# Design Document for 2b |! 2b

Table of Contents:
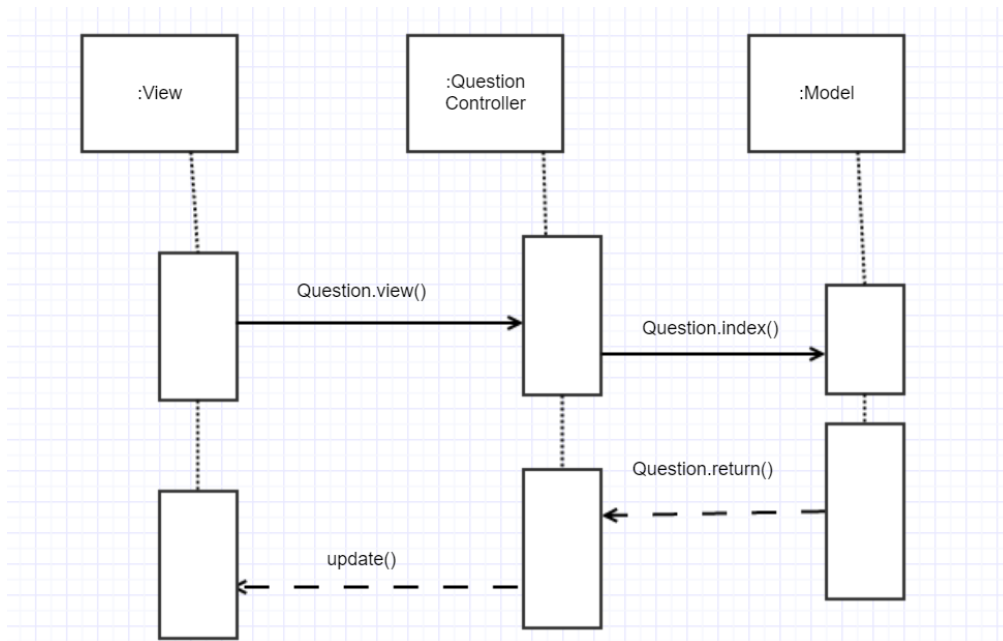
# Design Class Diagram

**User**

- id
- name
- email
- password
- role
- questions
- answers
- created_at
- updated_at

**Question**

- id
- title
- description
- user_id
- answers

**Answer**

- id
- content
- user_id
- question_id
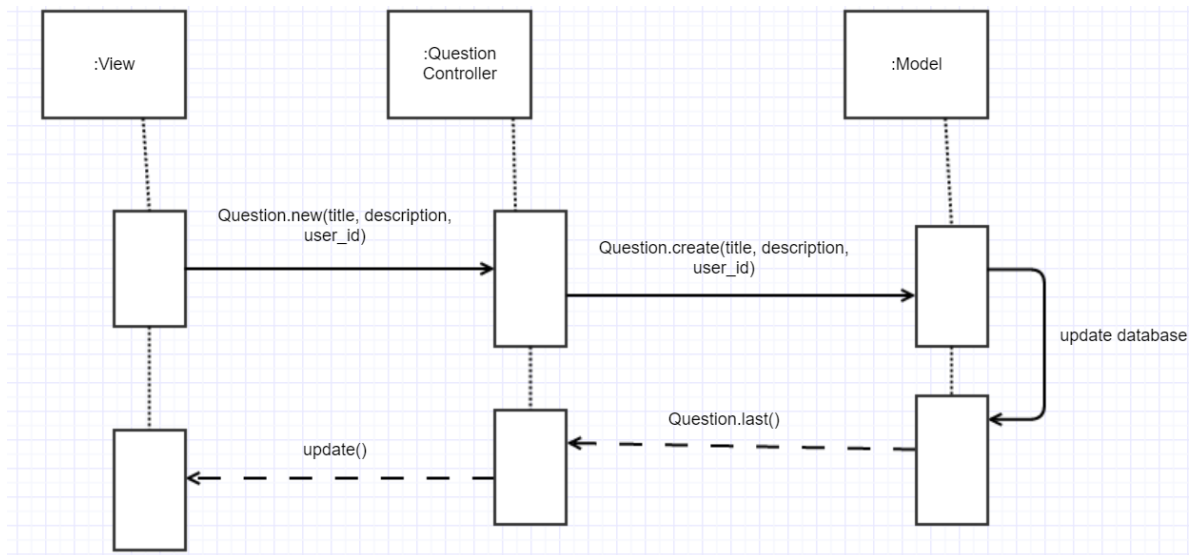- created_at
- updated_at

1 — 0..*

1 — 0..*

1 — 0..*

# Design level sequence diagrams

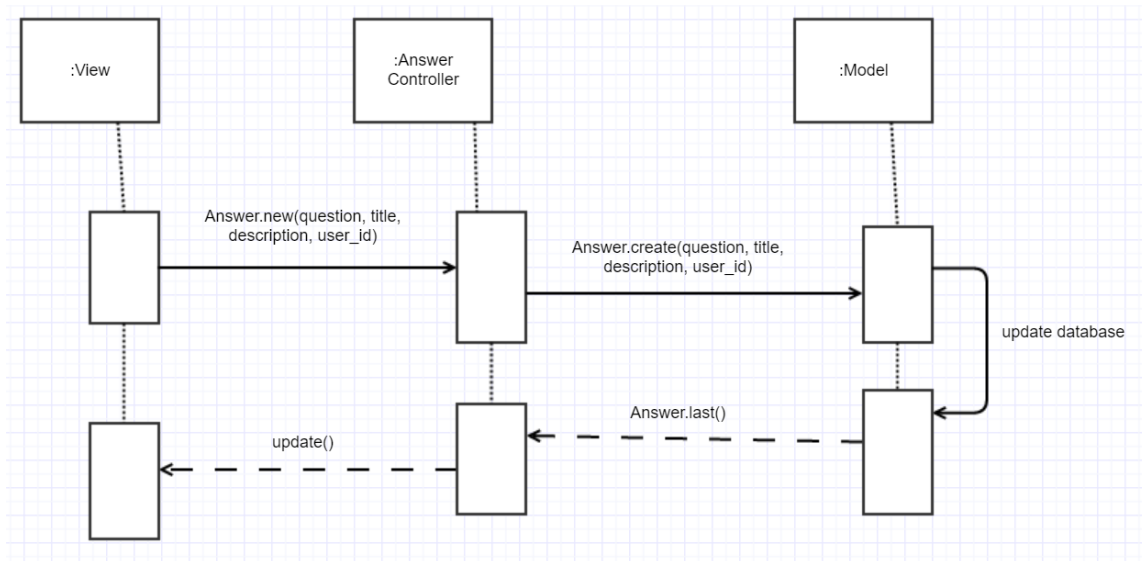## View Question



## Create Question

# Answer Question



**:View** → **:Answer Controller**: Answer.new(question, title, description, user_id)

**:Answer Controller** → **:Model**: Answer.create(question, title, description, user_id)

**:Model**: update database

**:Model** ⇠ Answer.last()

**:Answer Controller** ⇠ update()

# Create User



**:View** → **:User Controller**: User.new(name, user_id)

**:User Controller** → **:Model**: User.create(name, user_id)

**:Model**: update database

**:Model** ⇠ User.last()

**:User Controller** ⇠ update()
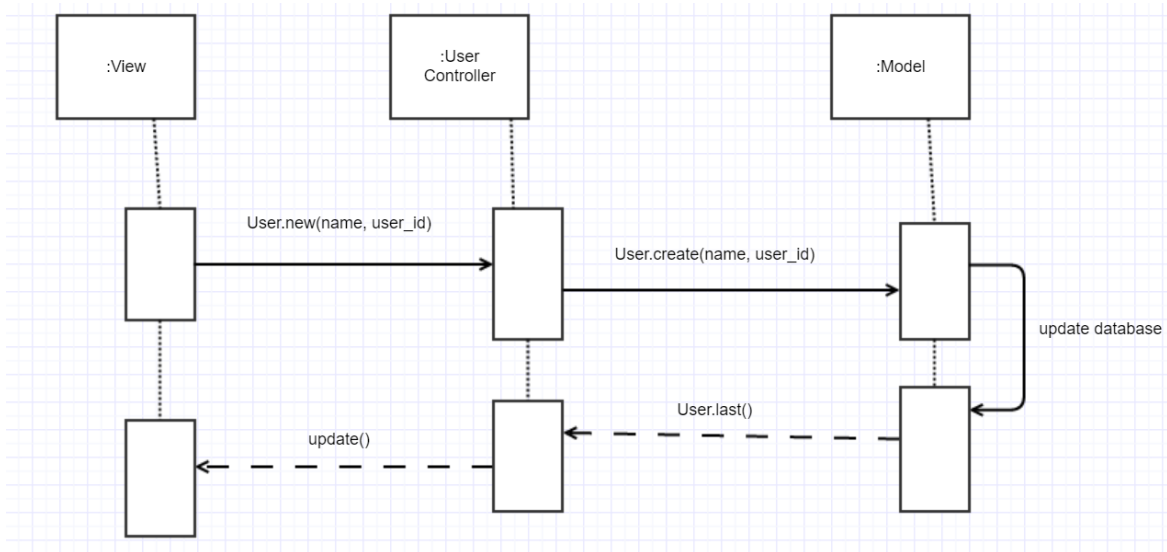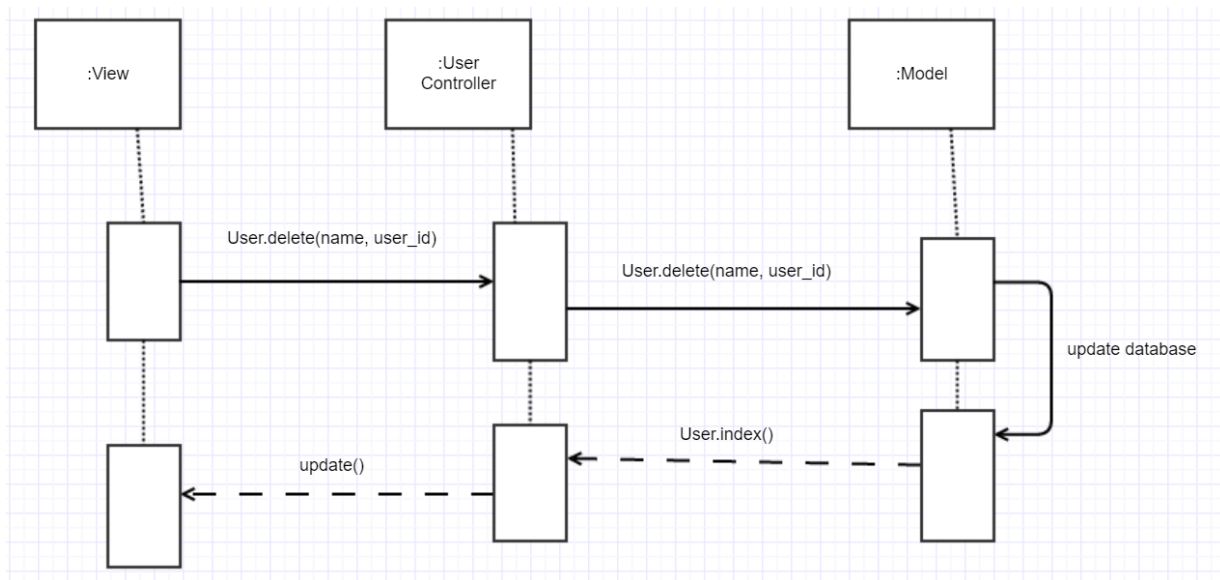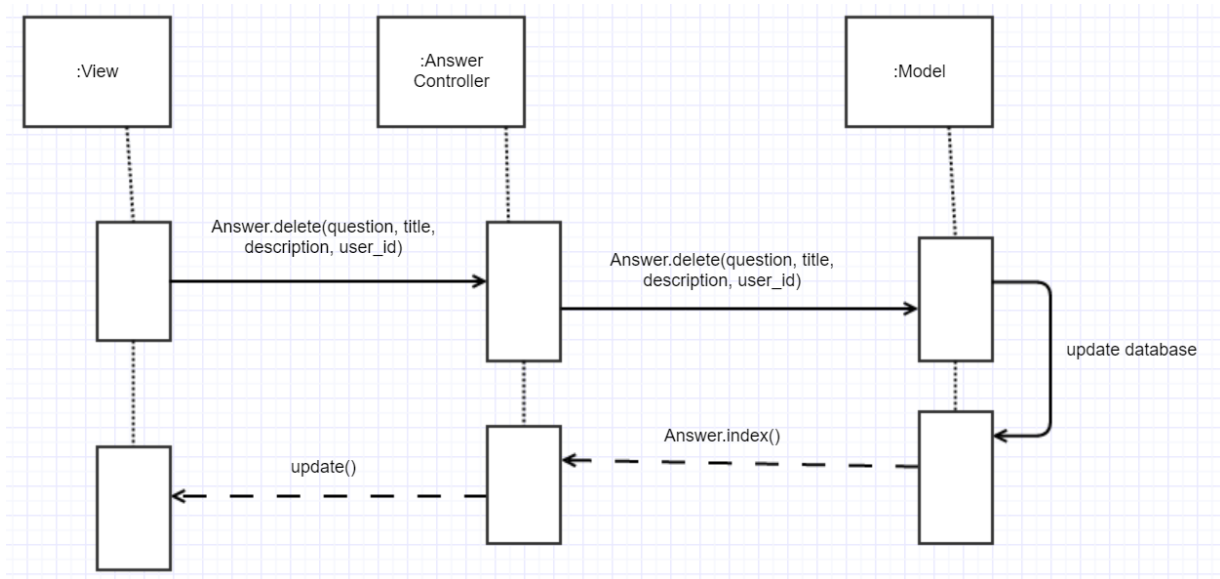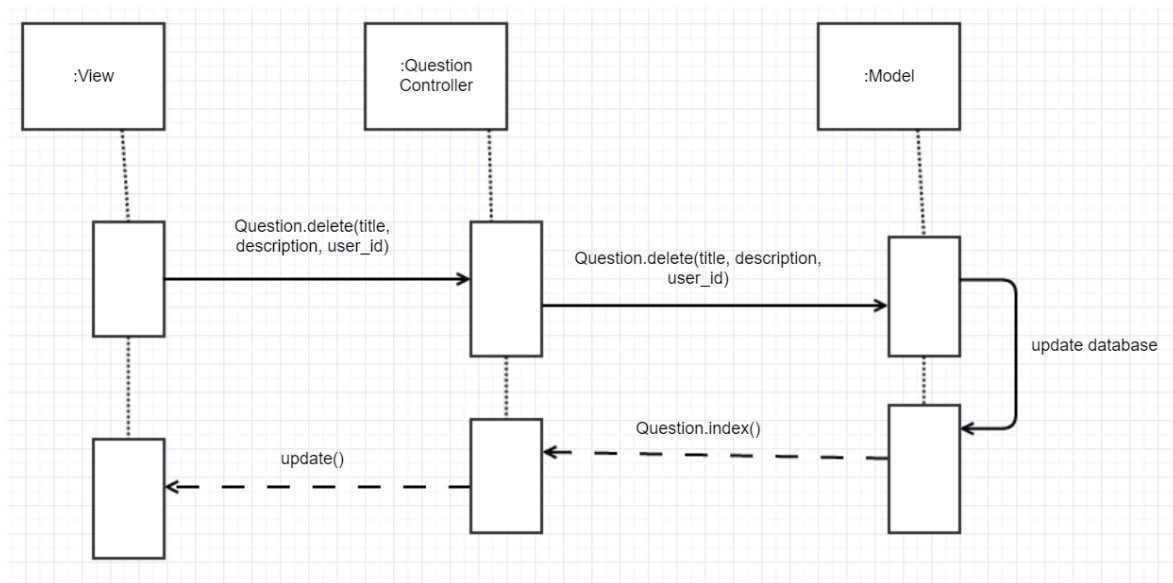
# Delete User



# Delete Answer

# Delete Question

## Discussion of design decisions

For our web application, we decided to have one single class that would contain our user. The "User" class would have a role tied to them that would determine if the user was a student, faculty, registrar or administrator. We achieved this by implementing a role variable, that when defined would allow the user to have the privileges of that role. The users who can post questions and answers (students, faculty and administrators) are also allowed to edit the answers/comments but not the original question. Admins have this right, so they can delete any question they see fit.

## Discussion of database design

The database, as of right now, consists of Users, Questions, and Answers. Each User can have many Questions and many Answers, which they have posted to the site. Each Question can have many Answers posted in the question thread.