Chitter Challenge Steps

1. Added data\_mapper and dm-postgres-adapter to Gemfile
2. Added data mapper set up page
3. Added app file and required data mapper setup at the top
4. Required capybara/rspec in spec\_helper
5. Created two databases after running psql: chitter\_development and chitter\_test
6. Changed datamapper setup to include chitter\_development database
7. Create a User model.
8. Add three properties: ID, username and password.
9. Just required user model in data\_mapper\_setup
10. Move data\_mapper\_setup to app folder. Or the User table won’t show up in the terminal.
11. Had to make sure DataMapper setup was correct. Mine said “chitter\_development\_#{env}”, whereas it was just meant to be ‘chitter\_#{env}
12. Ran bundle. Still didn’t work.
13. Created spec/features folder where feature tests will be
14. Create ‘peep’ (link) class.
15. Write capybara test for viewing peeps (links)
16. Failed because it said ‘uninitialized constant Peep’ despite Peep class existing.
17. Solved. I didn’t have a require in the viewing peeps spec file, so added require for data\_mapper\_setup and required peep class in data\_mapper\_setup
18. Now have a failing test saying “ Rack test requires rack application”.
19. To make sure we tell Capybara what it should be testing, I added ‘Capybara.app = Chitter’ into the spec\_helper and required the server file in the spec\_helper
20. Had to require Sinatra in Gemfile
21. To help pass first capybara test, create a /peeps path in app file
22. Create an index.erb file that’ll be rendered in the controller.
23. To make test pass, I changed Peep.new to Peep.create
24. Added a creating Peeps spec file.
25. Wrote Capybara test for creating new peeps.
26. Error message says Unable to find field "peeps". Need to pass it
27. Defined peeps/new route in app file that renders a new\_peep erb file.
28. Create erb file for new\_peep
29. Test still failing despite server file being correct and erb file being correct. Error message saying ‘Unable to find field “peeps”.
30. Included a set :views, proc in controller to see if it makes a difference.
31. Didn’t really make a difference to error message
32. Changed button value to ‘Send Peep’ to be consistent with test.
33. Test STILL FAILING! Unable to find field ‘peeps’
34. Had to change test ‘fill\_in’ to message, whereas before it said ‘peeps’
35. Installed database cleaner. Bundled and placed in spec\_file and Gemfile
36. Test passes! The problem was I had Peep.new instead of Peep.create and this wasn’t saving the peep. I also had to include a config.ru file for rackup to run.
37. All is working fine now. 2 tests passing, and the Peeps are being shown in the page and stored each time even when session closes.
38. Problem: Every time I run Rspec it deletes everything from the development database.
39. Included ENV['RACK\_ENV'] = 'test' at the top of spec helper.
40. Start adding user accounts
41. Added test for signing up as new user.
42. Test failing because status code 404 is showing instead of 200.
43. To pass, add a new route in the controller for users/new page
44. Add new.erb file into views/user directory.
45. Make some changes so user is requested to give a username also
46. Added password\_digest property to user model as well as password method.
47. Enabled sessions and set encryption key (“session\_secret”)
48. Helper method added that returns an instance of User for the currently logged in user.
49. Test passing for signing up as new user
50. Added a test for matching password confirmation
51. Add dm-validations to dm setup
52. Adjusted post /users to accept password confirmation
53. Adjusted sign up form to have password confirmation
54. Test passed. Initially failed because the name in the form ‘Password confirmation’ did not match the tests, model or controller.
55. Updated password confirmation test so it doesn’t redirect to links page if wrong password is given. We need to not redirect the user is wrong password confirmation is given, so adjust controller to display same form again with an error message.
56. Installed Sinatra flash into Gemfile.
57. Registered Sinatra flash at the top of controller.
58. Included flash notice in layout
59. Test now passes.
60. Make user an instance variable in the controller so that information gets saved and user doesn’t have to fill out all forms again.
61. View now expects @user instance variable, so need to make sure it’s available in the /users/new get route. Placed @user = User.new into get request now all is fine
62. Wrote test for requiring an email address. Not passing as it says ‘wrong number of arguments (1 for 0)
63. Made changes to sign up method so it takes user as an argument
64. Had to change all the fill in properties in the test. So username became user.username. Password became user.password.
65. Test passes. Had to include validates presence of email in user model and change the tests so the sign up method takes one parameter (user).
66. Included Factorygirl gem.
67. Changed sign up as new user test to have user = create :user and saved user information in factory girl user.rb file.
68. Was stuck, test wasn’t passing, said “Failure/error: user = create :user

ArgumentError:

Factory not registered: user.

1. Solution: Require ‘./spec/factories/user in spec helper
2. Write test for cannot sign up with existing email
3. Include validates\_uniqueness\_of email in user model.
4. Test still not passing. Need to include text on page also.
5. Validates uniqueness of email made the ‘I can sign up as a new user’ test fail. Initially I created a new user, which means it saves it. Then I changed it to User.new and it passed. **WHY?**
6. included a flash.now[:errors] in controller. Also edited layout.erb to iterate over flash error messages.
7. Had to change password test as the custom message did not match the message we passed into the test.