

Quora Clone Part 3: Question & Answer Feature

Time	~ 3/4 day
Learning Goals	<ul style="list-style-type: none">• Deeper understanding of MVC• ActiveRecord associations• Understand authentication

Questions & Answers

Now that we're done with building users, let's build the question and answer features for our mini-quora app. Today you'll be implementing your wireframe and schema to build the following features:

- An index/ home page listing all the questions (top stories)
- A page showing a particular question including its answers
- A page showing all a user's submissions (e.g. "My Questions")
- A page showing all a user's answers (e.g. "My Answers")
- Should certain features or pages only be accessible by logged in users? How do you use the helper methods we created earlier to **locked** down certain pages? If a not-logged-in user tries to do anything that isn't permitted, redirect them away from the page to somewhere sensible, e.g., the homepage with a flash message indicating the error.

Don't worry about Quora's fancy sorting algorithms for the homepage and answer threads. Just sort them in some simple way: chronologically, alphabetically, etc.

Don't forget to commit early and commit often! Save your edits by committing often and show your friends the work done so far. Once again, work on a branch for each of the question and answer features. Every time you are completed with one feature, merge back to master and checkout a new branch again. In this case:

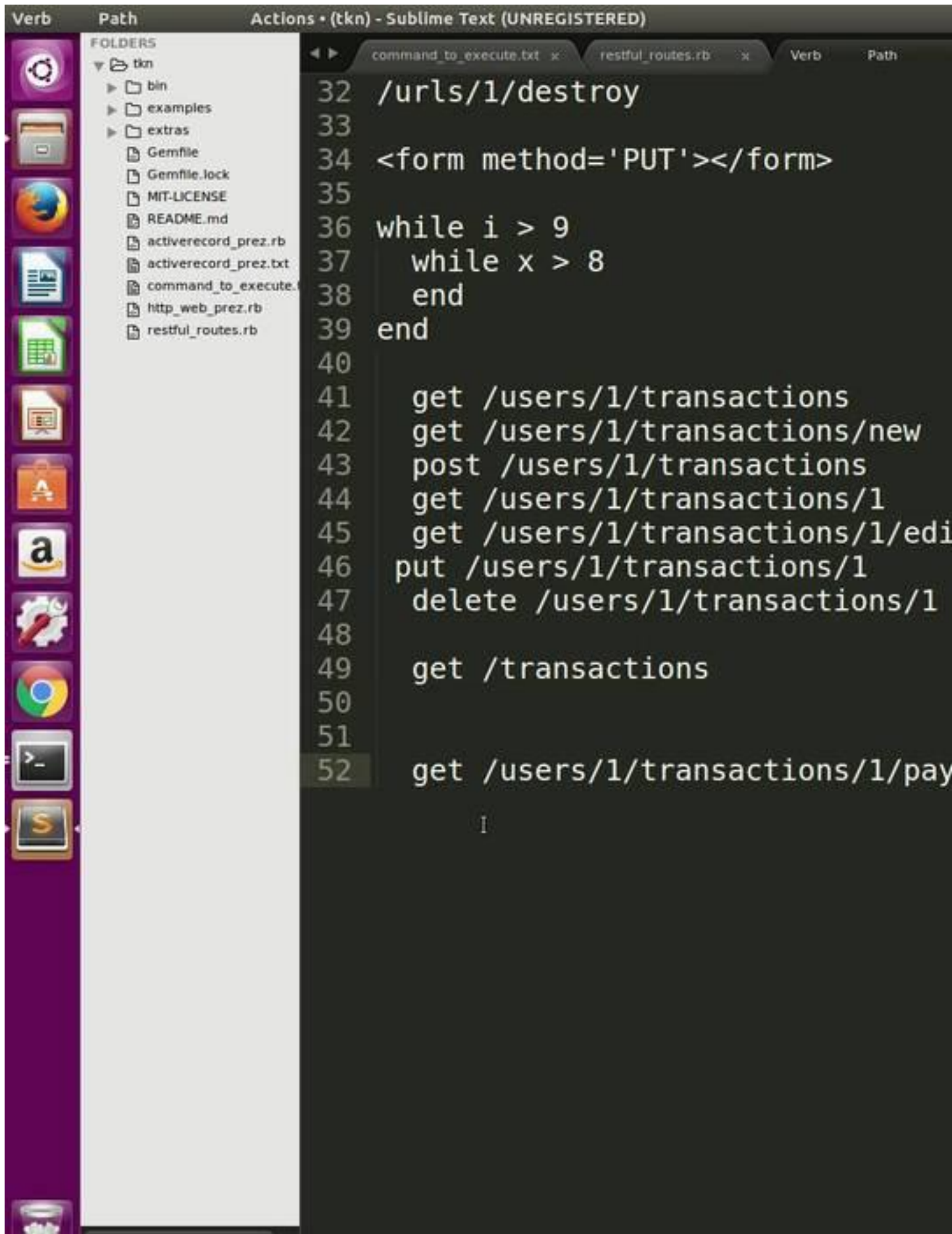
1. **git checkout -b feature/question**
2. Do the necessary **git add** and **git commit**.
3. **git merge master**
4. **git checkout master**
5. **git checkout -b feature/answer**
6. Do the necessary **git add** and **git commit**.

Quora Clone Part 3: Question & Answer Feature

7. `git merge master`
8. `git checkout master`

ps: This is a good exercise to reinforce your learning and understanding on Active Record associations. Please talk to your mentor if you need more guidance on Active Record. It will be very helpful for your assessment on Friday too!

Quora Clone Part 3: Question & Answer Feature



```
Verb    Path
FOLDERS
└─ tkn
  └─ bin
  └─ examples
  └─ extras
  └─ Gemfile
  └─ Gemfile.lock
  └─ MIT-LICENSE
  └─ README.md
  └─ activerecord_prez.rb
  └─ activerecord_prez.txt
  └─ command_to_execute.txt
  └─ http_web_prez.rb
  └─ restful_routes.rb

32  /urls/1/destroy
33
34  <form method='PUT'></form>
35
36  while i > 9
37    while x > 8
38    end
39  end
40
41    get /users/1/transactions
42    get /users/1/transactions/new
43    post /users/1/transactions
44    get /users/1/transactions/1
45    get /users/1/transactions/1/edit
46    put /users/1/transactions/1
47    delete /users/1/transactions/1
48
49    get /transactions
50
51
52    get /users/1/transactions/1/payments

I
```

Quora Clone Part 3: Question & Answer Feature

When you were building your user controller, did you, at any point, wonder how you should name your URL? Well, I did! When I built my first app, I wasn't aware that a convention was already in place. I thought I was pretty cool to come out with my own names, such as "show-user-profile", "delete-user", "show-all-users", after all, it was pretty easy to understand and read. But when I realised that I will have to keep thinking of fancy names every time I build an app, I realised it wasn't that fun after all.

Fortunately, I learnt about REST, and everything changed after that. REST, or Representational State Transfer, is a mapping between a HTTP verb (GET, POST, PUT, DELETE) and a CRUD action (create, read, update, delete). It provides a consistent pattern for naming URLs and does make the life of a developer much easier! For this reading challenge, we would like you to read through [this](#) and [this](#), and take note of the following:

- 1 What are the common HTTP verbs?
- 2 What are the corresponding CRUD actions of these verbs?
- 3 What is a REST resource?
- 4 How does a REST URL look like?
- 5 What does a RESTful route depend on?
- 6 What is the key to REST?
- 7 What is a nested resource?

Once you are more comfortable with RESTful routes, for the user model you have built, fill in the appropriate HTTP verb, path and the purpose of the action in the following table:

HTTP Verb	Path	CRUD Action	Purpose
GET	/users/index	#index	
		#new	
		#create	
		#show	
		#edit	
		#update	
		#destroy	

Once you are done, continue working on the Quora clone. Don't forget to jot down what you have learnt in your notes.