

Beginning Ruby on Rails

Session Six

Advanced Topics

- A new app from scratch
- User Sessions
- More Associations

The New App

How about a simple “question and answer” app?

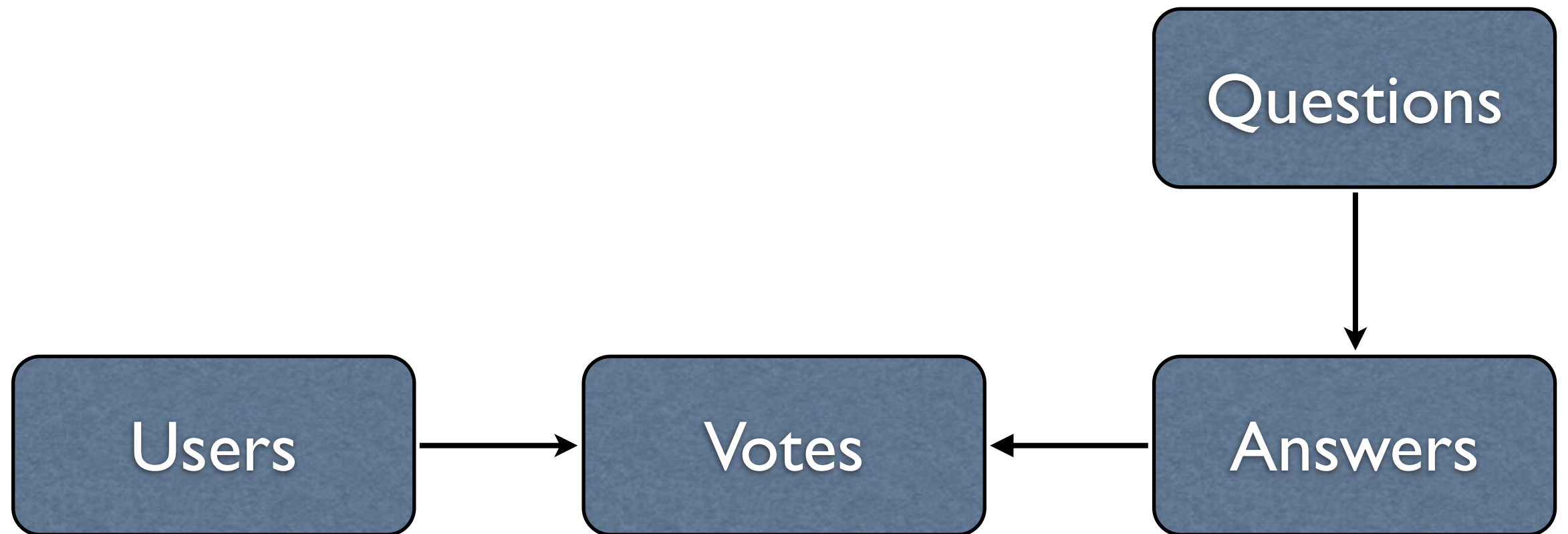
Features

- Anonymous users will be able to register an account and log in.
- Logged in users will be able to post questions and answers.
- Logged in users will be able to vote answers up or down.

Models

- We will need to create these models:
 - Question
 - Answer
 - Vote
 - User

Associations



Creating the App

- rails new qanda
- cd qanda
- bundle install
- rake db:create

Adding Models

- rails generate scaffold Question body:text
- rails generate model answer body:text
question:references
- edit app/models/question.rb
- rake db:migrate

Setup Routes

- Edit config/routes.rb
- Answers are nested in Questions
 - Just like Comments and Posts
- root :to => "questions#index"
- Delete public/index.html

Update Views

- Use basically the same views as the blog
- Obviously, change post to question and comment to answer
- <https://github.com/anthonylewis/qanda>

Sessions

Web servers have terrible memory

Sessions

- The web is “stateless”
- We generate a session id for each user so the server can remember who they are.
- The session id is stored in a cookie that is passed to the server with each request.
- This sounds like a lot of work...

Devise

- <https://github.com/plataformatec/devise>
- A flexible Rails authentication system
- Takes care of user registration, log in, log off, password resets, etc.
- Devise is a Rails Engine - it has models, views and controllers

Installing Devise

- Edit Gemfile
 - `gem 'devise'`
- `bundle install`
- `rails generate devise:install`
- `rails generate devise user`

Configuring Devise

- Make sure you have a root route
- Make sure your views include flash messages (notice and alert)
- Customize the User model and migration for your application
- `rake db:migrate`

Devise Helpers

- `authenticate_user!`
- `user_signed_in?`
- `current_user`
- `user_session`

Update View

- Edit app/views/layouts/application.html.erb

```
<% if user_signed_in? %>
  <%= link_to "Sign out", destroy_user_session_path,
    :method => :delete %>
<% else %>
  <%= link_to "Sign in", new_user_session_path %> |
  <%= link_to "Sign up", new_user_registration_path %>
<% end %>
```

Update Controller

- Edit app/controllers/questions_controller.rb

```
before_filter :authenticate_user!,  
              :except => [:show, :index]
```

More Associations

Everything is connected

User Associations

- Questions belong to a User
- Answers belong to a User
- A User has many Questions and Answers

Adding Columns

- rails generate migration add_associations
- Edit the migration file
 - add_column :questions, :user_id, :integer
 - add_column :answers, :user_id, :integer
- rake db:migrate

More Models

- We have Questions, Answers, and Users
- Let's add Votes
- rails generate model vote vote:integer
answer:references user:references
- rake db:migrate

Vote Associations

- Votes belong to a User and an Answer
- A User has many votes
- An Answer has many votes

More Associations

- has_many :through
 - A user has many answered questions
 - A user has many voted answers

Has Many Through

- Edit app/models/user.rb

```
has_many :answered_questions,  
  :through => :answers, :source => :question
```

```
has_many :voted_answers,  
  :through => :votes, :source => :answer
```

Votes & Scores

Your vote counts

Vote Routes

- Create either an “up” or “down” vote
- Vote belongs to current user and answer
- The URL could look like this:
 - `/vote/up/:answer_id`
 - `/vote/down/:answer_id`

Vote Routes

- Edit config/routes.rb

```
match 'votes/up/:answer_id' => 'votes#up',  
  :via => :post, :as => :vote_up  
match 'votes/down/:answer_id' => 'votes#down',  
  :via => :post, :as => :vote_down
```

Vote Actions

- rails generate controller votes
- Edit app/controllers/votes_controller.rb

```
def up  
  create_vote( 1, params[:answer_id], current_user)  
end
```

```
def down  
  create_vote(-1, params[:answer_id], current_user)  
end
```

Vote Actions

```
def create_vote(vote, answer, user)
  @answer = Answer.find answer

  @vote = @answer.votes.create :user_id => user,
    :vote => vote

  redirect_to question_path(@answer.question)
end
```

Counting Votes

- Rather than total up the votes every time, let's add a score to each answer
- rails generate migration add_score_to_answers
- Edit the migration
- rake db:migrate

Updating Scores

- This is “business logic” and should be a method of the Answer model

```
def update_score
  self.score =
    Vote.where(:answer_id => self.id).sum('vote')

  self.save
end
```


Counting Votes

```
def create_vote(vote, answer, user)
  @answer = Answer.find answer

  @vote = @answer.votes.create :user_id => user,
    :vote => vote

  @answer.update_score
  redirect_to question_path(@answer.question)
end
```

Votes in the View

- Edit app/views/questions/show.html.erb

```
<p>Score: <%= answer.score %></p>
<% if user_signed_in? %>
  <%= button_to 'Vote Up', vote_up_path(answer) %>
  <%= button_to 'Vote Down', vote_down_path(answer) %>
<% end %>
```

Homework

Thought exercises

Work On The App

- Clone the app from GitHub
- Make the views a little prettier
 - Look at sites like Quora or Stack Overflow for inspiration
- Think about other features to add

