



# DESIGN DOCUMENT

CSC 440 Semester Project

Brian Vach, Ethan Perry, Jared Bryant, Chris Fitzgerald

## Table of Contents

Iteration 1 Design Decisions	2
Iteration 2 Design Improvements	3
Design Class Diagram	4
UC1: Add User (Design Diagram)	5
UC2: Edit User (Design Diagram)	6
UC3: Ask Question (Design Diagram)	7
UC4: Answer Question (Design Diagram)	8
UC4: Answer Question (Alternate Flow Design Diagram)	9
UC5: Authenticate User (Design Diagram)	10
UC6: Vote on Question (Design Diagram)	11
UC7: Vote on Answer (Design Diagram)	12
UC8: Logout User (Design Diagram)	13

## **Iteration 1 Design Decisions:**

The CSC 440 Iteration 1 Demo App was built using Ruby on Rails, taking advantage of the scaffolding feature that is made available in Rails. Upon loading the website, the user is presented with a list of courses, each with a link to questions related to the course. The user can then click on one of these links, and view the questions, or submit a new one. If the user clicks on a question, they are shown answers for that question, and a link to submit new answers. At the top of every page is a nav bar with a link to the home courses page, and to a user management page, where new users can be created and existing users edited. There is also a search bar with functionality that has not been implemented yet. It was decided to use Bootstrap as the CSS framework because it integrates with Rails extremely well, and is very easy to implement. The courses, questions, and answers are displayed as bootstrap “cards” because the card offers a built-in border with rounded edges that we liked. For the headers of the card lists on the questions and answers pages, we used a “jumbotron” object that complements the cards well. Overall we chose a blue/grey theme (except for the search button, which is green for a complement).

## **Iteration 1 Database Design Decisions:**

The database for the CSC Iteration 1 Demo App consists of 4 tables. These tables are “Courses”, “Users”, “Questions”, and “Answers”. The decision was made to limit the database to these 4 tables for now for simplicity. Later in the project, more tables may be added. First, let’s look at the “Courses” table. It was decided that this table only needed three fields for now, Name, Description, and Professor. This was enough to get meaningful data stored for a demo. Next is “Users”. This table contains Name, Email, and Course\_ID. It was decided that for now, a user will only belong to one course. Later this will be changed. Next is the table “Questions”. This table contains Course\_ID, Title, User\_ID, and Content. It was decided that a question can only belong to one class, and one user. We do not anticipate this to change. The last table is “Answers”. This table is very similar to “Questions”. It contains Question\_ID, User\_ID, and Content. An answer should only belong to one question and one user. Again, we do not expect this to change.

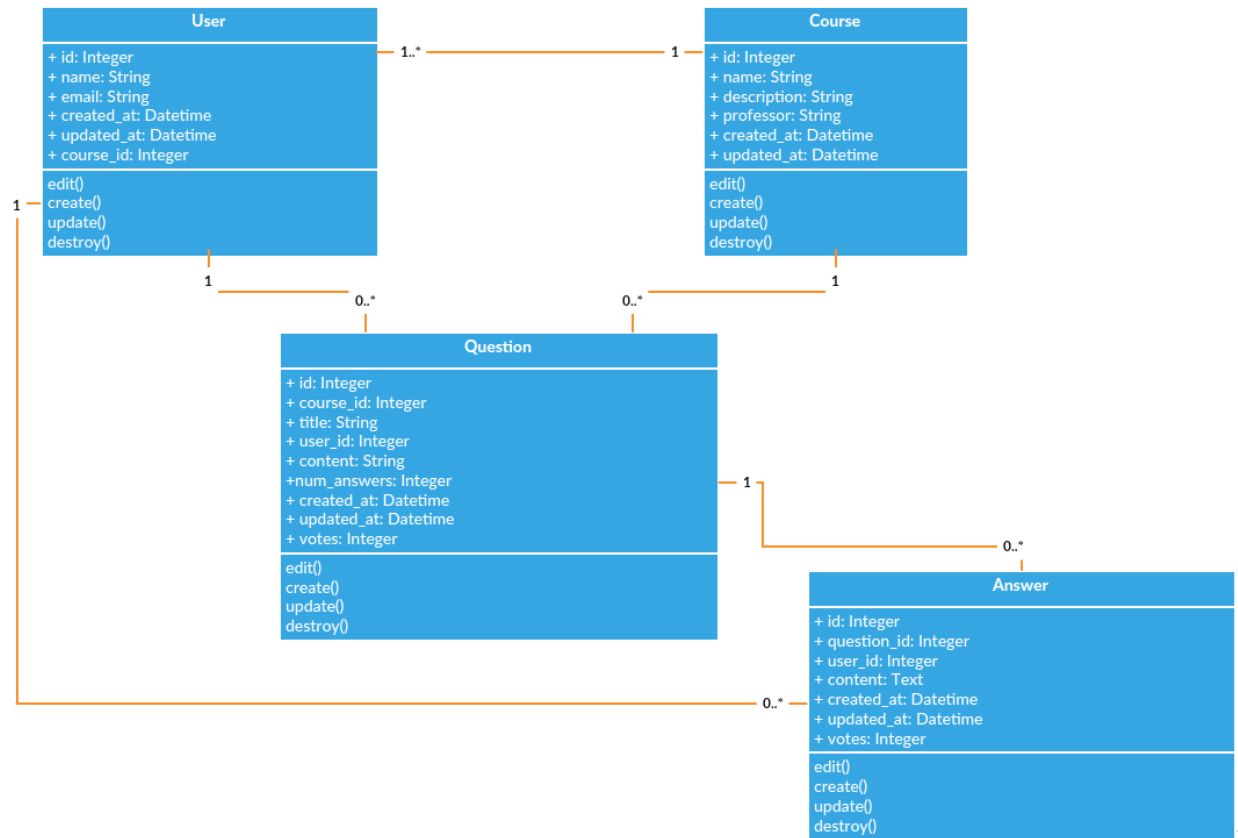
## **Iteration 2 Design Decisions:**

For this iteration, we have added more functionality to the website. We have added additional use cases which include add user, edit user, authenticate user, logout user, vote on question, and vote on answer. All the added functionality allows for users to interact with the system on a personal level. In the first iteration, users would select which user they would want to post as, which could obviously cause issues with building credibility on the website. The main functionality that was added to this iteration was the addition of voting, which will be most noticeable to the user. Adding and editing users is more of an extension of what was already present in the previous system. We have yet to include a search bar as was wanted from iteration 1.

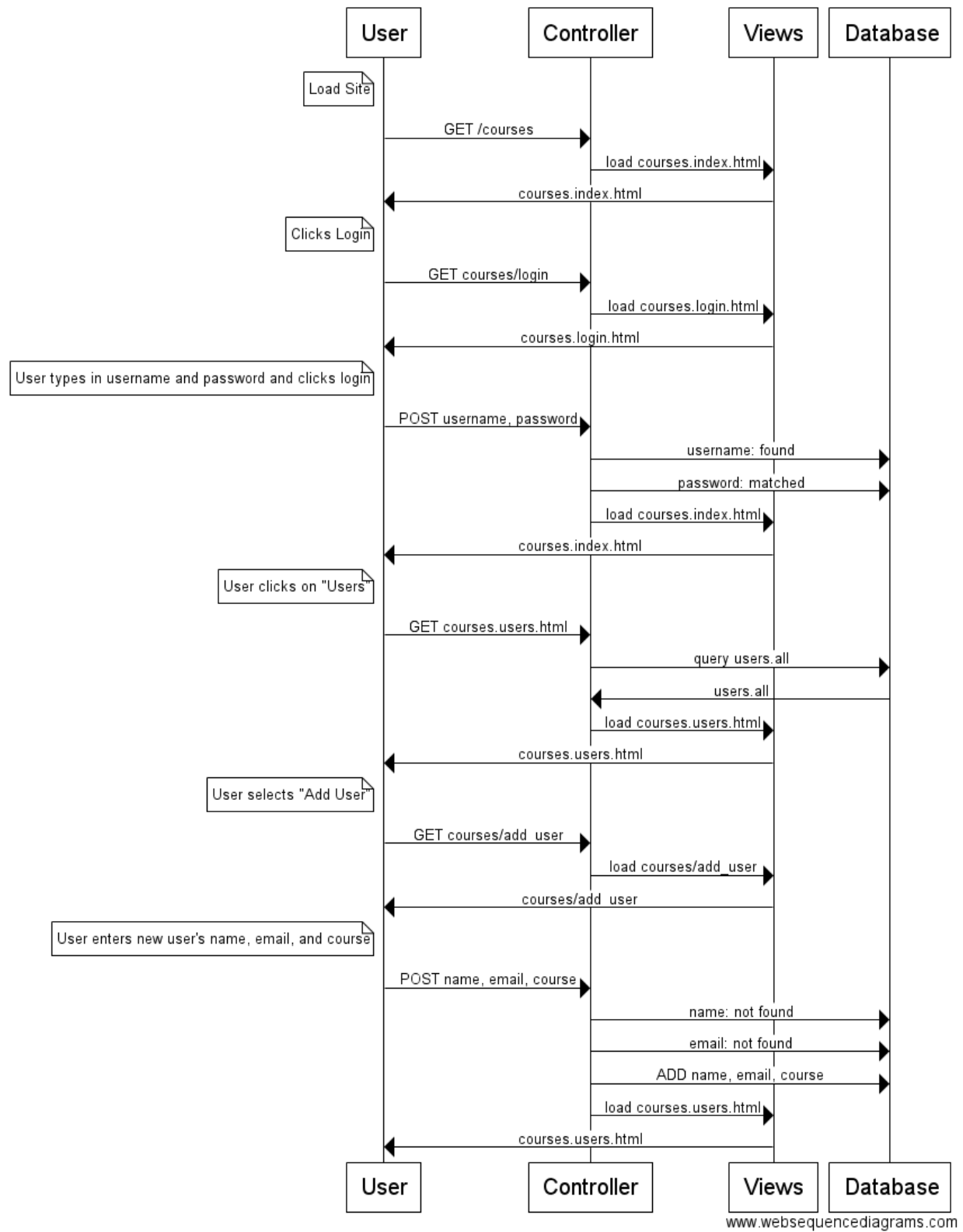
## **Iteration 2 Database Design Decisions:**

For this iteration, we have kept our database design decisions the same with 2 small changes. We now have votes added as a field for both questions and answers. Since users are now able to vote on questions and answers, the database needs to keep track of how many votes each has. We have also added an "is\_admin" field to the user's table, to separate admins from non-admins.

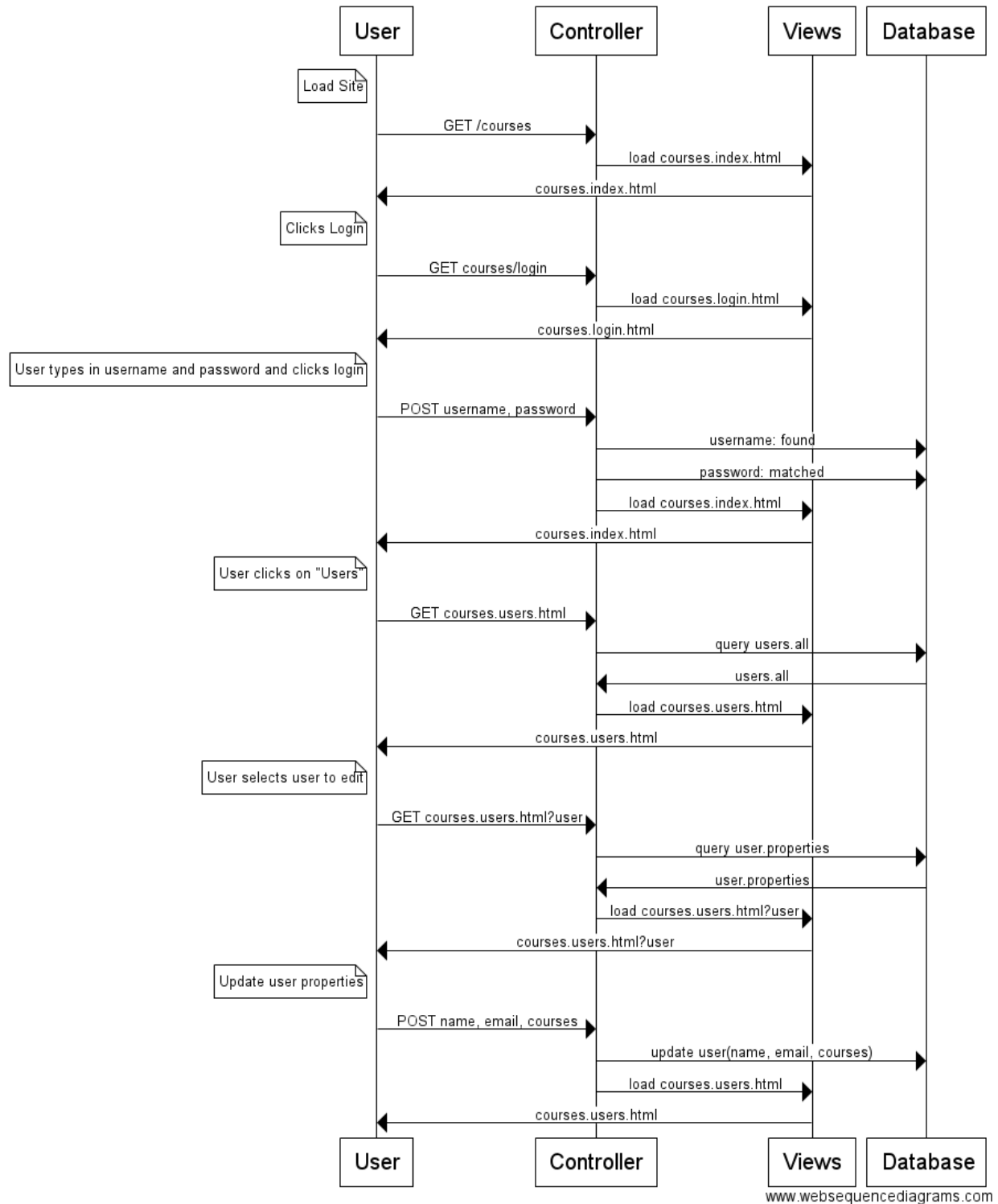
## Design Class Diagram



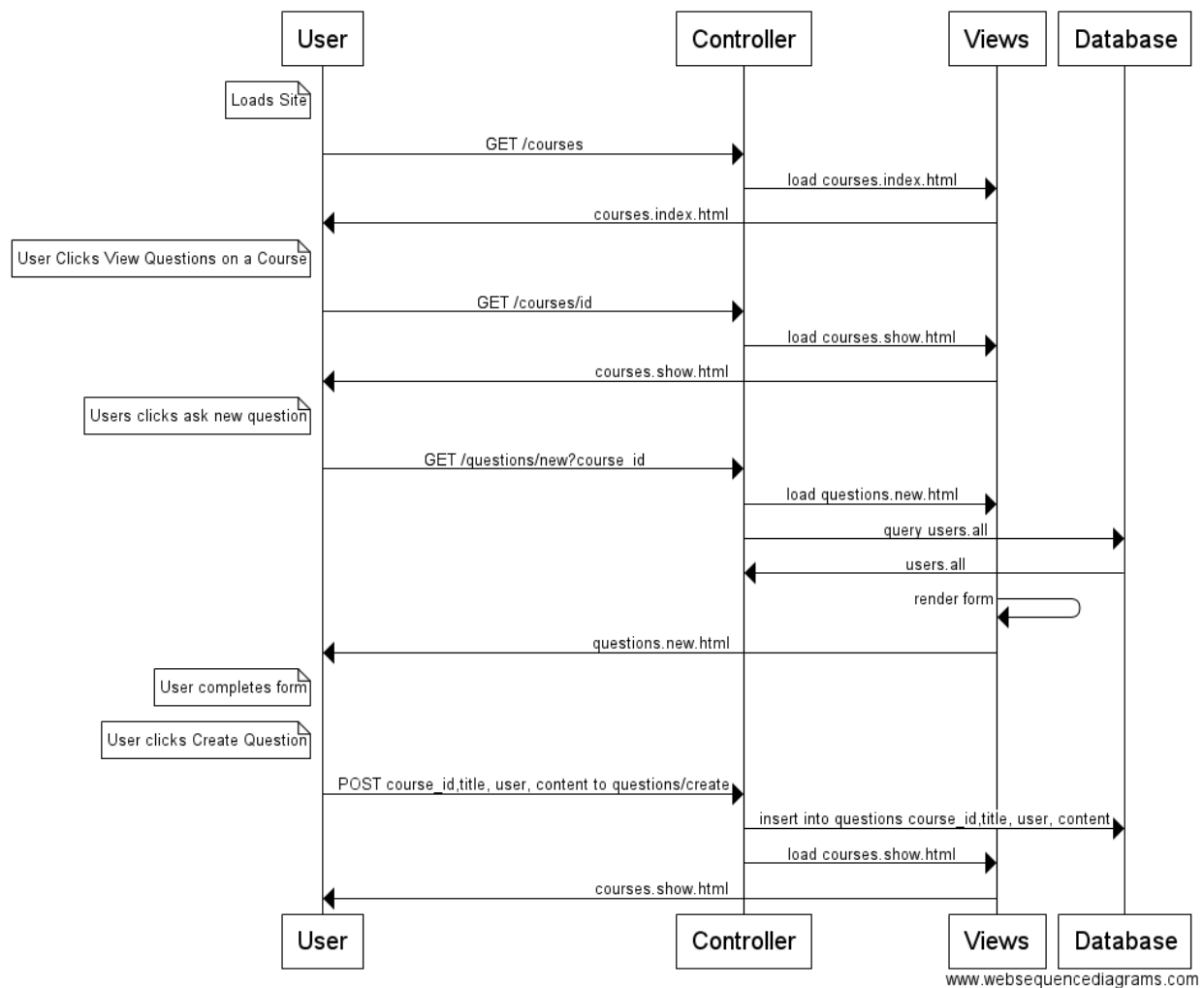
## Add User



## Edit User

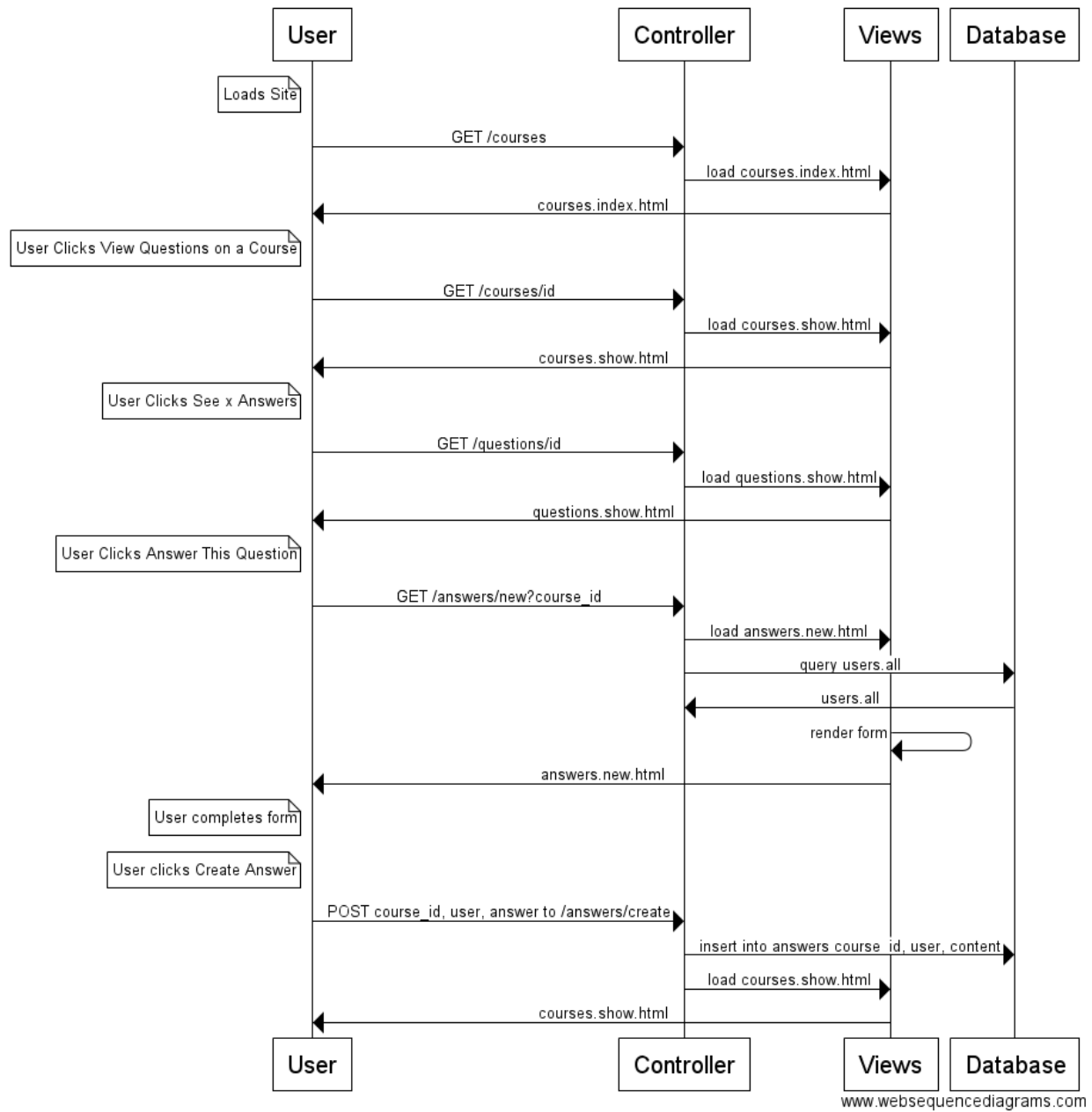


## Post Question

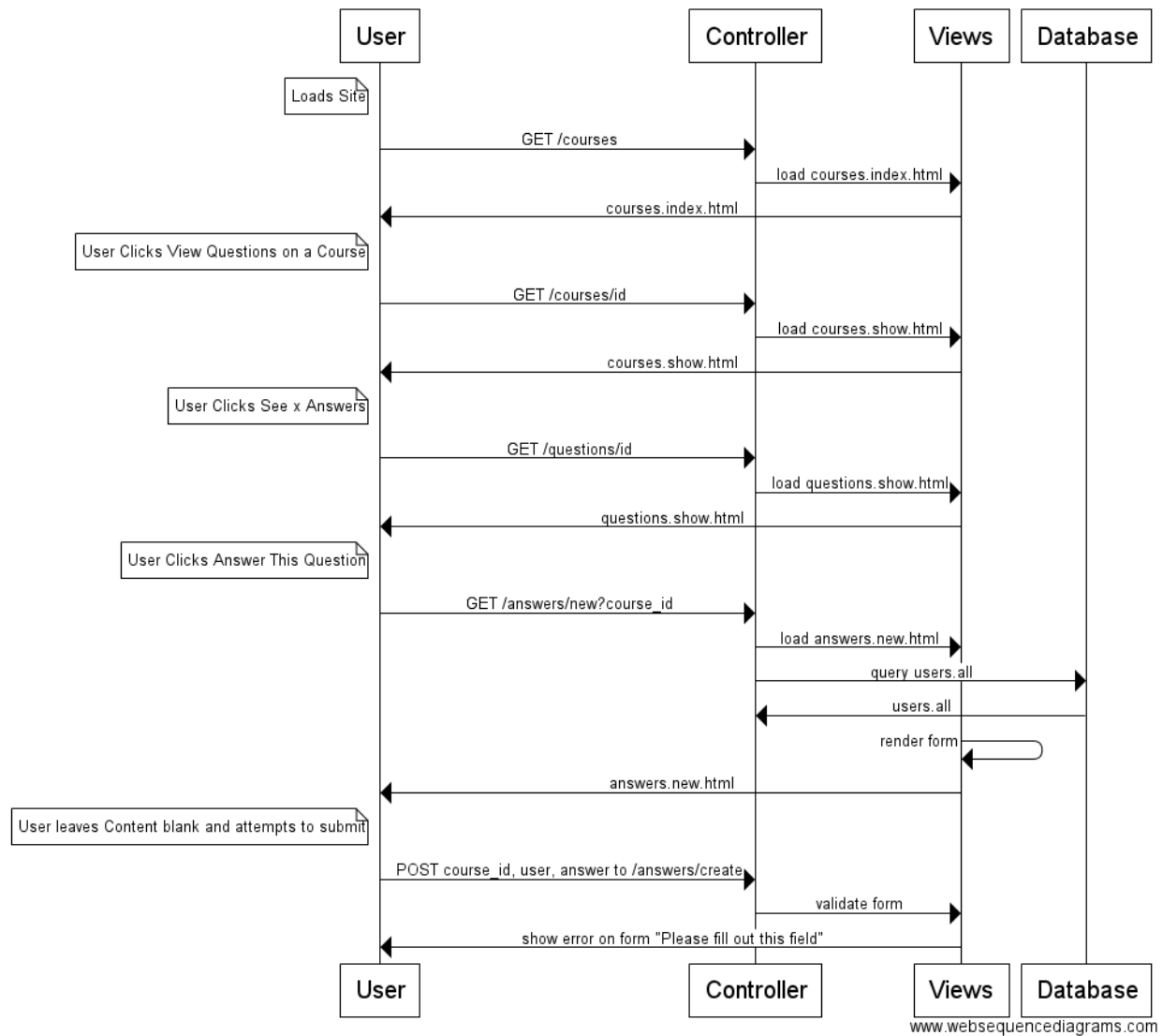




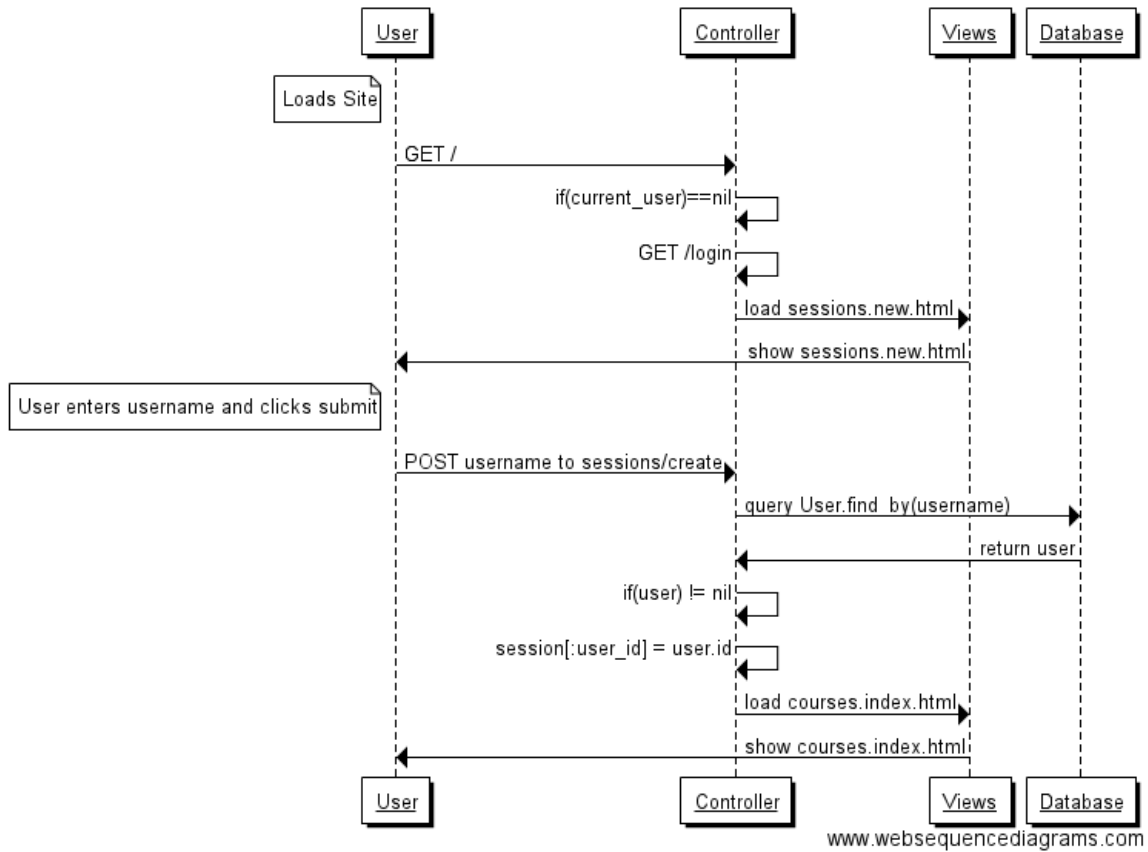
## Answer Question



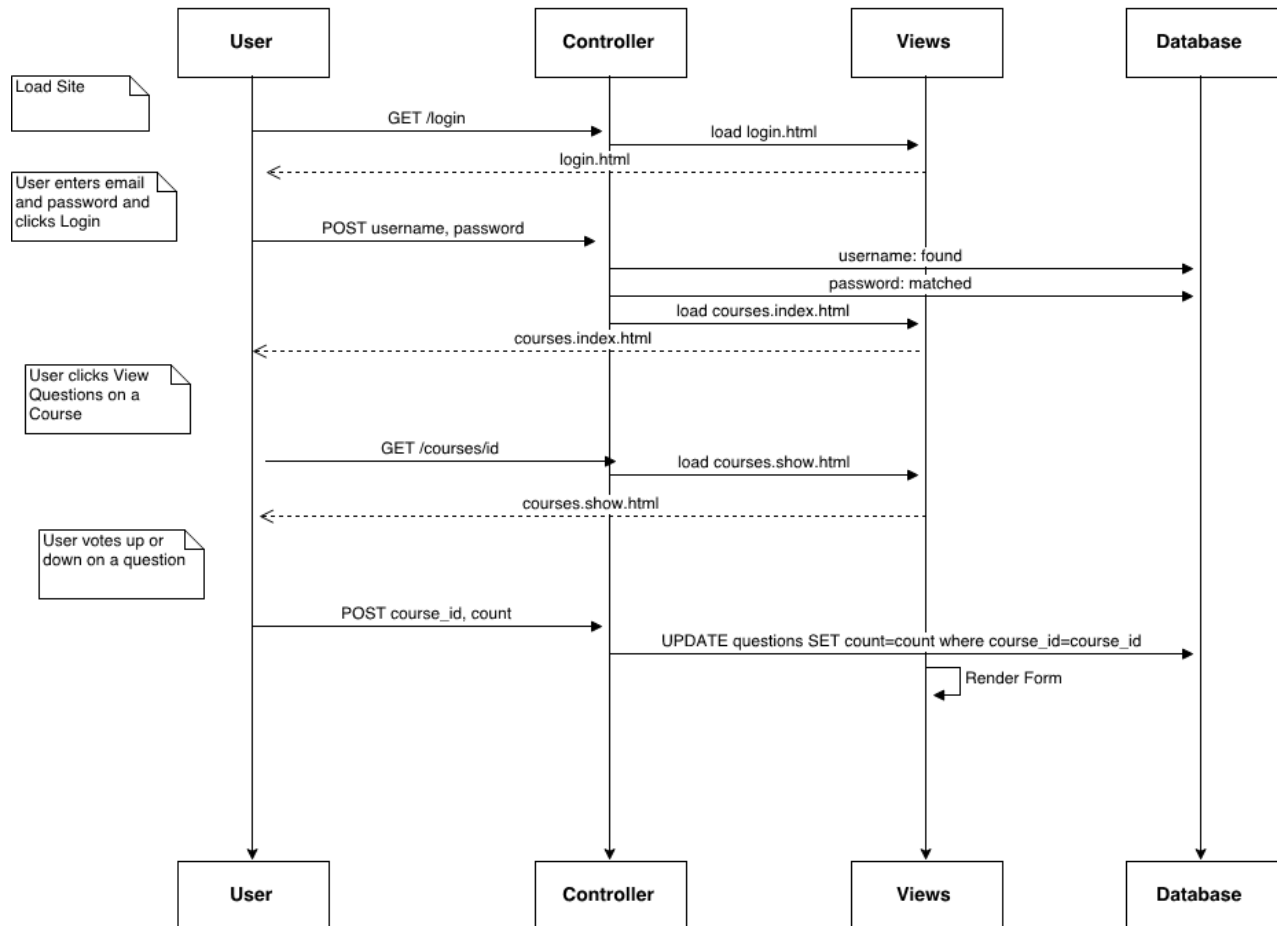
### Answer Question(Alternate Flow 4b)



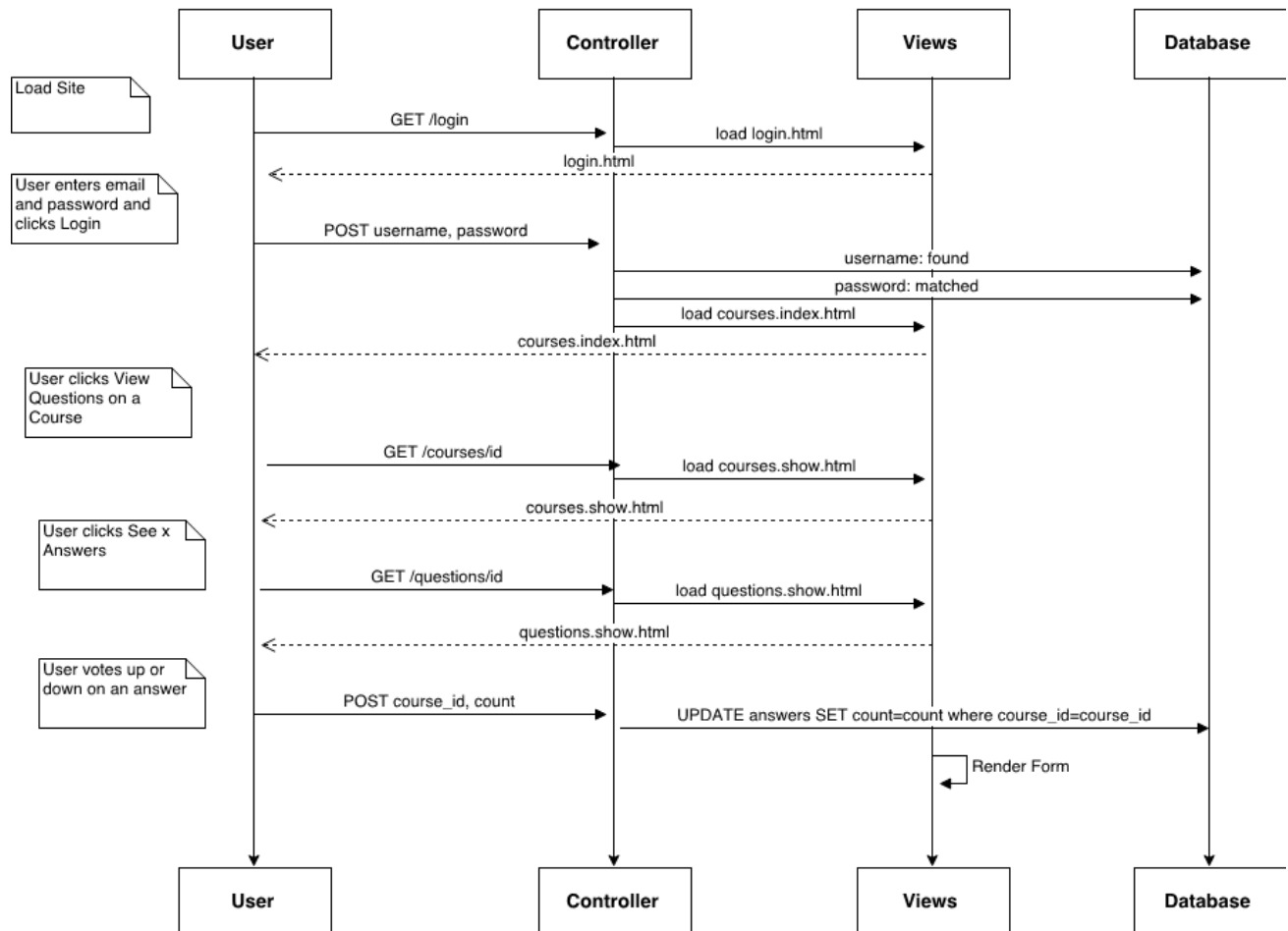
## Authenticate User



## Vote on Question



## Vote on Answer



## Logout User

