Team Super Ghost

Iteration 3

Hudson, Nathaniel - Jansen, Shane - Owens, Kirby - Zumwalde, Zach

Vision

(No new updates to vision)

Requirements

Requirements

Use Cases Chosen:

- (UC10) Registrar Add Student
- (UC11) Registrar Add Course
- (UC12) Student/Faculty Uploads Document

UC10: Add Student

Scope: Student Creation System
Level: registrar-goal
Primary Actor: Registrar
Stakeholders and Interests:

- Students: Wants to be able to access NKU-Net.
- Faculty: Wants proper students to be added to NKU-Net.
- Registrar: Wants to be able to add students as fast as possible.
- School: Wants student to be added quickly with little error.

Preconditions: User is already enrolled in school
Success guarantee (or Postconditions): Student is added and can access NKU-Net.
Main Success Scenario (or Basic Flow):

1. Registrar types in student's name and password.
2. System creates student's account.
3. System notifies success in creating user.

Extensions (or Alternative Flows):
*a. At any time, System fails:

1. To ensure security of system, Registrar is logged off and connection is shut off.

2a. Authentication Fails

1. System notifies Registrar of authentication failure.

Special Requirements:

- System adds student within 30 seconds, 90% of the time.
- Compatible with all major Internet browsers.

Technology and Data Variations List:
*a. Registrar can use any major web browser (Firefox, Google Chrome,
Safari, Microsoft Edge, etc.).

Frequency of Occurrence: Could be nearly continuous.

UC 11: Add Course

Scope: Course Creation System
Level: registrar-goal
Primary Actor: Registrar
Stakeholders and Interests:

- Students: Wants to be able to access their course.
- Faculty: Wants their course to be added quickly and accurately.
- School: Wants students/faculty to be able to access their course.

Preconditions: Registrar is signed in and authenticated.
Success guarantee (or Postconditions): The course is added to the database and the course page is accessible.

Main Success Scenario (or Basic Flow):

1. Registrar types in course information.
2. System creates the course.
3. System notifies the Registrar of successful course creation.

Extensions (or Alternative Flows):
*a. At any time, System fails:

1. To ensure security of system, Registrar is logged off and connection is shut off.

2a. Authentication Fails

1. System notifies Registrar of authentication failure.

Special Requirements:

- System creates course within 30 seconds, 90% of the time.
- Compatible with all major Internet browsers.

Technology and Data Variations List:
*a. Registrar can use any major web browser (Firefox, Google Chrome,
Safari, Microsoft Edge, etc.).

Frequency of Occurrence: Could be nearly continuous.

UC12: Student/Faculty Uploads Document

Scope: Upload Document
Level: user-goal
Primary Actor: Student/Faculty
Stakeholders and Interests:

- Student/Faculty: Wants to be able to upload a document to NKU-Net.
- School: Wants documents to be shared among students for collaboration.

Preconditions: Student/Faculty is logged into system upon being identified and authenticated.
Success guarantee (or Postconditions): Student/Faculty's document is uploaded to NKU-Net.

Main Success Scenario (or Basic Flow):

1. Student/Faculty views their profile..
2. Student/Faculty click the upload button.
3. Student/Faculty chooses file(s) from their local machine.
4. Document(s) are uploaded to NKU-Net.

Extensions (or Alternative Flows):
*a. At any time, System fails:

1. To ensure security of system, Student/Faculty is logged off and connection is shut off.

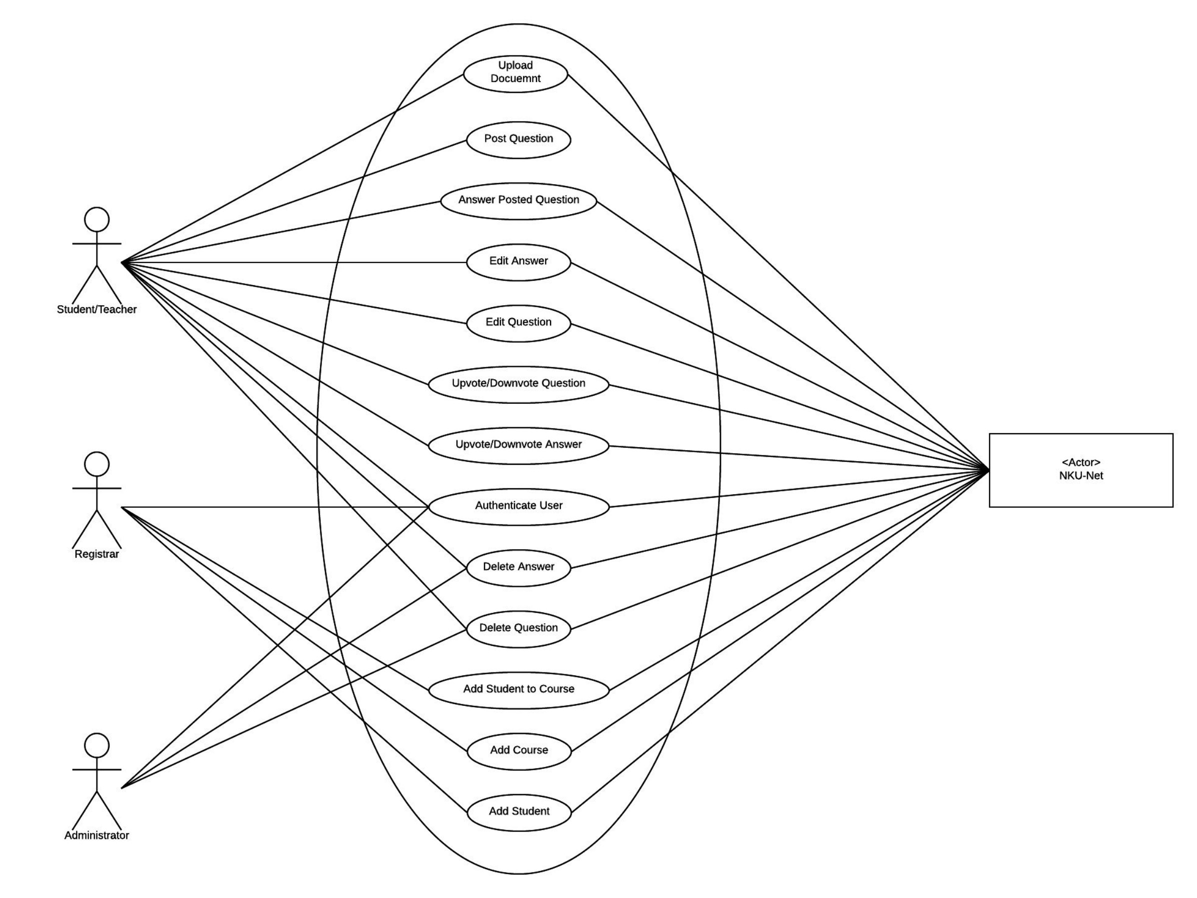
Special Requirements:

- System uploads document within 30 seconds, 90% of the time.
- Compatible with all major Internet browsers.

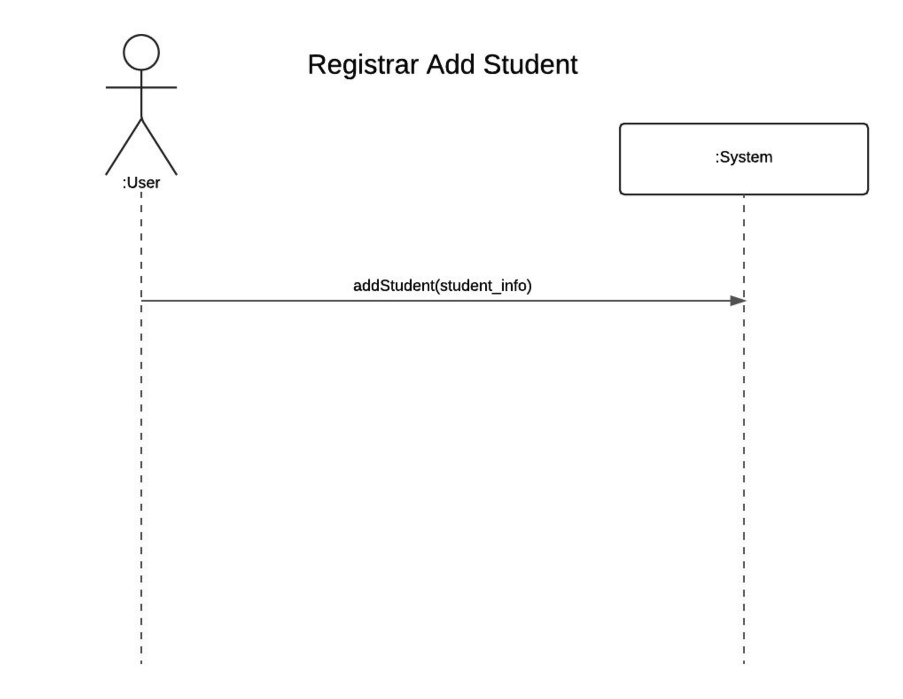
Technology and Data Variations List:
*a. Student/Faculty can use any major web browser (Firefox, Google Chrome,
Safari, Microsoft Edge, etc.).

Frequency of Occurrence: Could be nearly continuous.

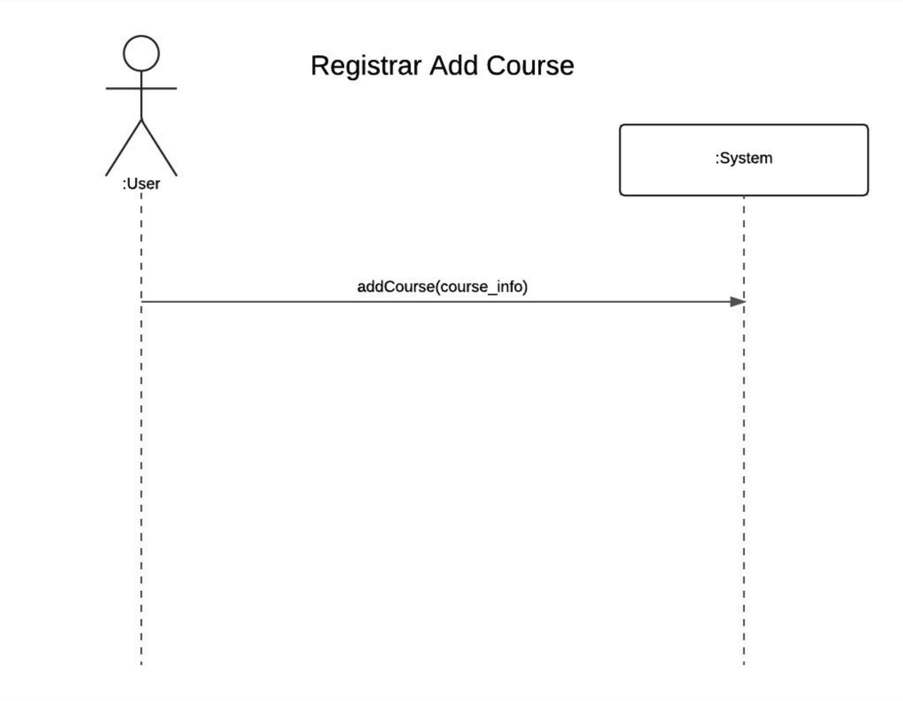
Use Case Model



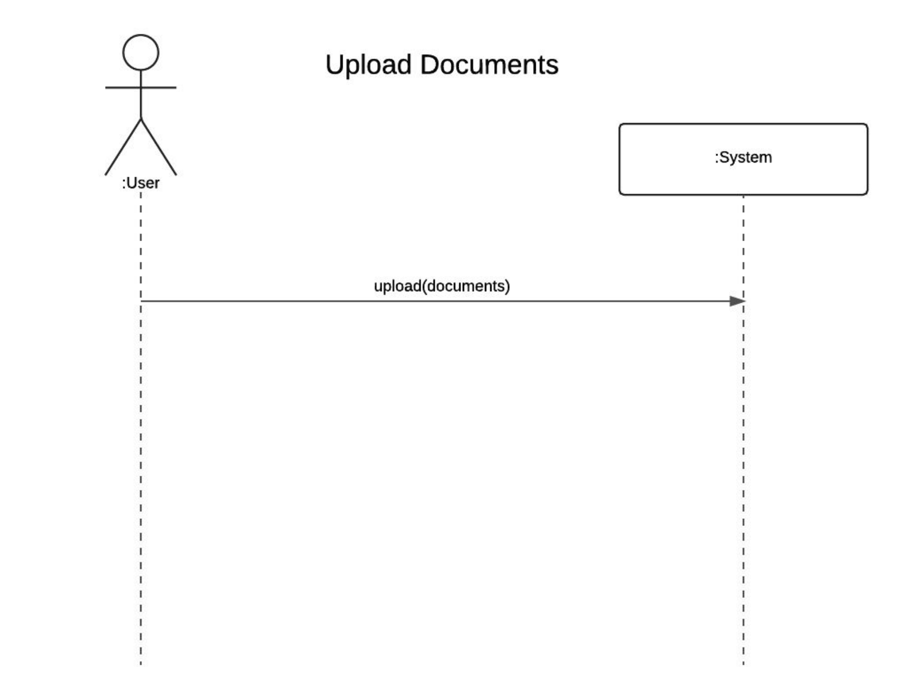
System Sequence Diagrams



System Sequence Diagrams



System Sequence Diagrams

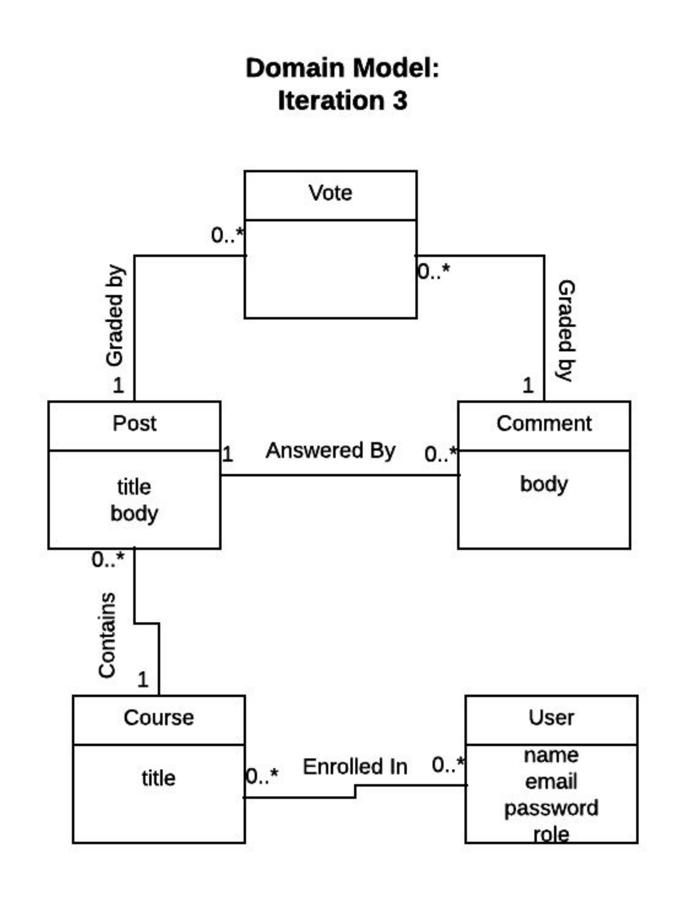


Design

Design: Overview

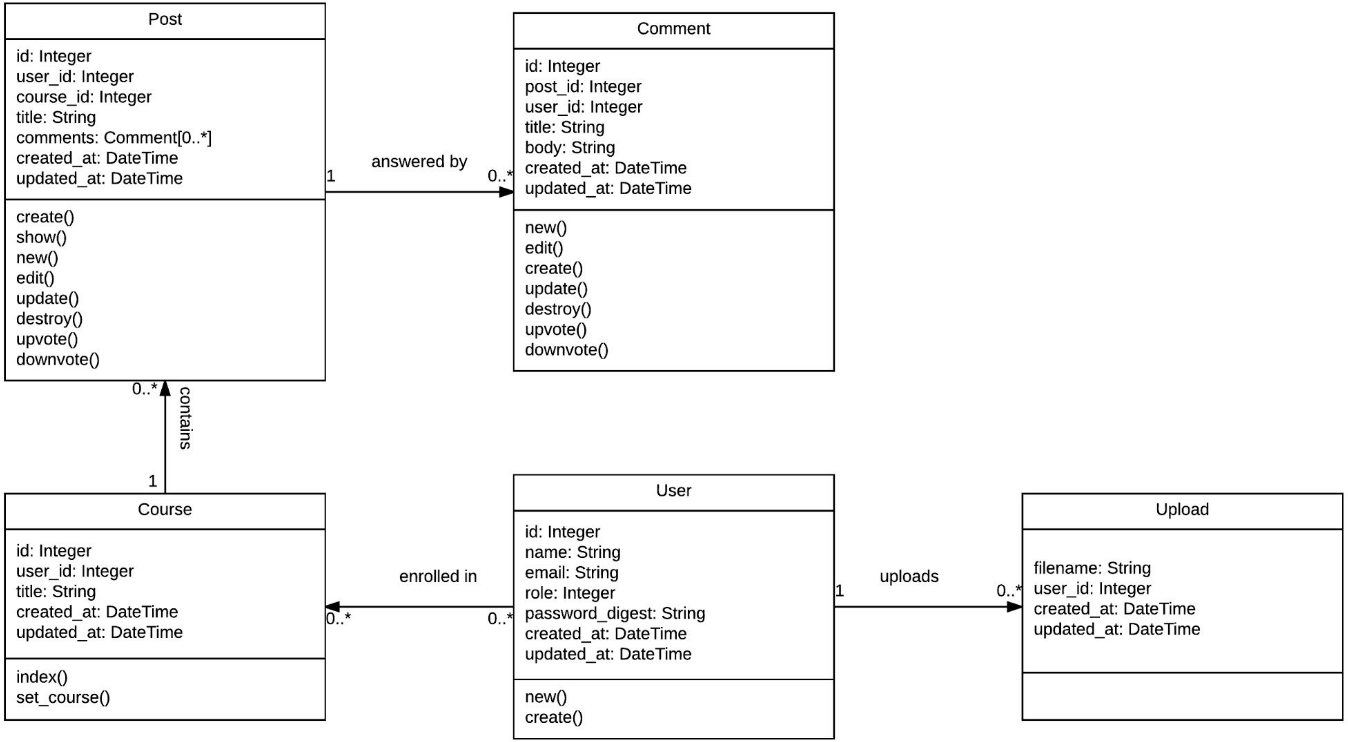
- Added uploads table to keep track of which documents were uploaded by which users.
- Added a directory where file uploads will be stored.
- Added role to user table to define Student, Teacher, Admin, or Registrar
- Added user_id to courses table so a teacher can be assigned to a specific course

Domain Model



Design Class Diagram

Design Class Diagram With All Classes & Associations



Post

id: Integer
user_id: Integer
course_id: Integer
title: String
comments: Comment[0..*]
created_at: DateTime
updated_at: DateTime

create()
show()
new()
edit()
update()
destroy()
upvote()
downvote()

Comment

id: Integer
post_id: Integer
user_id: Integer
title: String
body: String
created_at: DateTime
updated_at: DateTime

new()
edit()
create()
update()
destroy()
upvote()
downvote()

Course

id: Integer
user_id: Integer
title: String
created_at: DateTime
updated_at: DateTime

index()
set_course()

User

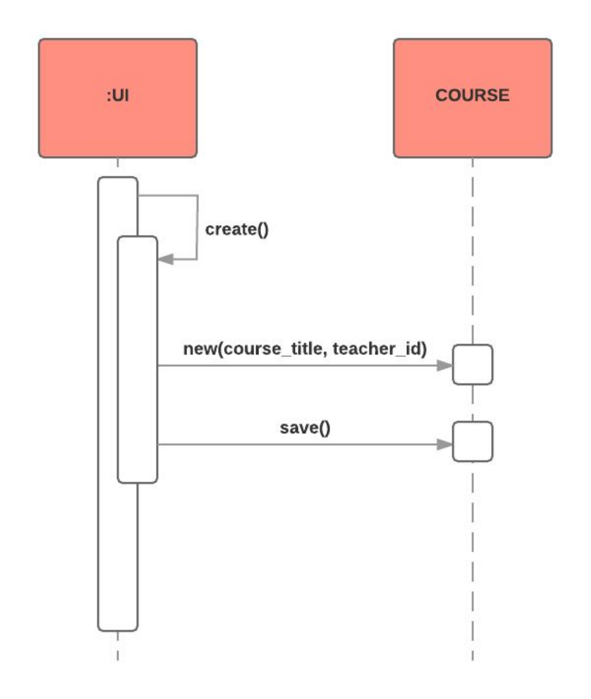
id: Integer
name: String
email: String
role: Integer
password_digest: String
created_at: DateTime
updated_at: DateTime

new()
create()

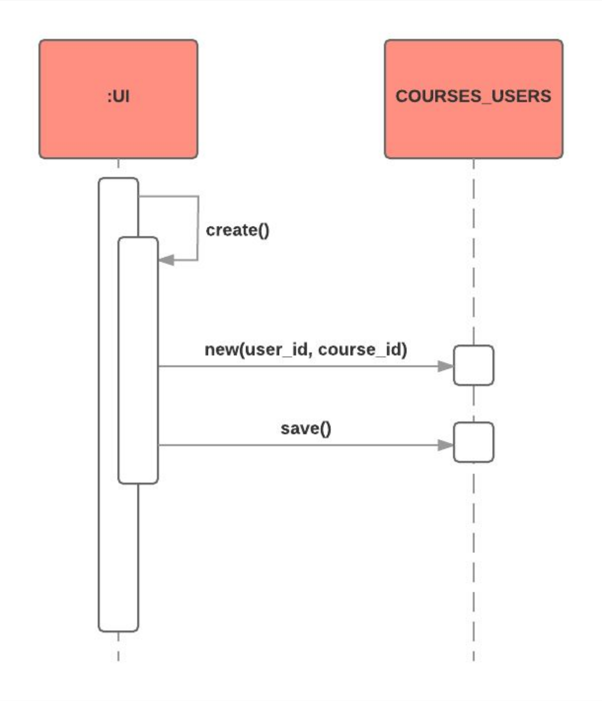
Upload

filename: String
user_id: Integer
created_at: DateTime
updated_at: DateTime

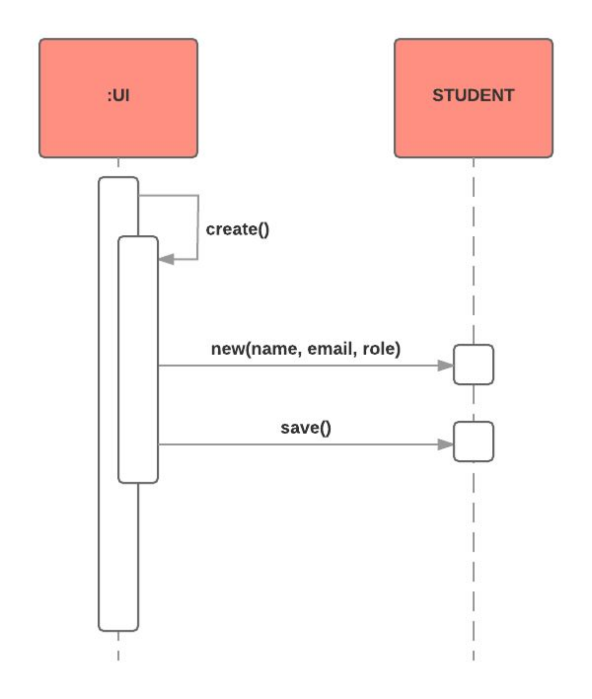
Design Level Sequence Diagrams - Add Course



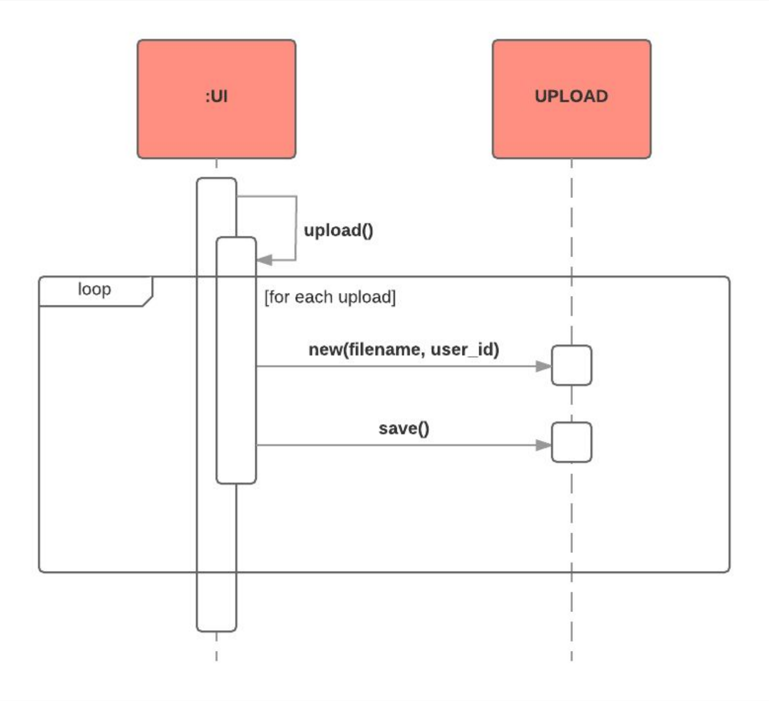
Design Level Sequence Diagrams - Add Student to Course



Design Level Sequence Diagrams - Add User



Design Level Sequence Diagrams - Upload



Demo