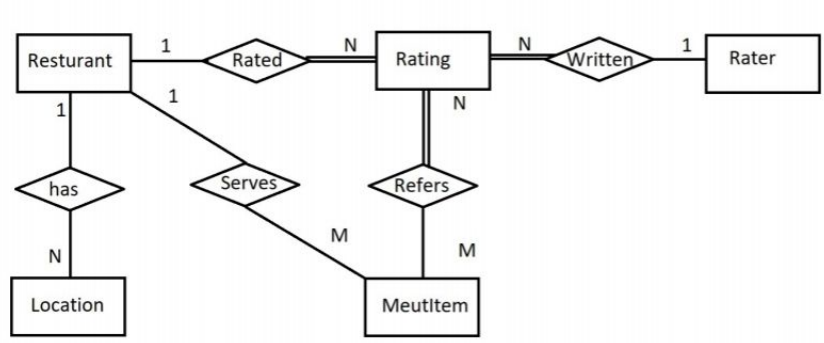
Todo

- Build and fill db
- Build APIs in java
- Build React front-end

Database



Rater

Userid	String	Prim
email	String	
join_date	Number	Timestamp
type	String	
username	String	
password	String	
Reputation	Int	1 to 5

Rating

UserID	String	Foreign key
Date	Number	Timestamp
Food	Number	1 to 5
Mood	Number	1 to 5

Price	Number	
Staff	Number	1 to 5
Comments	String	
RestaurantID	String	Foreign Key

Restaurant

RestaurantID	String	
Name	String	
Type	String	
URL	String	

Location

LocationID	String	Primary Key
open_date	String	
Manager_name	String	
Phone_number	String	
address	String	
opening_time	String	
closing_time	String	
RestaurantID	String	Foreign Key

Menu Item

ItemID	String	
Name	String	
Type	String	
Category	String	
Description	String	

Price	String	
RestaurantID	String	Foreign key

Rating Item

UserID	String	
Date	Number	Timestamp
ItemID	String	
Rating	Number	1 to 5
comment	String	

RatingVotes

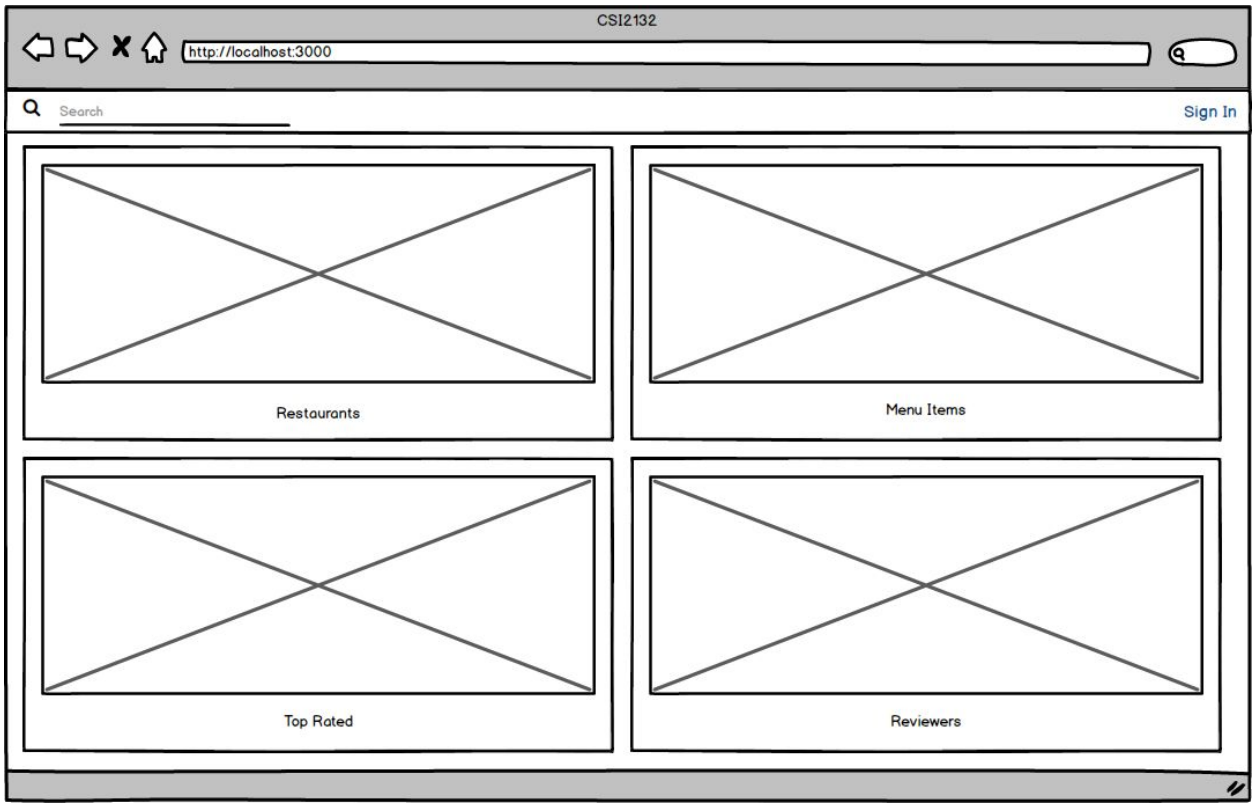
UserID	String	Foreign key
RaterID	Number	Foreign key
RestaurantID	String	Foreign key
Date	Number	Timestamp
Type	String	{ UP, DOWN }

React Packages

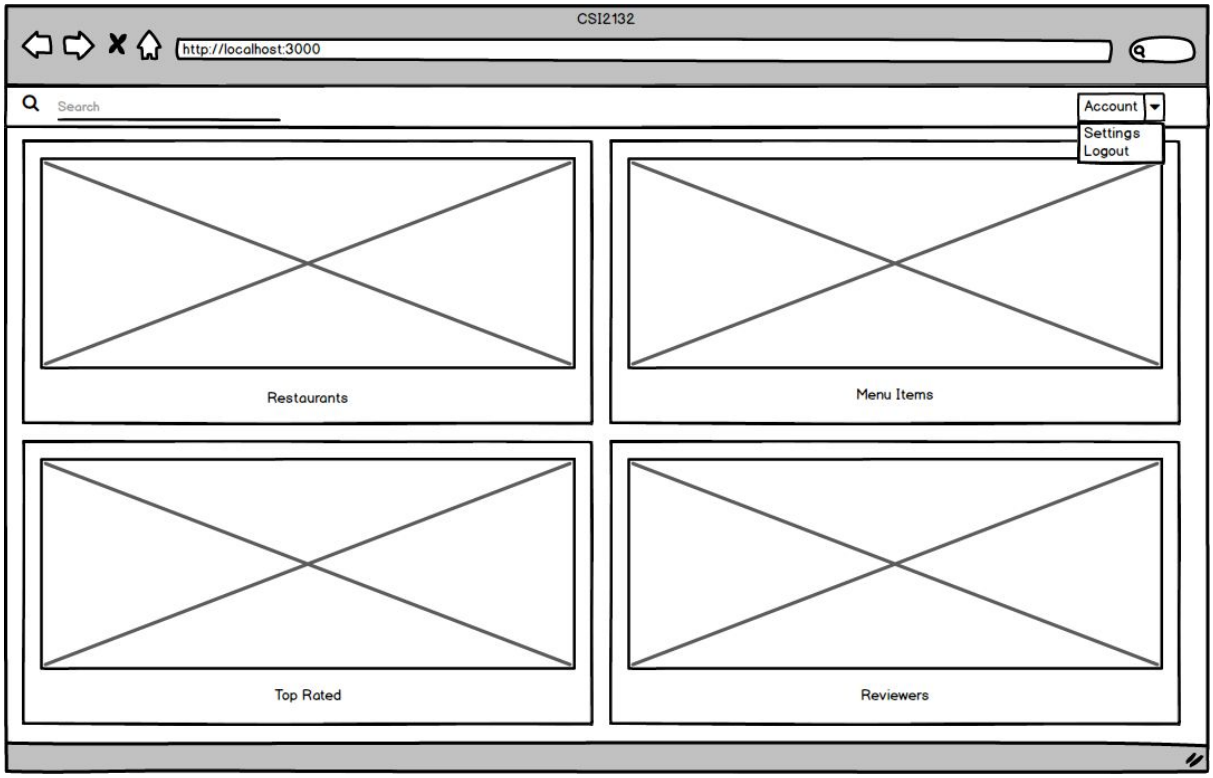
For each of these, navigate to the project folder in a command line and type:
npm install --save <list item>

- react
- redux
- react-redux
- redux-logger
- redux-thunk
- redux-promise-middleware
- redux-promise-axios
- axios
- material-ui
- lodash
- react-material-icons

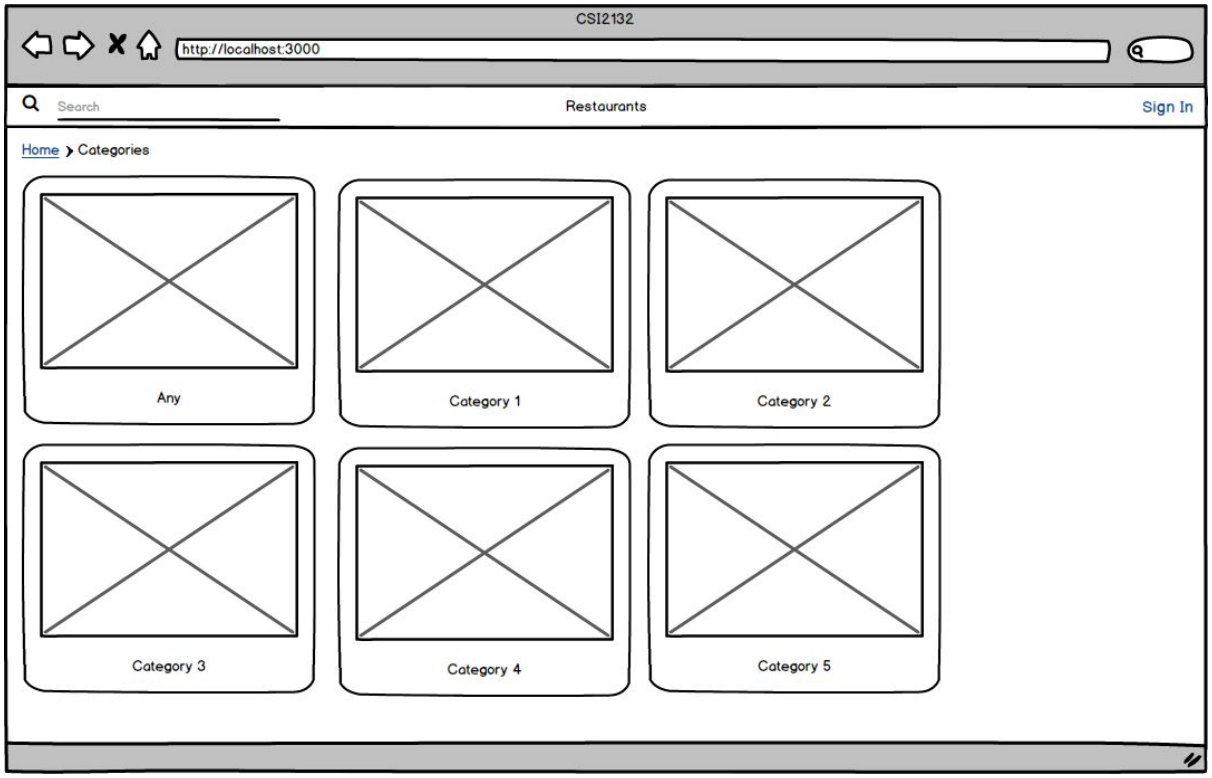
UI


CSI2132
http://localhost:3000
Search
Sign In
Restaurants
Menu Items
Top Rated
Reviewers

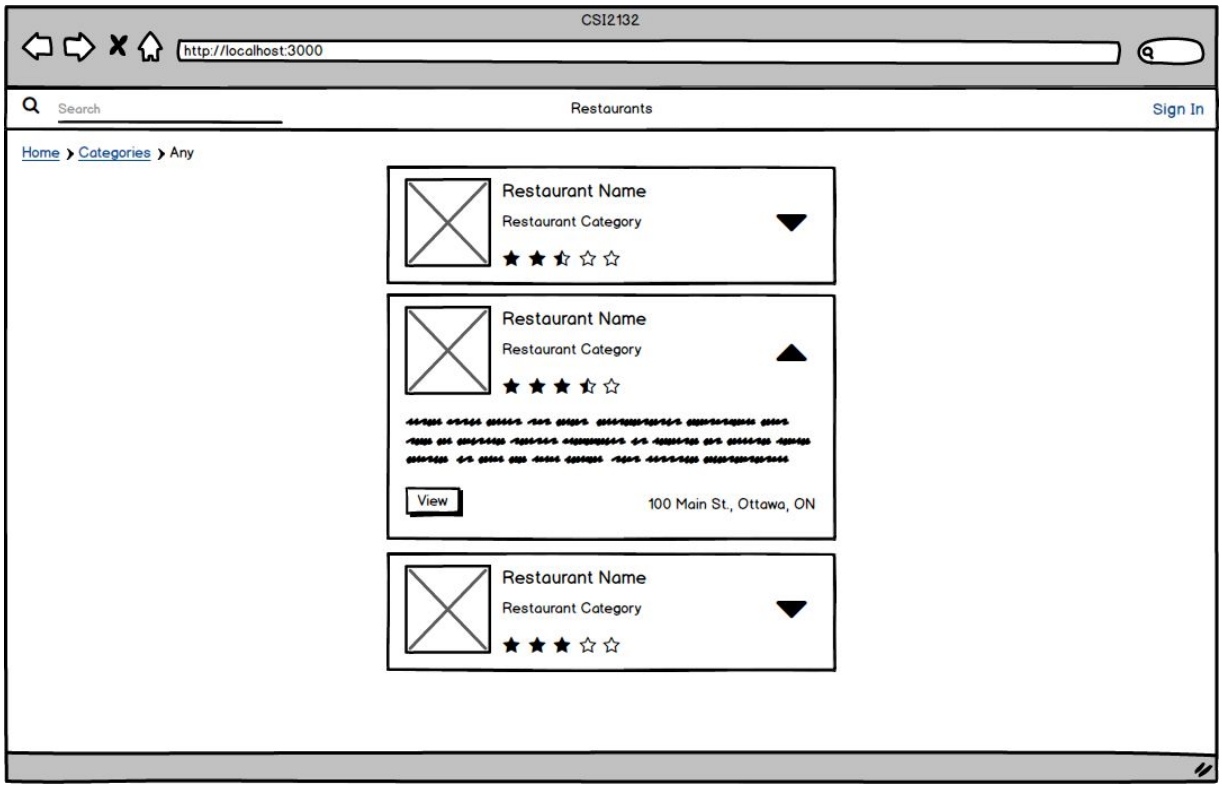
Home Page



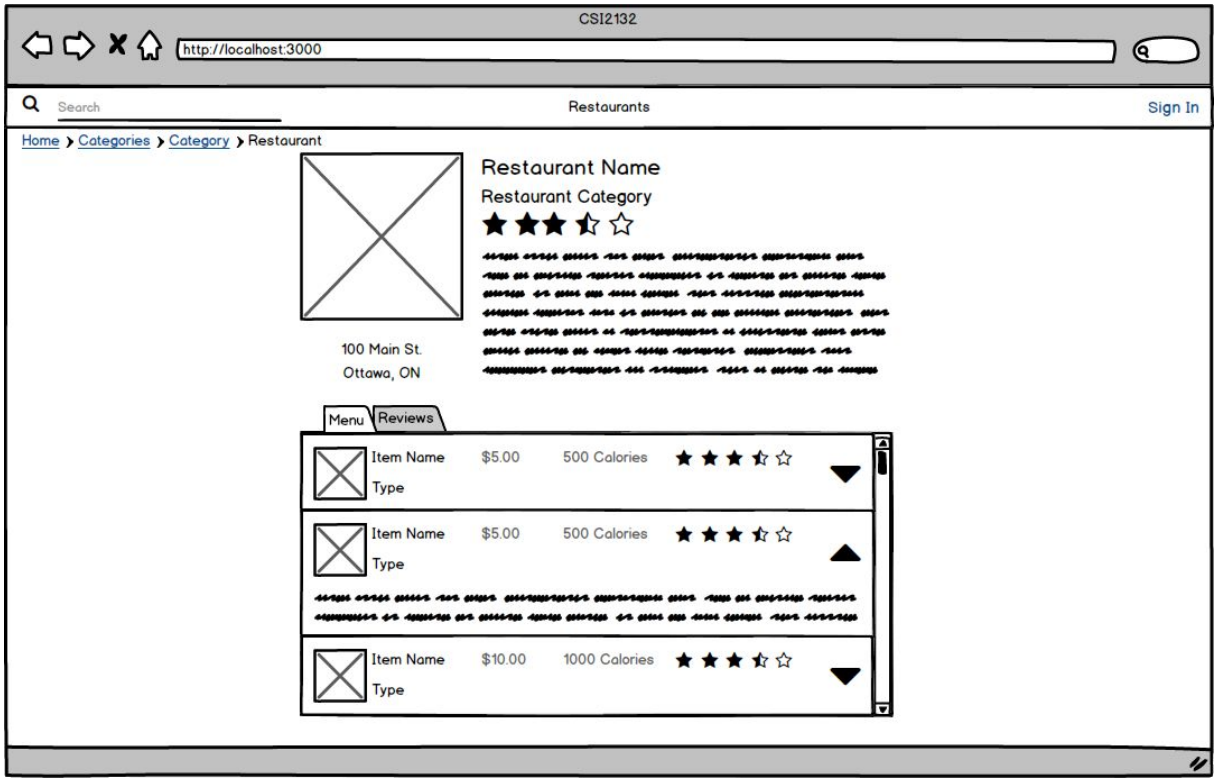
Home Page (Signed in to show nav bar)



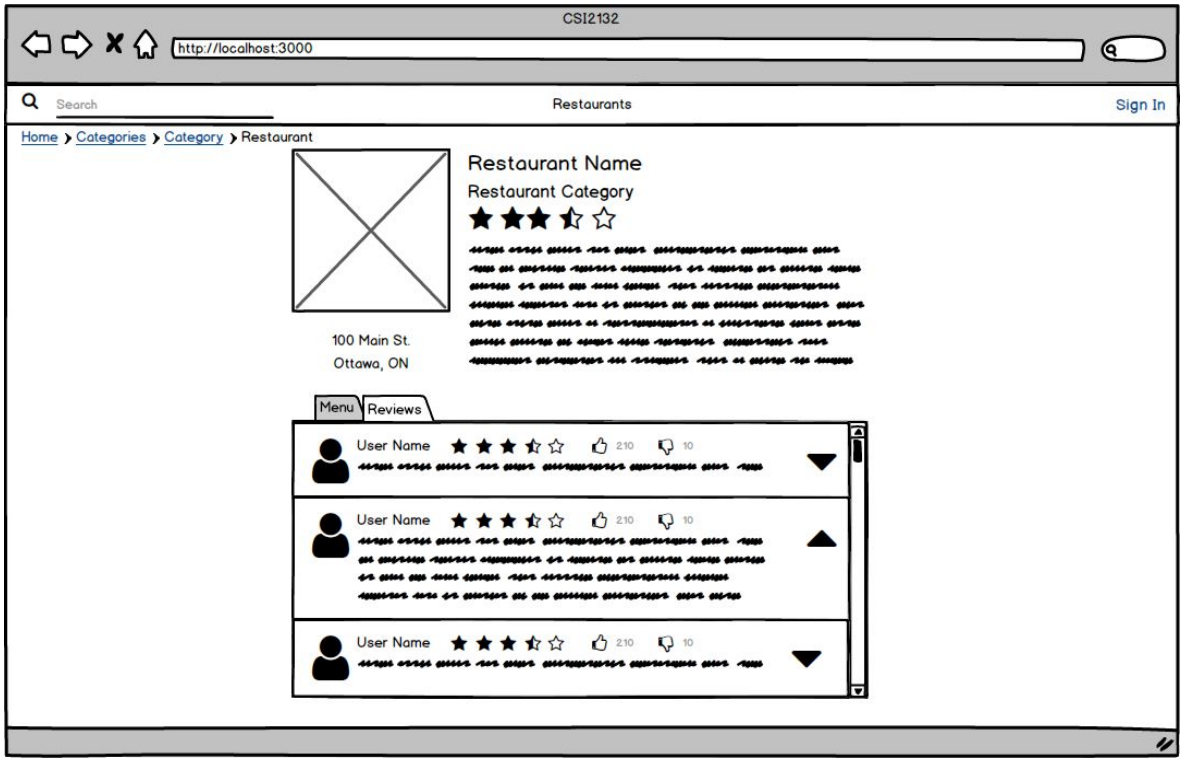
After Selecting Restaurants or Menu Items



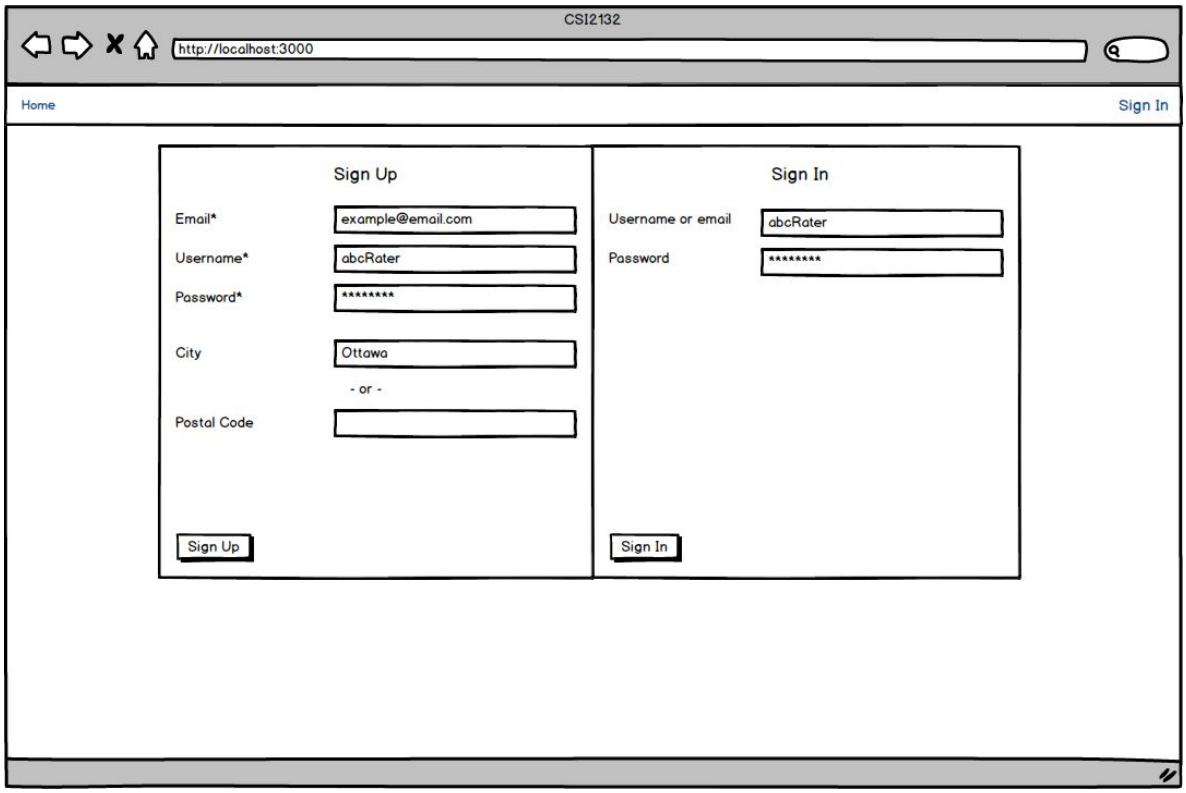
After Selecting a Category



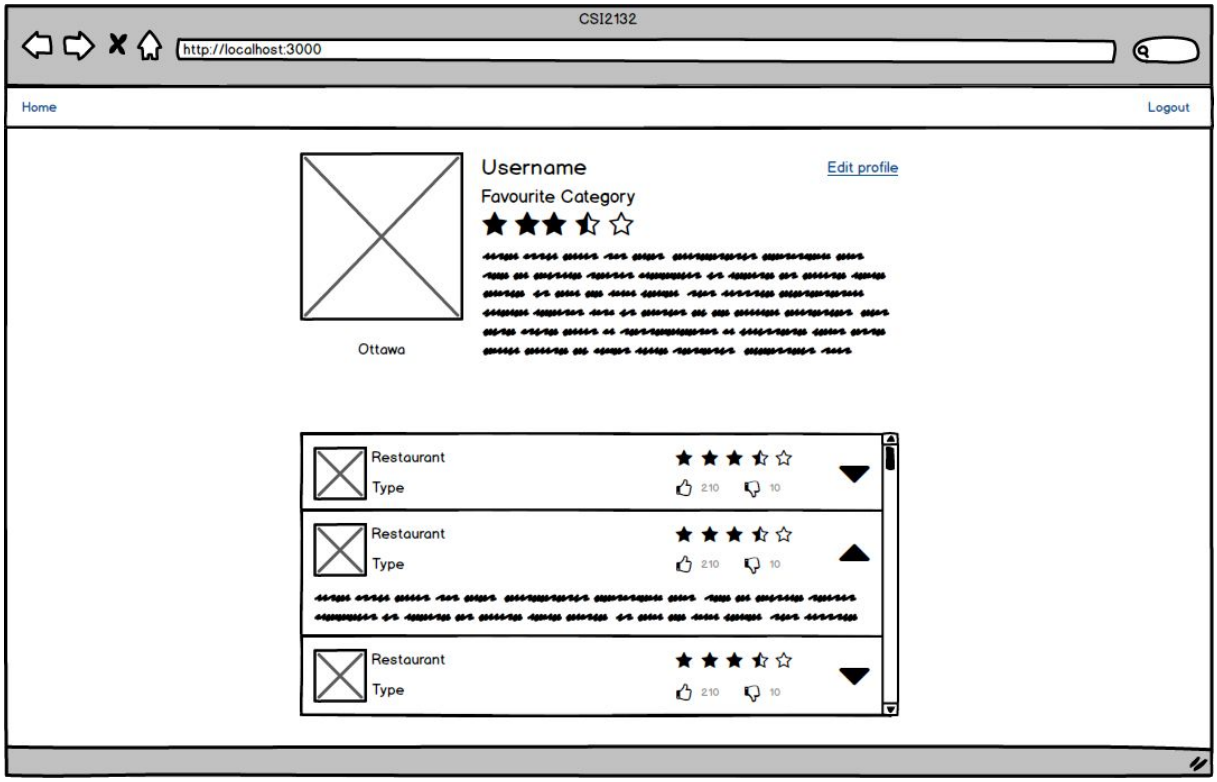
Restaurant / Menu Item Full View (Menu Item will not have tabs)



Restaurant / Menu Item Full View with “Reviews” selected



Sign In / Sign Up Page



Profile Page (To view other users and modify own settings)

API Calls

GET

/RestaurantAPI/rest/restaurant/get		Review Checked
/RestaurantAPI/rest/restaurant/get/<rid>		Review Checked
/RestaurantAPI/rest/category/get/<category>/restaurant		Review Checked
Return	RestaurantID	String
	Name	String
	Type	ENUM: (We need a list of types)
	URL	String
	Rating	Number: The sum of all ratings (across all categories i.e. Price, Food, Mood, Staff) divided by all non-zero ratings (a

		zero rating in any category should not be counted)	
	Locations	LocationID	String
		First-open-date	String
		Manager-name	String
		Phone-number	String
		Street-address	String
		Hour-open	String
		Hour-close	String

/RestaurantAPI/rest/restaurant/get/<rid>/menuItem			Review Checked
/RestaurantAPI/rest/restaurant/get/<rid>/menuItem/<mid>			Review Checked
/RestaurantAPI/rest/category/get/<type>/menuItem			Review Checked
Return	ItemID	String	
	Name	String	
	Type	ENUM: { Food, Beverage }	
	Category	ENUM: { Starter, Main, Dessert, Side }	
	Description	String: Descriptive statements	
	Price	Number: Dollars	
	RestaurantID	String	

/RestaurantAPI/rest/restaurant/get/<rid>/rating			Review
/RestaurantAPI/rest/restaurant/get/<rid>/rating/<uid>/<date>			Review
Return	UserID	String	

	Date	Number: Unix timestamp (could be string)
	Mood	Number: 1 to 5 (or zero for no rating)
	Staff	Number: 1 to 5 (or zero for no rating)
	MenuItemID	String: (this will be gotten from the ratingItems table) An empty string would indicate no menu item rating, meaning the below two rows aren't needed.
	Food	Number: 1 to 5 (or zero for no rating). Gotten from the ratingItems table
	ItemComment	String: from ratingItems table
	Comment	String
	RestaurantID	String

/RestaurantAPI/rest/restaurant/get/<rid>/menuItem/<mid>/rating		New
/RestaurantAPI/rest/restaurant/get/<rid>/menuItem/<mid>/rating /<userId>/<date>		New
Return	UserID	String
	Date	Number: Unix timestamp (could be string)
	ItemID	String
	Rating	Number: 1 to 5 (or zero for no rating). Will be "Food" rating for restaurant
	Comment	String

NOT NEEDED IF WE ENUM TYPE. I can just hard-code them. I think that would work out better anyway.

/RestaurantAPI/rest/category/get/menuItemType		Review
/RestaurantAPI/rest/category/get/restaurantType		Review
Return	Type	String

/RestaurantAPI/rest/rater/get	Review

/RestaurantAPI/rest/rater/get/<uid>		Review
Return	UserID	String
	Email	String
	Join_Date	Number: Unix timestamp (could be string)
	Type	String
	Reputation	Number: ceiling((Upvotes/Total Votes)*5)
	Username	String

All rating reviews for a restaurant. It is easier for me to just make 1 call and sort it out on my end
The returned json has a list called ‘ratingreviews’

/RestaurantAPI/rest/restaurant/get/<rid>/ratingReview		New
Return	UserID	String: ID of the rater
	Type	ENUM: { UP, DOWN }

Pass a json with ‘username’ and ‘password’

/RestaurantAPI/rest/rater/Login		New
Return	loginSuccess	String: true or false

We will need a way to send data. Tomcat by default is readonly. That is why PUT, POST and DEL are disabled. We can have delete as a GET, but update and create will have to be either a PUT or a POST.

The convention is:
GET: Read
PUT: Update
POST: Create
DEL: Delete

List of APIS
ADD/UPDATE/DELETE Restaurant
ADD/UPDATE/DELETE Menu Item
ADD/UPDATE/DELETE Rating
ADD/UPDATE/DELETE Rating Item

ADD/UPDATE/DELETE Rater
ADD/UPDATE/DELETE Location

Should not be able to update any keys. Other attributes will be named the same as the database attributes. Foreign key attributes are required to create. Primary key will be generated. Always return the ADDED/UPDATED/CREATED item.

Hope this helps: http://www.codereye.com/2010/12/configure-tomcat-to-accept-http-put.html

I forgot, some sort of login API where I can send a username and password and you return the user or something on success would be great.

How to start API on Tomcat 9 Server

If you have tomcat 9 installed then go to:
C:\Program Files\Apache Software Foundation\Tomcat 9.0\webapps

and put the RestaurantAPI.war file I uploaded on github there.

Then go to:
C:\Program Files\Apache Software Foundation\Tomcat 9.0\bin

and double click the Tomcat9.exe

this will open a command line looking window that will run a bunch of start up and start the server.
Once you close that window it stops the server.

When ever you make an api call and the api picks it up i made it output what call is being run so we can double check.

Will only work if your database is under the 'postgres' database with schema named 'project'
and your pgadmin username should be 'postgres' and password is 'password'.

You can change it in pgadmin if you click File-> Change Password at the top