*1 Instructions*

*Please submit your final project report via Sakai. Your codebase should reside on github, and*

*be sure to include a link to your github repo in your submission.*

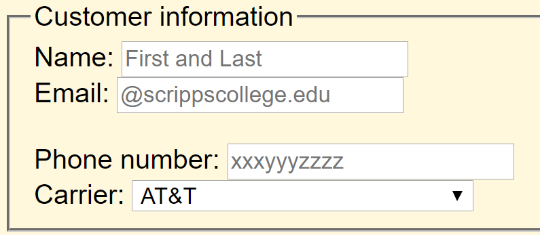
*You final project report should contain the following sections:*

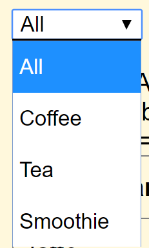
**1.1 Introduction**

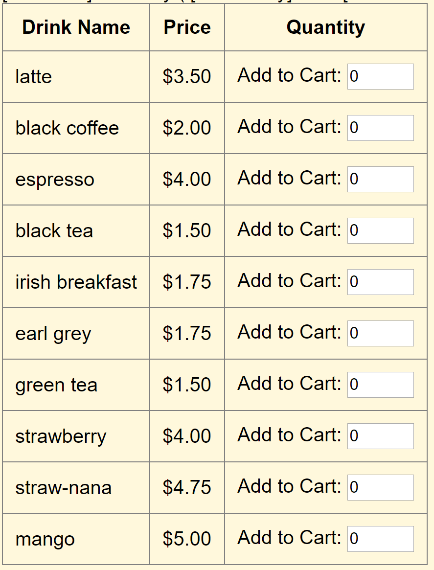
Introduce and motivate your application. Describe the core functionality of the application

(at the level of a user guide). While screenshots are not required, we often find that the app

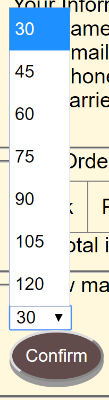
flow and page functionality makes more sense with screenshots.

Our idea started with a passion for coffee, more specifically a way to get coffee more efficiently. Inspired by Starbucks’ mobile app and the girl scout homework, we decided to create a web order form for the Motley coffeehouse. A customer simply pulls up the site, and will fill in the fields with their information (name, phone #, email and carrier):

Following this, they will select which drink they want to order. The display can be changed to make navigation of the menu easier. All the drink options are displayed per default, but the dropdown navigation allows the user to narrow down their selection with 1 of 3 criterion: coffee, smoothie or tea.



The customer selects his or her drink by entering (or incrementing) the quantity of the drink. They can order a maximum of 5 of any drink.

After he or she has made her selection, the customer clicks the “go to cart” button, which takes them to the checkout page to confirm their order. On this page, their information is displayed, their drink, quantity and price as well as the total quantity of drinks ordered and the total price. The user selects the time which they would like to pick up their drink(s) in (with a min of 30 minutes) and then ‘confirms’ the order.

After the order has been confirmed. The user is redirected to a page that simply says ‘thank you for your order’ and is emailed a response to confirm their order.

**1.2 Software Design**

Provide a high-level overview of your software architecture. Did you implement any

Again, while diagrams are not required, they are highly encouraged. What were the key

ideas/concepts in web architecture/programming that we studied this semester that you used

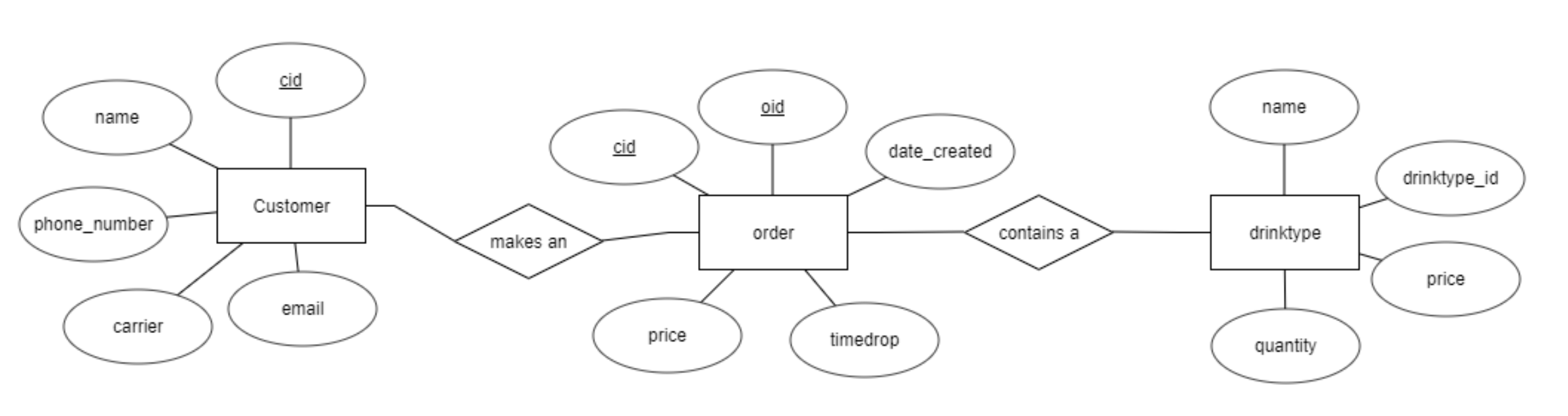
as part of your final project.

This section should also include a discussion of the server application and the database. You

must include an ER diagram describing the database design. Consider utilizing free software

like (https://erdplus.com/#/standalone) to generate the diagrams.

We worked to implement the model-view-controller method with our project. The view holds the pages that the user sees: the order page, and the confirmation/checkout page. On the submission of the form of the order page the post array is received by the invoke() function in the controller. This holds the customer information and the drink order. This information is passed to the model in its relevant function (relating to customer or drink order) and then inputted to our database, “motley”. The page is directed then to the checkout page which is a form when submitted holds the time until pickup and the current time upon submission.



**1.3 Contributions**

Discuss the contribution of each team member.

Lindsey:

**1.4 Future Work**

Future Work: Describe in detail limitations of your system and recommendations for improvements

or future work.

**Limitations** : Although we all strive the get the architectural aspects of our coding right

the first time, it is rarely the case that everything goes according to plan. Describe flaws in your

architecture and design. Some examples: Does your code follow good design principles and

practices (e.g. MVC, high cohesion-low coupling / single responsibility principle / separation

of concerns, test-driven-design)? If not, what are the major issues, and how could these be

fixed?

-there are some areas that if updated, would need to be updated in other areas as well…

-issues with the database that were not resolved until close to the due date resulted in the table needing to be hardcoded in

-put more interaction between controller and model, had to hardcode the drink list and access it from the view which in a true MVC pattern would not happen.

**Future work** : If you had more time on this project, describe what you would do next –

this could include refactoring and new features.

* Implement the table pulling from the database
* The customer being able to edit the order before confirmation
* The employee side implemented

**1.5 Reflection**

If you could go back and start over, describe what you would do differently, if anything. This

could include. If you have lots of advice, try to limit yourself to the top one or two things

that you believe would have most improved your development.

-start on the small stuff first

-had trouble figuring out how to split up the work

-a lot of design work before the underground was finished

**1.6 Libraries / Themes / etc.**

List any external libraries (Angular, Flask, FireBase, Google Maps API), CSS themes or

templates, etc. utilized in the final project.

*2 Submission*

*Please submit the project writeup/report via Sakai and be sure that the most up-to-date*

*codebase is on github. The codebase should be complete and should include a README file*

*that provides instructions on how I can deploy your codebase on my machine. Be sure to also*

*include any needed .sql files that will create the DB relations required by your application*