We worked on creating a library checkout type app. Here is the list of the commands that I ran (minus some unimportant ones that I deleted to save space) comments on commands will be in red.

501 mkdir libraryTestApp

502 cd libraryTestApp/

509 git init

510 git commit -m "initial commmit"

511 git remote add origin https://github.com/mpanoratra/libraryTestApp.git

512 rails new ./ -d postgresql

513 git add .

514 git commit -m "first commit"

515 git status

516 git push -u origin master

524 rake db:create

523 rails s

Check to make sure everything is installed and running correctly at this point

525 vim config/database.yml

Added the line to use the postgres default user: postgres

526 rake db:create

527 git add .

528 git commit -m "fix database default user"

529 git push

517 vim Gemfile

Add gem ‘devise’ to gem file

518 bundle install

519 rails g devise:install

520 rails g devise:views

521 rails generate devise User username:string

516 rake db:migrate

517 rails g scaffold Book name:string genre\_id:integer

This generates the scaffold for books… but we didnt put in the author id… why? so I could show you how to add columns to a model later. Hence the next command.

518 rails g migration AddAuthorIdToBook author\_id:integer

Add a column to a model. You can do this at any time to add data to a model. You may have to update views and controllers to show this data and to accept it as a parameter. If you need to look up how these migrations work go here: [Migration Syntax](http://guides.rubyonrails.org/migrations.html)

For more info about the parameter thing go here :[Strong parameters](http://edgeguides.rubyonrails.org/action_controller_overview.html) look in section 4.5 I think

519 vim db/migrate/20140615190358\_add\_author\_id\_to\_book.rb

Do this to check the syntax to make sure its correct… when we did it I made the mistake of using number instead of integer… just watch out.

520 rake db:migrate

524 git add .

525 git commit -m "added devise users and books"

566 rails g scaffold Genre name:string

567 rails g scaffold Author name:string

These make the controllers and views and models for genre and author. You can check them out by going to the respective urls. localhost:3000/genres etc..

568 cd app/assets/stylesheets/

569 ls

570 rm books.css.scss authors.css.scss genres.css.scss

571 ls

I don’t like these stylesheet, to me it seems like they just get in the way. You can choose to delete them or not. Rails also creates javascript files, sometimes these can be useful if you have behavior that you only want to tie to that object… really depends. This is optional for now.

572 cd ../../..

575 cd app/models/

576 ls

577 vim book.rb

I also opened the other models and created the relations between them. for a really good explanation of how this works and what the syntax is see: [Active Reco rd Relations](http://guides.rubyonrails.org/association_basics.html)

579 rails c

I added some test data here to check and make sure that the models are working. I added this to a seeds file later. I added the gem seed\_dump to create the seeds file for this so that I could share the data with you guys later. One thing to note is that the seeds file SHOULD NOT contain confidential information as it will be pushed to github.

581 rake db:migrate

584 rake db:seed:dump

Need seed\_dump gem first. add it to gemfile and bundle install if you dont have it. Or locally install it with gem install seed\_dump

588 cd views/books/

591 ls

592 vim \_form.html.erb

First I added the fields for author id since that was not generated along with the scaffold. easiest way at this point is to just copy what is there for genre.

594 rails s

Visually confirm that the form has been changed. You might have to try this a few times if you get syntax errors or other issues.

595 vim index.html.erb

Add views for the data that we just started accepting

596 ls

597 vim show.html.erb

Add views for the data that we just started accepting. I also made an unless check that if the book doesn’t have that data it won’t show. If you don’t have this check you may get errors because active record is trying to look up an author or genre with nil data. You can use the unless check like I did or set up the database with default data and constraints so the field is never nil. really depends on the implementation you want.

599 vim index.html.erb

Added unless checks here too.

607 vim db/seeds.rb

607 vim ../../controllers/books\_controller.rb

Here I went down to the private methods and added the author\_id to the list of permitted parameters. When I demoed what was going on prior to this the app was not saving the author\_id.

592 vim \_form.html.erb

Here I updated the form with collection selects instead of letting the user input any random integer. Makes the selection a lot cleaner.

608 git status

609 git add .

610 git commit -m "books form updated with author and genre...

Also dumped to seeds.rb"

611 git push

Summary. So far we have created the models for books, author and genre. A user can create update, read and delete each of these via the default scaffold set up.

TODO: still a lot of functionality left. create user stories for Sean and Brandon to follow.