

## Laboratory 8: Rails Dash

At first, we created a new directory for our Ruby project. Then we checked if we have ruby and rails on our machine. We've got Rails 5.2.1 and ruby 2.3.3.296.

Then we tried to imitate the steps from the lab. As we tried 'rvm list', it didn't work and we realized that we've not have rvm installed. Then we started googling to fix that. But despite with the help of the professor we didn't figure out a way to install rvm on Windows. Then somebody noticed that gem is installed, so we used this to update our gems. Then we continued the imitate. We used 'gem install bundle' and it successfully installed bundle-0.0.1. Then we used 'gem install rails' and it successfully installed rails-5.2.2, so we updated it from 5.2.1 to 5.2.2.

Now we used 'rails new vpw' to set up our project. Then we navigated to the config folder and used 'rails server' to set up our server. As we tried to open the webpage with '0.0.0.0:3000' and 'tcp://0.0.0.0:3000' it didn't work, but with 'localhost:3000'. YAY!

After that we created a new database, which represents our Ingredients. In the lab report, we decided that we have Ingredients and possibleIngredients as class. While doing the Ingredients database in ruby, we decided, that we don't need the possibleIngredient class.

### Ingredients

Name	Price	Maxusage	Available	
Cheeeeeeeeeese	10	100	true	<a href="#">Show</a> <a href="#">Edit</a> <a href="#">Destroy</a>
Corn	15	8	true	<a href="#">Show</a> <a href="#">Edit</a> <a href="#">Destroy</a>
<a href="#">New Ingredient</a>				

We just kept track with a boolean, if the Ingredient is available.

We successfully set up our database and tested it with the server.

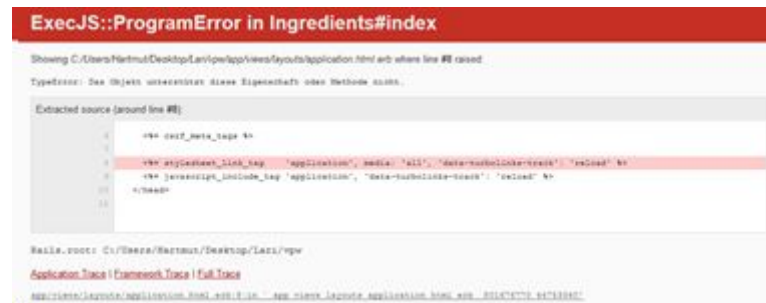
In the next lab session, we wanted to create the other databases. We started with the database for the Hoagie. We created it the same way we did the Ingredient database. But we had some trouble creating this database. We named it hoagie, this caused some errors as ruby decided that the plural is hoagies but the singular hoagy. So, in some files it was called hoagie, like we declared it, and in some others hoagy. This caused the error message „param is missing or the value is empty“. We googled a bit around but couldn't fix it. Then we had the idea with the naming problem. We created a new scaffold hoagy and the problem was gone.

Then we tried to connect the ingredient and hoagie model, so that the user can choose ingredients when creating a Hoagie. We tried some tutorials but didn't get it to work.

After that we started working from home. We decided to use git for that, so that we can keep track of the different versions and can go back if we should break it.

We pulled the project and tried to run it on a laptop. As it has different versions of Ruby and Rails it didn't work. After a long time of trying to fix that. We decided to start over again.

So, we created a new project. We created a new database and immediately got an error message. After some googling we were able to fix that. We used the fixing description of this website



```
ExecJS::ProgramError in Ingredients#index

Showing C:/Users/Hartmut/Desktop/Larissa/app/views/layouts/application.html.erb where line #8 raised:

TypeError: Das Objekt unterstützt diese Eigenschaft oder Methode nicht.

Extracted source (around line #8):

  <%= next_page_tag %>
  <%= stylesheet_link_tag "application", media: "all", "data-turbolinks-track" => "reload" %>
  <%= javascript_include_tag "application", "data-turbolinks-track" => "reload" %>
</head>

Rails root: C:/Users/Hartmut/Desktop/Larissa/vgr
Application Trace | Framework Trace | Full Trace
app/assets/javascripts/application.html.erb:8:in `app_views_layouts_application_html_erb_8114777c_b47338d'
```

(<https://code.i-harness.com/de/q/1b0038c>).

After we finally created the new project we retried to connect hoagie and ingredient.

We tried a lot of different tutorials, also Andromachi recommended us a good tutorial. In the end we found one understandable tutorial

(<https://www.youtube.com/watch?v=ZNrNGTe2Zqk>). We followed the steps and adapted it to our Hoagie Shop. We created a model ingredient\_hoagie to connect hoagie and ingredient. Also we connected the two databases using belong\_to and has\_many. Then we adapted the html file, to create a dropdown menu. We tried it and it didn't work at all. We got a lot of error messages. After fixing one we got a new one. After finally fixing hundreds of errors the dropdown menu was displayed.

## New Hoagy

Name

All ingredients  

Cheese

Salami

But then new errors occurred, when creating the new hoagie, we tried a lot of fixing, but couldn't get it to work.

We decided to move on and created our Hoafie (a mix between selfie and Hoagie) database. We used the tutorial, which was shown in the lecture(<https://guides.railsirls.com/app>). We created the scaffold for the hoafie. To

upload a picture, we add the gem 'carrierwave' in the Gemfile. Then we ran bundle. After that we generated the picture uploader.

In the ruby file of hoafie we added the PictureUploader. Also we changed the line <%=

form.text\_field :picture, id: :idea\_picture %> to <%= form.file\_field :picture, id: :idea\_picture %>, so that we can upload a file. To show the picture on the page we changed the html page. We tested it and it worked!

## Hoafies

Picture	Comment	Username	Verified	Likes	Hoafie of the week			
	so yummy! Best Hoagie ever!!	lari	true	9	false	<a href="#">Show</a>	<a href="#">Edit</a>	<a href="#">Destroy</a>
	best htw shop! so yummy	doro	true	17	false	<a href="#">Show</a>	<a href="#">Edit</a>	<a href="#">Destroy</a>

[New Hoafy](#)

To make our hoagie shop a little bit beautiful we changed the css-file.

We added some borders and paddings so that the table looks more structured. Also, we changed the colour and underlined links. We changed the table header and inserted blanks.

Furthermore, we used two general stylesheets, suggested in the railsgirls-tutorial. We reloaded the pages and it looked much better.

Then we tried to create a new start page, for that we used the tutorial ([https://guides.rubyonrails.org/getting\\_started.html](https://guides.rubyonrails.org/getting_started.html)).

At first we created a new controller. To set the new homepage we changed the file routes.rb in the config file. We specified that our application should root to our new webpage. We tested it and it worked.

Our code is on <https://github.com/LaWa15/HoagieShop.git>.

Reflection:

Larissa:

It was really frustrating to work with Ruby on Rails. When I managed to fix something the next error occurred. I think for starters it's really hard, but when working more with it, it's a really fast way to set up an online shop or other homepages.

But also it was nice to see that we had a clear idea how our hoagie shop should work, even if we couldn't quite get it to work.

I spend around 13 hours to work on this lab and googling around for fixes.

I want to give credits to Andromachi, which recommended some tutorials to me.

Chantal: Time spent 5 hours.

It was really exhausting and frustrating to do this lab. I think I didn't understand much of ruby on rails and was not quite a help. Like David, I couldn't manage to really do something at home. I spent at least 2,5 hours with installing and similar to get ruby on rails on my system and that was exhausting too. It is sad that we couldn't manage to create the webshop, because until this lab it was really fun to plan this and discuss it. Together in the lab it was kind of easier to achieve something but because of the exam phase we had to work alone and that was kind of frustrating.

David: Time spent 7 hours

Actually, only Larissa managed to get something going. I got stuck with running this thing and fixing errors. I also had some difficulty understanding Ruby on rails and fixing errors. I'm really disappointed that I was not a big help and that we couldn't manage to create our shop how we wanted to. I just hope I don't have to work with it in the future. So Larissa definitely deserves more than one beer here.