**Project Guideline**

## Working on your project

We see that having at least 3 strong projects and a strong portfolio site is required for those looking to make a career switch. Each of your projects must be more comprehensive than the assignments you've done.  Here are things to remember.

## Before you work on your project

Please remember to get your belt first.  Knowing the materials in the platform are critical to building any project.  You must first get the knowledge and then apply the knowledge by building your project.

## What Project to Work on

1. Please either work on the e-commerce system, User Dashboard, or employee clock in/out system.
2. If you have another project idea to work on, please sketch it out on the whiteboard/paper, use Balsamiq or Powerpoint to layout how your website would look like and get your instructor to approve your wireframe no later than 24 hours after you've started working on your project.

* E-Commerce System: [eCommerce.pdf](https://s3.amazonaws.com/General_V88/boomyeah2015/codingdojo/curriculum/content/chapter/eCommerce.pdf)
* EPS (employee clock in/out management system): [epswireframe.pdf](https://s3.amazonaws.com/General_V88/boomyeah2015/codingdojo/curriculum/content/chapter/epswireframe.pdf)
* User Dashboard: [user-dashboard.pdf](http://s3.amazonaws.com/General_V88/boomyeah/company_209/chapter_3834/handouts/chapter3834_6841_user-dashboard.pdf)

## While Working on the Project

Please do NOT build any of the backend features immediately.  Please create a folder where you create a bunch of html files and a static folder containing all of your css files, image files, and javascript files.  This should be very similar to how you've created html and css pages during your Web Fundamentals.  Once you have this all of the pages built (which is essentially your clickable prototype), then create ERD, and then add the backend codes using Django or Flask.

At least 30-50% of your project time should be spent on building this clickable prototype and building out the ERD.

Please use Bootstrap or Foundation to make it look professional.  If you're interested in front end, it's a great idea to make your site responsive (at least the first few pages).

## Towards the end of your project week

You may not finish all the features (it always takes longer to build than what everyone first imagines).  Please do not get discouraged as you can finish the features later on.  Do look at this as an opportunity to apply and combine all of the concepts you've learned and do reflect back to recognize how much you've learned and grown over the last few weeks.

Please do present your project at the end of the project week and also show your ERD.  Even if you don't have all the features built, you can always present your clickable prototype.

# Belt Reviewer

Before you work on the belt reviewer assignment below, please make sure you are able to address the following questions.  If you're on-site, it's strongly encouraged that you go through these list of questions with someone else in your cohort.

## Conceptual Questions

Make sure you can answer the following questions (some of these questions may be asked during the belt exam):

1. What is the flow of information in a typical request, from when we type an address on our browser, to when we receive the response on our browser?
2. What is MVC, OOP and procedural programming? Why would we use each?
3. What is jQuery and why do we use it?
4. Consider a database with the following tables: users, friendships, where users includes id, name, and email, and friendships includes id, user\_id, friend\_id; If our friendships table includes a single association to confirm a friendship between two users, what is the QUERY to pull all of your non-friends?
5. What are some ways to make your website uniform across multiple browsers?
6. What are the differences between submitting a form via method="post" vs method="get"?
7. What are the advantages/disadvantages of sending data to the server in the url vs making a post request?
8. Why should we never render a page on a post request.
9. You notice that when you click submit on a form, your app breaks. Describe how you would approach debugging this problem.
10. What are a couple security threats and how do we defend against them?
11. What is the difference between Session, Post, and Cookies?
12. Explain why and when we use session.
13. What is an ORM and why do we use it?  What are its advantages and also its disadvantages.
14. In your models, you may make a class User with the following code: class User(models.Model): Why do we have models.Model inside the parentheses?
15. In your models, you may make a class UserManager with the following code: class UserManager(models.Manager): Why do we have models.Manager inside the parentheses?
16. What is self and why does it appear in methods that we define in a class?
17. What is the \_\_init\_\_ method in a class?
18. Name 2 HTTP Verbs and when would we use each
19. What is RegEx and what do we use it for?
20. On large web applications, what are the benefits of using a framework?
21. What does a templating engine do for us?
22. What are 3 different type of database table relationships?
23. Why do we use routes and how do they work?
24. What is a virtual environment and why do we use it?
25. Explain what this code does: “from flask import Flask”.
26. What is the importance of normalizing your database?
27. What is an API?
28. What is AJAX and why do we use it?
29. What is the difference between an HTTP request/response and an AJAX request/response?
30. What is the difference between client and server validation, and when do we use either or both?
31. What are the major differences between Flask and Django?
32. (VB) What are differences between tuples, lists, and dictionaries.