W4D2 Homework

Download the provided party-planner code. It has a form with fields for first name, last name, and favorite food and uses AJAX to send JSON data to the server.

Once you have extracted the zip file you can install all needed dependencies by going into the directory and executing:

```
$ npm install
```

Assignments:

1. Update the provided party planner to also have a field indicating the guest's arrival time. Be sure check for updates to the form, the client side JavaScript, the server side JavaScript, and the template that generates the list.

2. Write a Magic 8 ball application. The way it should work is: make a text field to type a question, and then a button with the text "Ask 8 Ball".

   When the button is pressed a GET request to /8ball should be sent to the server, which returns a random answer from the list shown below. When the answer is received the page should replace the question text with the answer provided by the 8 ball.

   ["It is Certain", "It is decidedly so", "Without a doubt", "Yes definitely", "You may rely on it", "As I see it, yes", "Most likely", "Outlook good", "Yes", "Signs point to yes", "Reply hazy, try again", "Ask again later", "Better not tell you now", "Cannot predict now", "Concentrate and ask again", "Don't count on it", "My reply is no", "My sources say no", "Outlook not so good", "Very doubtful"]

   Optional Extras:

   a) Make it so that the user only has to press enter in the textbox in order to send the request. Depending on how you structured the HTML you may already have this functionality.

   b) Have the answer selected after it is put in the textbox. That way if the user starts typing the 8ball answer is immediately replaced with what the user types. An easy way to do this is inside a focus handler execute this.select(). Then simply trigger the focus handler.

3. Update question q3_4 from W4D1 to use AJAX on the product template to add the product to the shopping cart (and then show a success message). Make your page have a link to view the cart along with a number indicating how many items are in the cart. This number should be updated when your product is added

   In essence you should POST the data to /addToCart, which should then return a number (the amount of items in the cart), which you then use to update the view-cart link.