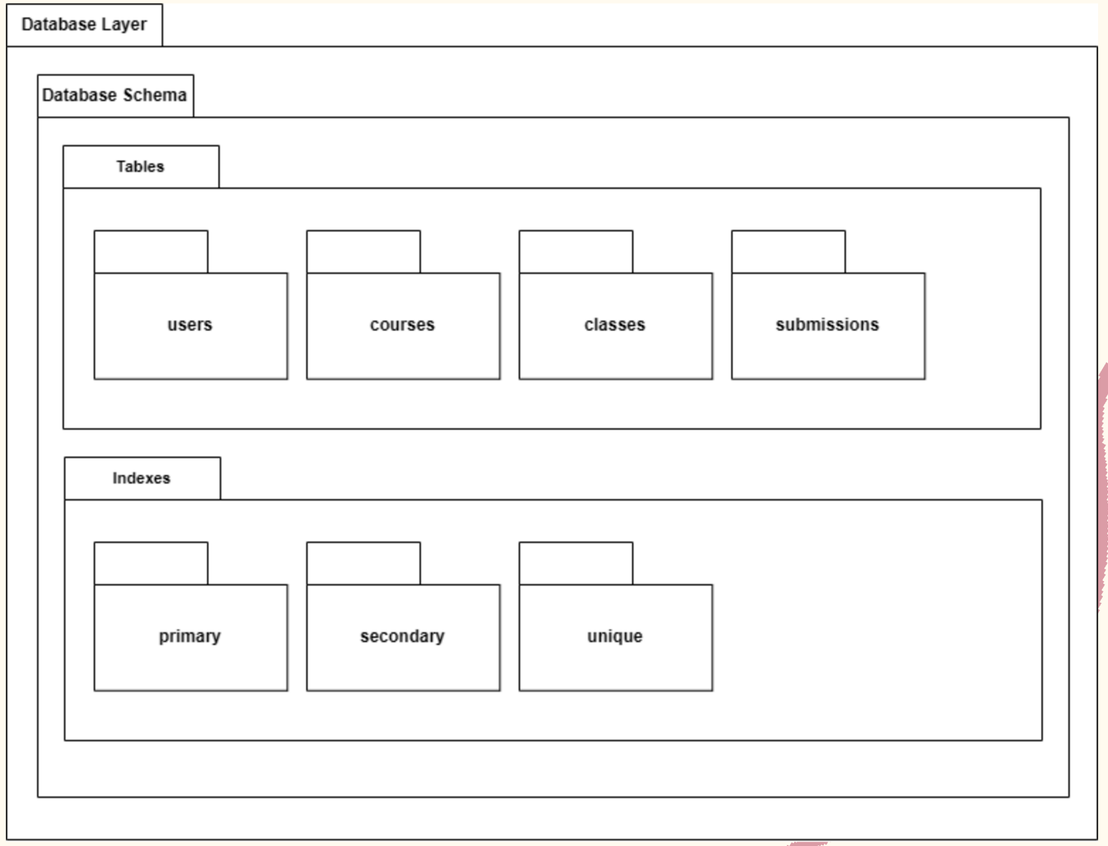
Deployment view

Deployment view

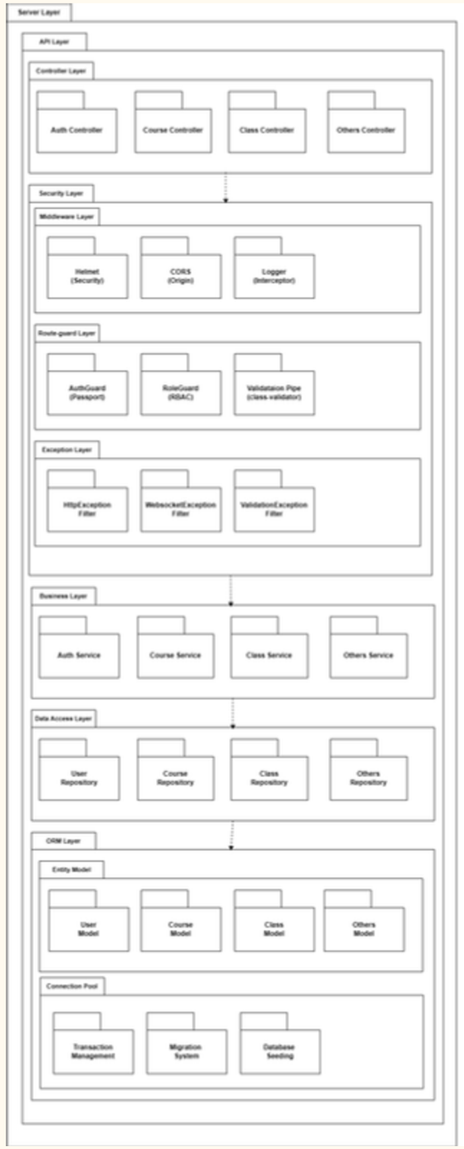


Implementation view

Implementation view- database layer



Implementation view



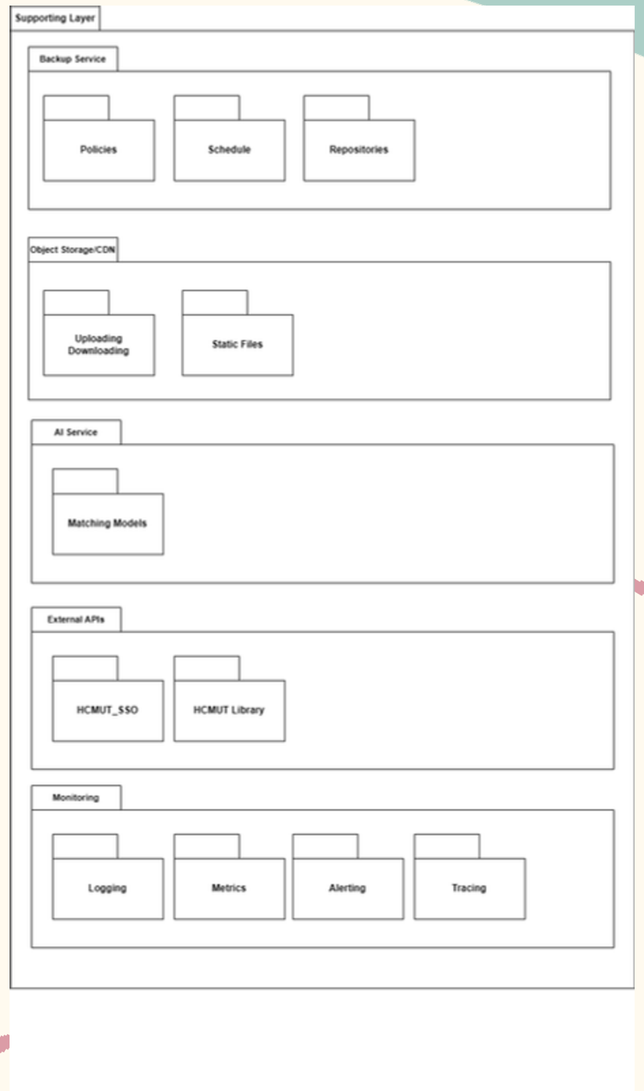


Table of contents

AI matching module

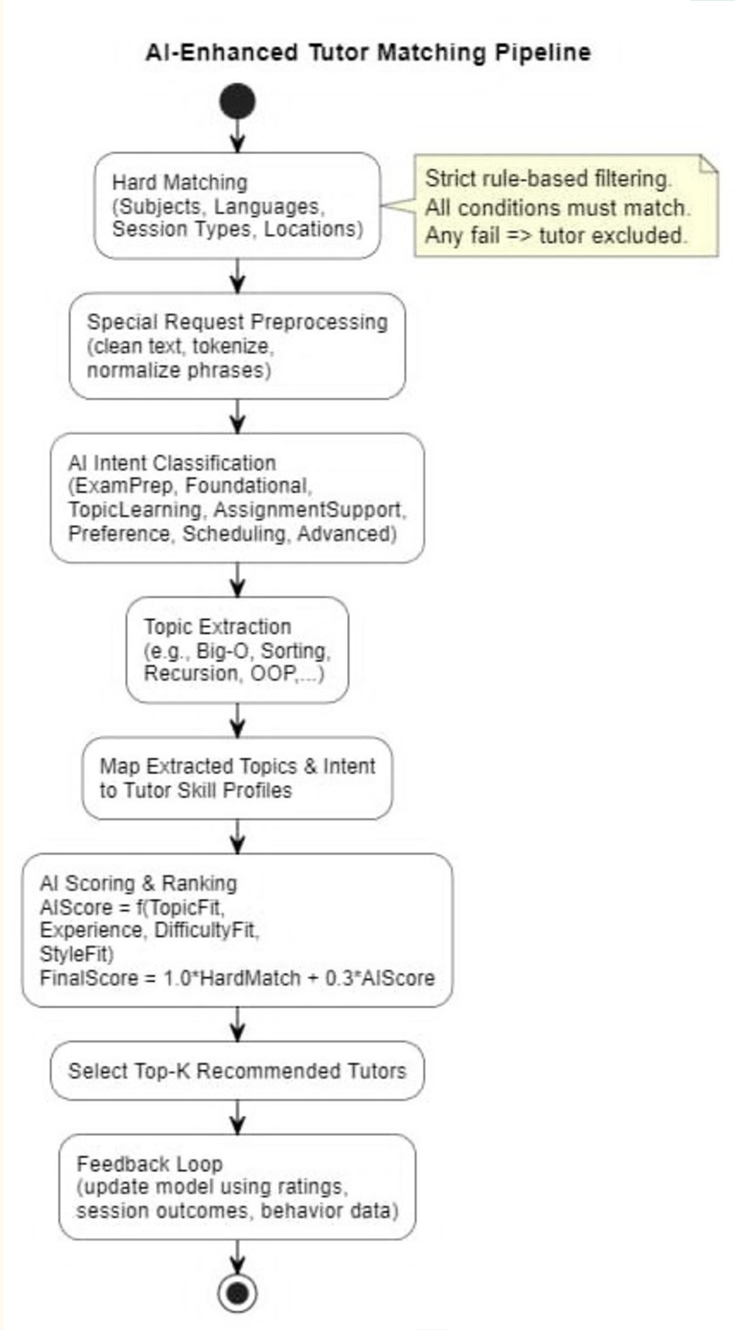
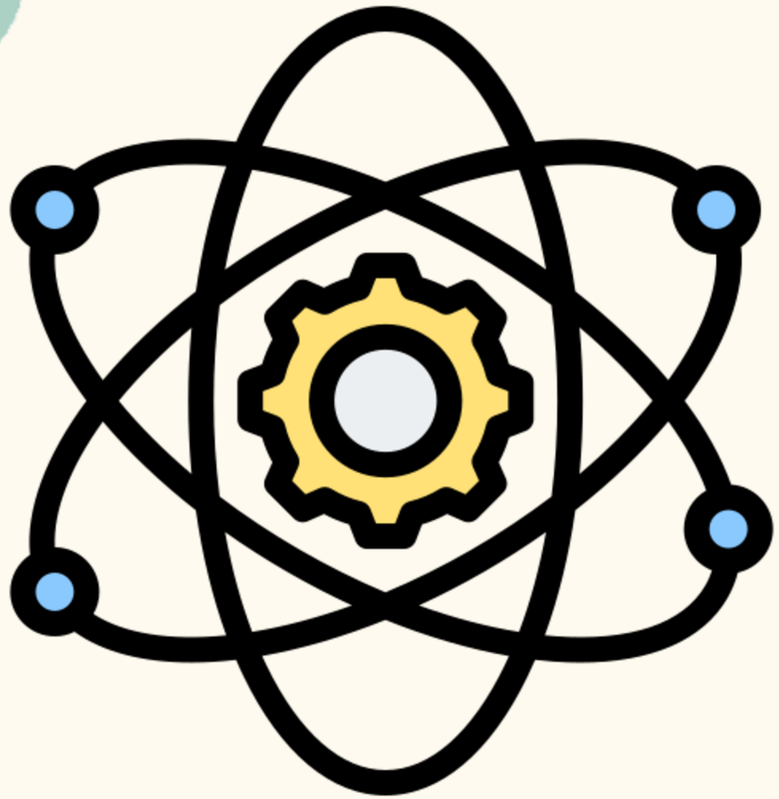


Table of contents

LIVE DEMO FROM THE GROUP

Table of contents

Conclusion



Core Features

- **Robust Architecture: Secure MVC design integrated with HCMUT services.**
- **Proven Reliability: Validated via 15+ test cases and AI matching logic.**
- **Ready for Impact: A unified platform optimizing academic support.**

THANKS FOR LISTENING