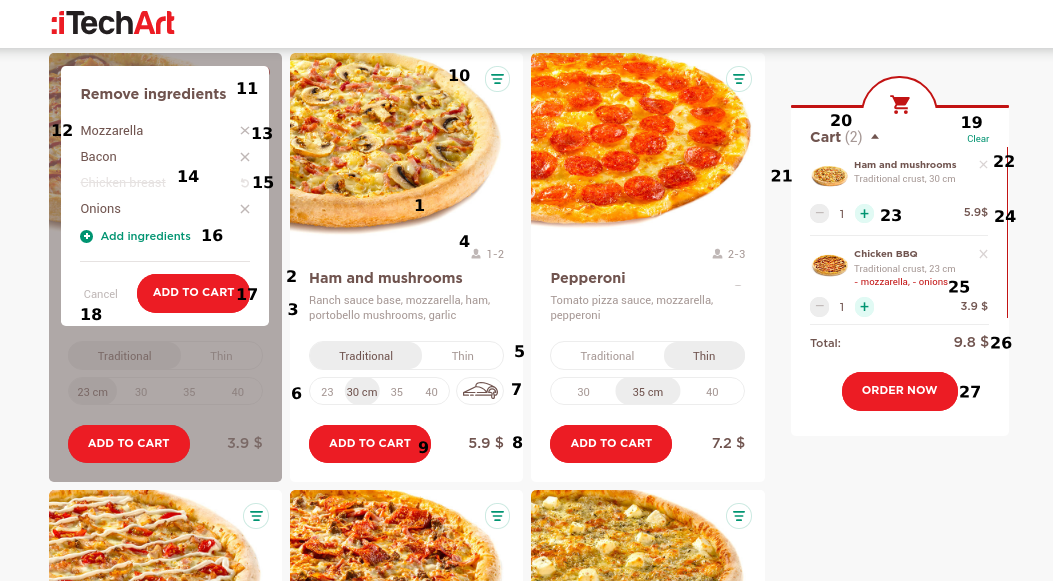


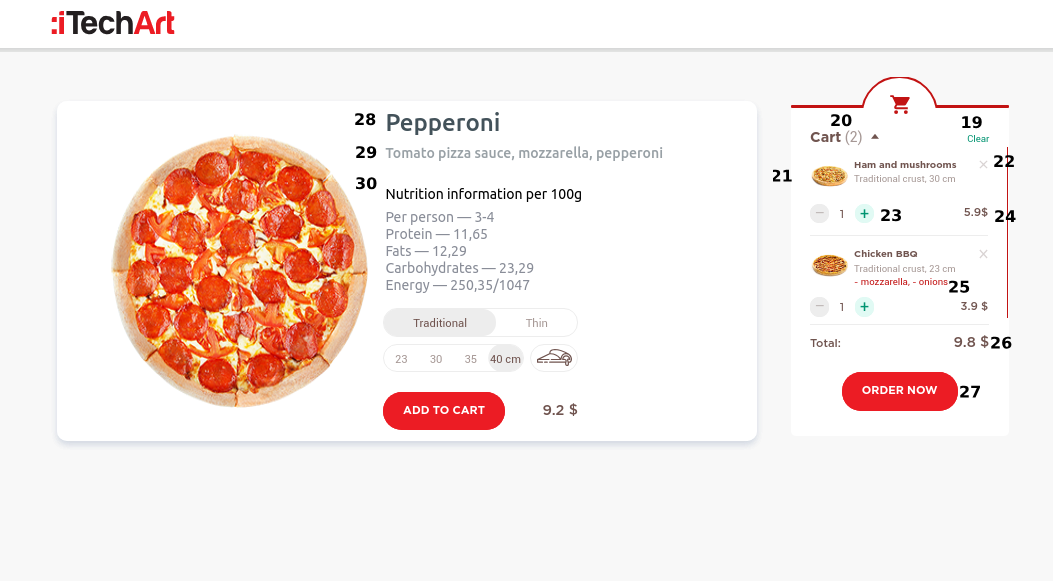
**Pizzeria Single Page Application**

**Design:**

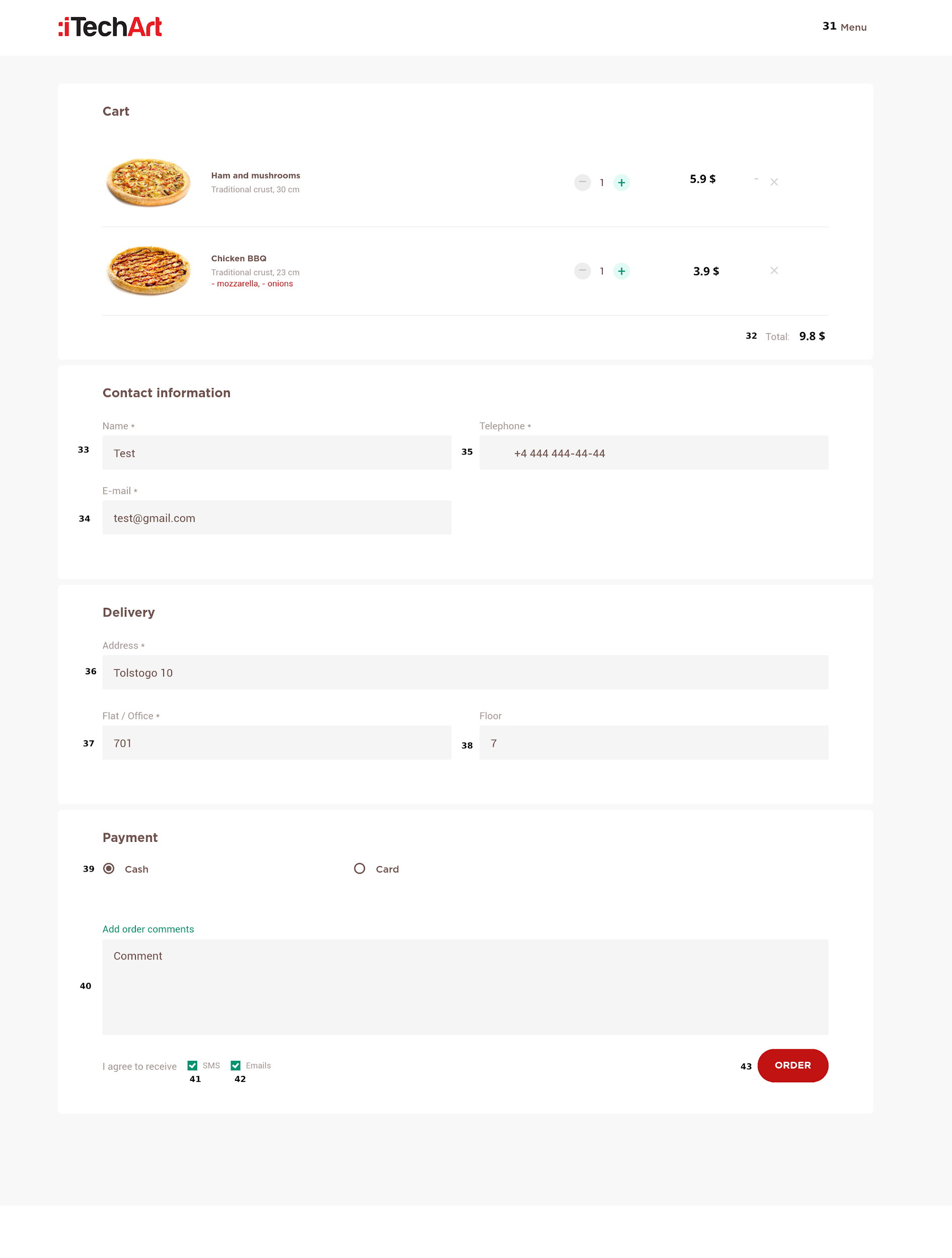
**[main view]**

****

**[product view]**

****

**[checkout view]**

****

**Specification:**

| **#** | **Item name** | **Description** | **Action** |
| --- | --- | --- | --- |
| **1** | Pizza image | Static image | **onClick:** go to product page |
| **2** | Pizza title | Static text |  |
| **3** | Pizza ingredients description | Static text |  |
| **4** | Number of persons | Dynamic text, depends on pizza size |  |
| **5** | Dough thickness | Radio group | **onChange:** recalculate pizza price |
| **6** | Pizza size | Radio group | **onChange:** recalculate pizza price |
| **7** | Cheese board | Checkbox, visible from size 30cm | **onChange:** increase pizza price by 15% |
| **8** | Pizza price | Dynamic price |  |
| **9** | Add to cart | Static button | **onClick:** add current pizza to cart |
| **10** | Change ingredients | Static button | **onClick:** show ingredients management modal |
| **11** | Modal title | Static text |  |
| **12** | Ingredient name | Static text |  |
| **13** | Exclude ingredient | Static button | **onClick:** exclude ingredient from pizza |
| **14** | Excluded ingredient | Static text |  |
| **15** | Rollback change | Static button | **onClick:** bring back the ingredient |
| **16** | Add ingredients | Input (Select) with autocomplete | **onChange:** recalculate pizza withingredient price |
| **17** | Add to cart | Static button | **onClick:** add current pizza to cart |
| **18** | Cancel | Static button | **onClick:** close ingredients modal |
| **19** | Cart clear | Dynamic button, should appear with cart items | **onClick:** clean out cart |
| **20** | Cart pizza quantity | Dynamic text |  |
| **21** | Cart item | Static text and image |  |
| **22** | Delete cart item | Static button | **onClick:** delete pizza from the cart |
| **23** | Cart item quantity | Dynamic counter, that affects the total pizza cost | **onClick:** decrease/increase count(min value = 1) |
| **24** | Total pizza cost | Dynamic text |  |
| **25** | Excluded ingredients | Dynamic text |  |
| **26** | Cart total cost | Dynamic text |  |
| **27** | Order | Static button | **onClick:** go to checkout page |
| **28** | Pizza name | Static text |  |
| **29** | Pizza ingredients | Static text |  |
| **30** | Nutrition information | Static text |  |
| **31** | Menu link | Static link | **onClick:** go to main page |
| **32** | Cart total cost | Dynamic text |  |
| **33** | Checkout name input | Input for checkout name.  Validators: minLength, required | **onChange:** should check field validation |
| **34** | Checkout email input | Input for checkout name.  Validators: email, required | **onChange:** should check field validation |
| **35** | Checkout phone input | Input for checkout name.  Validators: pattern (phone), required | **onChange:** should check field validation |
| **36** | Checkout address input | Input for checkout name.  Validators: minLength required | **onChange:** should check field validation |
| **37** | Checkout flat input | Input for checkout name.  Validators: pattern (number), required | **onChange:** should check field validation |
| **38** | Checkout floor input | Input for checkout name.  Validators: pattern (number), required | **onChange:** should check field validation |
| **39** | Checkout payment method | Radio group |  |
| **40** | Checkout comment textarea | Textarea for comment |  |
| **41** | Checkout SMS notifications | Checkbox |  |
| **42** | Checkout Emails notifications | Checkbox |  |
| **43** | Order button | Static button, should be disabled if any field is invalid | **onClick**:  clean up cart,  console.log checkout and cart form value, go to main page |

Single page pizza application should be powered by Angular or React. You can use mock API of your choice( http://www.mockapi.io, https://www.mockable.io/ ) or even JSON mocks .

User can select pizza: size(#6), dough thickness(#5) and add cheese board(#7), also you can include/exclude ingredients(#10). You can change pizza count (#23) in your cart or remove some (#22).

User can open pizza page when click on pizza image (#1).

Total price(#24) is a dynamic text value, displaying current sum of all pizzas total (count \*

pizza price).

Afterwards you can go to checkout page and fill all checkout fields with valid information.

After checkout form submitting (#43), you have to output a message to the web console with cart and checkout forms value, clean up cart and change route to main page.

**Choose your side: React or Angular :)** (better ask your manager)

**Angular**

**Useful links:**

* <https://www.typescriptlang.org/docs/home.html>
* <https://www.learnrxjs.io/>
* <https://angular.io/docs>
* <https://egghead.io/browse/frameworks/angular>
* <https://learn.javascript.ru/screencast/angular>
* <https://scrimba.com/g/gyourfirstangularapp>

**Use** ChangeDetectionStrategy.OnPush **and** async pipe **in all components in which it’s possible.**

**Various options for storing data:**

1. Services (<https://angular.io/guide/architecture#services-and-dependency-injection>)
2. Store (<https://ngrx.io/> or <https://github.com/ngxs/store>)

**React**

For quick start you can use create-react-app ( <https://github.com/facebookincubator/create-react-app> ) or any other react/redux boilerplate.

**Useful links:**

* <https://facebook.github.io/react/>
* <https://github.com/ReactTraining/react-router>
* <http://redux.js.org/>
* <https://github.com/reduxjs/react-redux>
* <https://egghead.io/courses/getting-started-with-redux>
* <https://egghead.io/courses/building-react-applications-with-idiomatic-redux>

**Workflow steps:**

1. **React + React-Router:** You can use mocked data for pizza list. Make in **master** branch.
2. **Adding Redux:** Add Redux to the application. Put pizza list, cart and checkout in the Redux store. Make in **feature/redux** branch**.**

**Common information**

**Notes:**

Focus on application functionality, not on visual styling, you can use any CSS-

framework (Material, Bootstrap, Foundation).

**Browser compatibility:**

Latest versions of Chrome, Safari, FF, IE.

**How To Do:**

1. Create private repo

2. Make project in development branch

3. Send access repo to manager.

Any questions? Mail me at [Dzmitry.Sabaleuski@itechart-group.com](mailto:Dzmitry.Sabaleuski@itechart-group.com)

|  |
| --- |