

**Plan for implementation Firebase into Restaurant project:**

Download zip file via Code button: [https://github.com/DonskiLive/71-10\\_11-Home-Task-71](https://github.com/DonskiLive/71-10_11-Home-Task-71)

After unzipping, inside application folder run command "npm install" - to get node\_modules

**In original project used json-server and axios**

For testing original project (make sure that <http://localhost:3000> is free  
and json-server already was installed with command "npm install -g json-server")  
go to src folder and run command "json-server --watch db.json".

After that, add new Git Bush Terminal and inside main folder of project run command "npm start"

<b>1 Firebase / Create new project ( <a href="https://console.firebaseio.google.com/u/0/">https://console.firebaseio.google.com/u/0/</a> )</b>	2
<b>2 Firebase and Visual Studio / Adding Firebase to your App</b>	3
<b>3 Firebase / Create Data-Base</b>	5
<b>3.1 Firebase / Create collection in Cloud Firestore</b>	5
<b>3.2 Firebase / Check (and if needed modify) Rules for your Cloud Firestore</b>	7
<b>4 Visual Studio / create Functions for communication with Firebase</b>	8
<b>5 Visual Studio / modify actions and reducer for communication with Firebase</b>	8
<b>6 Visual Studio / modify your application for new actions</b>	9
<b>7 Visual Studio and Firebase / test your modified application and Cloud Firestore</b>	10
<b>8 Firebase and Visual Studio / create Hosting for your application</b>	11
<b>9 Firebase / test your Hosting</b>	13

## 1 Firebase / Create new project (<https://console.firebaseio.google.com/u/0/>)

The screenshot shows the Firebase console homepage. At the top, there's a navigation bar with icons for 'Go to docs', a bell, and a user profile. Below the header, there's a section titled 'Your Firebase projects' featuring two projects: 'booksdb' (with ID 'booksdb-a718a') and 'PersonalTodo' (with ID 'personaltodo-b0b8f'). To the left of these projects is an illustration of a person working at a desk, and to the right is another illustration of a person working at a desk. Below the projects, there are two cards: one for 'Explore a demo project' (iOS) and one explaining that 'Firebase projects are containers for your apps'. This second card includes icons for mobile, iOS, web, and real-time database.

The screenshot shows the first step of creating a new project. It has a title 'Create a project (Step 1 of 3)' and a sub-section 'Let's start with a name for your project<sup>®</sup>'. A text input field is pre-filled with 'MenuOrder'. Below the input field is a small placeholder 'menuorder-bda1a'. A blue 'Continue' button is at the bottom. To the right of the form is an illustration of a man in a suit holding a smartphone and a woman in a yellow shirt working on a computer.

The screenshot shows the second step of creating a new project. It has a title 'Create a project (Step 2 of 2)' and a sub-section 'Google Analytics for your Firebase project'. It explains that Google Analytics enables targeting, reporting, and more. A list of features includes A/B testing, crash-free users, event-based Cloud Functions triggers, free unlimited reporting, user segmentation & targeting across Firebase products, and predicting user behavior. A checkbox labeled 'Enable Google Analytics for this project' is checked and marked as 'Recommended'. A blue 'Create project' button is at the bottom. To the right of the form is an illustration of a man in a suit working on a computer.

The screenshot shows the final step of creating a new project. It displays a circular logo for 'MenuOrder'. Below the logo, a green checkmark icon indicates 'Your new project is ready'. A blue 'Continue' button is at the bottom. To the right of the form is an illustration of a person's head in profile.

## 2 Firebase and Visual Studio / Adding Firebase to your App

Firebase Project Overview for 'MenuOrder'.

**Project Overview**

**Build**

- Authentication
- Firestore Database
- Realtime Database
- Storage
- Hosting
- Functions
- Machine Learning

**Release & Monitor**

**Analytics**

Get started by adding Firebase to your app

iOS | Android | </> | Add an app to get started

Add Firebase to your web app

1 Register app

App nickname: menu\_project\_firebase

Also set up Firebase Hosting for this app. [Learn more](#)

Register app

2 Add Firebase SDK

```
$ npm install firebase
```

Add Firebase to your web app

1 Register app

2 Add Firebase SDK

Use npm  Use a <script>-tag

If you're already using npm and a module bundler such as webpack or Rollup, you can run the following command to install the latest SDK:

```
$ npm install firebase
```

Then, initialize Firebase and begin using the SDKs for the products you'd like to use.

```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup/available-libraries

// Your web app's Firebase configuration
const firebaseConfig = {
  apiKey: "AIzaSyCwX...",
  authDomain: "menuorder-bd1a.firebaseio.com",
  projectId: "menuorder-bd1a",
  storageBucket: "menuorder-bd1a.appspot.com",
  messagingSenderId: "394694972173",
  appId: "1:394694972173:web:...",
};

// Initialize Firebase
const app = initializeApp(firebaseConfig);
```

Note: This option uses the modular JavaScript SDK, which provides reduced SDK size.

Learn more about Firebase for web: [Get Started](#) [Web SDK API Reference](#) [Samples](#)

Continue to console

ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ ТЕРМИНАЛ КОНСОЛЬ ОТЛАДКИ

```
,"arch":"x64"})  
added 1947 packages from 749 contributors and audited 1953 packages in 84.014s  
136 packages are looking for funding  
  run "npm fund" for details  
Found 93 vulnerabilities (66 moderate, 26 high, 1 critical)  
  run "npm audit fix" to fix them, or "npm audit" for details  
vitch@APTOP-RVQED1D: MINGW64 /c/hostinger/DONSKI_live/reactProjects/MenuOrder/menu_project_firebase (main)  
$ npm install firebase
```

**Add Firebase to your web app**

Register app

Add Firebase SDK

Use npm  Use a <script> tag

If you're already using [npm](#) and a module bundler such as [webpack](#) or [Rollup](#), you can run the following command to install the latest SDK:

```
$ npm install firebase
```

Then, initialize Firebase and begin using the SDKs for the products you'd like to use.

```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries

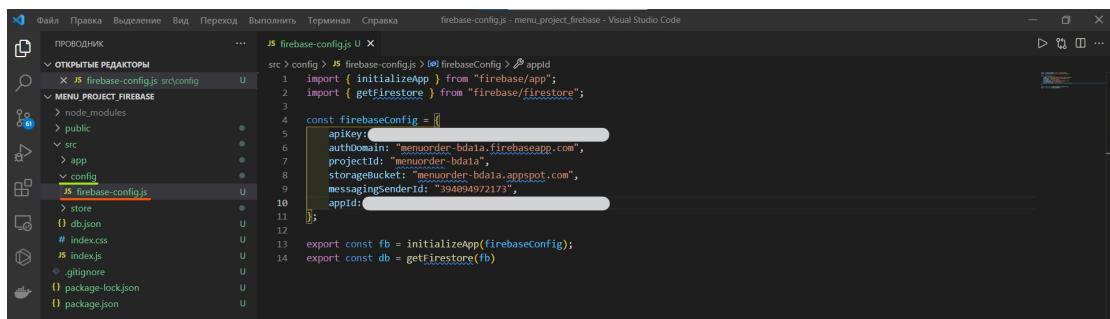
// Your web app's Firebase configuration
const firebaseConfig = {
  apiKey: "AIzaSyCm...",
  authDomain: "menurorder-bd1a.firebaseio.com",
  projectId: "menurorder-bd1a",
  storageBucket: "menurorder-bd1a.appspot.com",
  messagingSenderId: "394094972173",
  appId: "1:394094972173:web:...",
};

// Initialize Firebase
const app = initializeApp(firebaseConfig);
```

Note: This option uses the [modular JavaScript SDK](#), which provides reduced SDK size.

Learn more about Firebase for web: [Get Started](#) | [Web SDK API Reference](#) | [Samples](#)

[Continue to console](#)



**Add Firebase to your web app**

Register app

Add Firebase SDK

Use npm  Use a <script> tag

If you're already using [npm](#) and a module bundler such as [webpack](#) or [Rollup](#), you can run the following command to install the latest SDK:

```
$ npm install firebase
```

Then, initialize Firebase and begin using the SDKs for the products you'd like to use.

```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration
const firebaseConfig = {
  apiKey: "AIzaSyCm...",
  authDomain: "menurorder-bd1a.firebaseio.com",
  projectId: "menurorder-bd1a",
  storageBucket: "menurorder-bd1a.appspot.com",
  messagingSenderId: "394094972173",
  appId: "1:394094972173:web:...",
};

// Initialize Firebase
const app = initializeApp(firebaseConfig);
```

Note: This option uses the [modular JavaScript SDK](#), which provides reduced SDK size.

Learn more about Firebase for web: [Get Started](#) | [Web SDK API Reference](#) | [Samples](#)

[Continue to console](#)

### 3 Firebase / Create Data-Base

**Project Overview**

**Build**

- Authentication
- Firebase Database
- Realtime Database
- Storage
- Hosting
- Functions
- Machine Learning

**Release & Monitor**

**Analytics**

**Extensions**

**Spark** Free \$0/month      **Upgrade**

**MenuOrder** Spark plan

Choose a product to add to your app

Store and sync app data in milliseconds

**Authentication** Authenticate and manage users

**Cloud Firestore** Realtime updates, powerful queries, and automatic scaling

**Cloud Firestore**

Realtime updates, powerful queries, and automatic scaling

Create database

Is Cloud Firestore right for you? Compare Databases

**Create database**

Secure rules for Cloud Firestore

Set Cloud Firestore location

After you define your data structure, you will need to write rules to secure your data.

Learn more

Start in production mode

Your data is private by default. Client read/write access will only be granted as specified by your security rules.

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /{document} {
      allow read, write: if false;
    }
  }
}
```

Start in test mode

Your data is open by default to enable quick setup. However, you must update your security rules within 30 days to enable long-term client read/write access.

All third party reads and writes will be denied

Enabling Cloud Firestore will prevent you from using Cloud Datastore with this project, notably from the associated App Engine app

**Create database**

Secure rules for Cloud Firestore

Set Cloud Firestore location

Your location setting is where your Cloud Firestore data will be stored.

⚠ After you set this location, you cannot change it later. Also, this location setting will be the location for your default Cloud Storage bucket.

Cloud Firestore location

eur3 (europe-west)

Enabling Cloud Firestore will prevent you from using Cloud Datastore with this project, notably from the associated App Engine app

Cancel      Next

Enable

#### 3.1 Firebase / Create collection in Cloud Firestore

**Project Overview**

**Build**

- Authentication
- Firebase Database
- Realtime Database
- Storage
- Hosting
- Functions
- Machine Learning

**Release & Monitor**

**Analytics**

**Extensions**

**Cloud Firestore**

Data    Rules    Indexes    Usage

Prototype and test end-to-end with the Local Emulator Suite, now with Firebase Authentication      Get started

menuorder-bda1a

+ Start collection

Your database is ready to go. Just add data.

**Start a collection**

1 Give the collection an ID 2 Add its first document

Parent path /

Collection ID menudatabase

Cancel Next

```
[{"menu": [{"id": 1, "title": "Cesar salad", "price": 12, "url": "https://static.1000.menu/img/content/21458/_salat-cezar-s-kr-salat-cezar-s-krevetkami-s-malonezom_1501173720_1_max.jpg", "category": "salads", "id": 1}, {"id": 2, "title": "Pizza Margherita", "price": 10, "url": "https://blog.kesari.in/wp-content/uploads/2017/11/PizzaMargherita-Kesari-Tours.jpg", "category": "pizza", "id": 2}, {"id": 3, "title": "Pizza Napoletana", "price": 13, "url": "https://www.pizzanapolestan.org/struttura/pagine_bicolore/mobile/decalogo_avpn.jpg", "category": "pizza", "id": 3}, {"id": 4, "title": "Greece salad", "price": 8, "url": "https://assets.epicurious.com/photos/576454fb42e4a5ed06d1df6b/master/pass/greek-salad.jpg", "category": "salads", "id": 4}, {"id": 5, "title": "Cowboy Steak", "price": 25, "url": "https://i.cbc.ca/1.4491288.15162082291/fileImage/httpImage/image.jpg_gen/derivatives/16x9_780/cowboysteak.jpg", "category": "meat", "id": 5}], "orders": []}]
```

Document parent path /menudatabase

Document ID menu

Field	Type
menu	array

Type map

Field	Type	Value
id	number	1
title	string	Cesar salad

Field	Type	Value
url	string	https://i.cbc.ca/1
category	string	meat

Cancel Save

**Firebase**

Project Overview

Build

- Authentication
- Firestore Database**
- Realtime Database
- Storage
- Hosting
- Functions
- Machine Learning

Release & Monitor

Analytics

Cloud Firestore

MenuOrder

Data Rules Indexes Usage

Prototype and test end-to-end with the Local Emulator Suite, now with Firebase Authentication Get started

menu

menuorder-bda1a menudatabase menu

+ Start collection + Add document + Start collection + Add field

menu

- 0 (category: "salads", id: 1...)
- 1 (category: "pizza", id: 2...)
- 2 (category: "pizza", id: 3...)
- 3 (category: "salads", id: 4...)
- 4 (category: "meat", id: 5...)

**Add a document**

Parent path /menudatabase

Document ID orders

Field Type

orders array

Cancel Save

```
[{"menu": [{"id": 1, "title": "Cesar salad", "price": 12, "url": "https://static.1000.menu/img/content/21458/_salat-cezar-s-kr-salat-cezar-s-krevetkami-s-malonezom_1501173720_1_max.jpg", "category": "salads", "id": 1}, {"id": 2, "title": "Cowboy Steak", "price": 25, "url": "https://i.cbc.ca/1.4491288.15162082291/fileImage/httpImage/image.jpg_gen/derivatives/16x9_780/cowboysteak.jpg", "category": "meat", "id": 5}], "orders": []}]
```

The screenshot shows the Firebase Cloud Firestore interface. On the left, there's a sidebar with project settings like Authentication, Firestore Database, and Storage. The main area is titled 'Cloud Firestore' and shows a hierarchical database structure. At the top level is 'menu', which has a single child 'orders'. There are also 'Start collection' and 'Add document' buttons. A message at the top of the interface reads: 'Prototype and test end-to-end with the Local Emulator Suite, now with Firebase Authentication'.

### 3.2 Firebase / Check (and if needed modify) Rules for your Cloud Firestore

This screenshot shows the 'Edit rules' tab in the Cloud Firestore interface. It features a sample security rule:

```

1 rules_version = '2';
2 service cloud.firestore {
3     match /databases/{database}/documents {
4         match /{document=**} {
5             allow read, write: if false;
6         }
7     }
8 }
```

A 'Develop & Test' button is located in the top right corner.

This screenshot shows the 'Edit rules' tab with unpublished changes. It displays a sample security rule:

```

1 rules_version = '2';
2 service cloud.firestore {
3     match /databases/{database}/documents {
4         match /{document=**} {
5             allow read, write;
6         }
7     }
8 }
```

A 'Publish' button is located in the top right corner.

This screenshot shows the 'Edit rules' tab with a warning message: 'Your security rules are defined as public, so anyone can steal, modify, or delete data in your database'. A 'Learn more' link is located next to the warning message. The sample security rule is identical to the previous ones:

```

1 rules_version = '2';
2 service cloud.firestore {
3     match /databases/{database}/documents {
4         match /{document=**} {
5             allow read, write;
6         }
7     }
8 }
```

4 Visual Studio / [create Functions for communication with Firebase](#)

```
ПРОВОДНИК ... JS firebase-config.js U JS menuService.js x ...
src > service > JS menuService.js > [e] addOrderItem
1 import { db } from '../config.firebaseio-config'
2
3 import { doc, getDoc, setDoc, updateDoc,
4 arrayUnion } from 'firebase/firestore'
5
6 export const getAllMenus = async () => {
7   try {
8     const docRef = doc(db, 'menudatabase',
9       'menu')
10    const docData = await getDoc(docRef)
11    if (docData.exists()) {
12      return docData.data()
13    }
14    return { menu: [] }
15  } catch (error) {
16    console.log(error.message)
17  }
18
19 export const addOrderItem = async ({id, order}) =>
20 {
21   try {
22     const docRef = doc(db, 'menudatabase',
23       'orders')
24     const docData = await getDoc(docRef)
25     if (docData.exists()) {
26       await updateDoc(docRef, {
27         orders: arrayUnion({
28           id, order
29         })
30       })
31     } else {
32       await setDoc(docRef, {
33         orders: {
34           id, order
35         }
36       })
37     }
38   } catch (error) {
39     console.log(error.message)
40   }
41
42 export const getAllOrders = async () => {
43   try {
44     const docRef = doc(db, 'menudatabase',
45       'orders')
46     const docData = await getDoc(docRef)
47     if (docData.exists()) {
48       return docData.data()
49     }
50   } catch (error) {
51     console.log(error.message)
52   }
53 }
```

5 Visual Studio / modify actions and reducer for communication with Firebase

```
src > store > JS ActionCreator.js < ...
1 import Types from './ActionType';
2 // import axios from 'axios'
3 import { addOrderItem, getAllMenus, getAllOrders } from '../service/menuService'
4
5 const client = axios.create({
6   baseURL: "http://localhost:3000"
7 }) */
8
9 /* export const getMenu = () => {
10   return (Dispatch) => {
11     Dispatch({ type: Types.changeLoader, payload: true })
12     setTimeout(() => {
13       client.get('/menu')
14         .then(response => {
15           Dispatch({ type: Types.getMenu, payload: response.data })
16         }).catch(error => {
17           Dispatch({ type: Types.error, payload: error.message })
18         }).finally(() => {
19           Dispatch({ type: Types.changeLoader, payload: false })
20         })
21     }, 1500)
22   }
23 } */
24
25 export const getMenu = () => {
26   return async dispatch => {
27     Dispatch({ type: Types.changeLoader, payload: true })
28     try {
29       const response = await getAllMenus()
30       console.log(response.menu)
31       Dispatch({ type: Types.getMenu, payload: response.menu })
32     } catch (error) {
33       Dispatch({ type: Types.error, payload: error.message })
34     }
35     Dispatch({ type: Types.changeLoader, payload: false })
36   }
37 }
...
src > store > JS ActionCreator.js < ...
38 */
39 export const setOrder = (order) => {
40   let id = 0
41   const curOrder = {
42     id: id++,
43     order
44   }
45   return (dispatch) => {
46     client.post('/orders', JSON.stringify(curOrder), { headers: { 'Content-Type': 'application/json' } })
47     .then((data) => {
48       dispatch({ type: Types.setOrder, payload: data.id })
49     }).catch(error => {
50       dispatch({ type: Types.error, payload: error.message })
51     })
52   }
53 }
54
55 export const setOrder = (order) => {
56   return async dispatch => {
57     Dispatch({ type: Types.changeLoader, payload: true })
58     try {
59       const responseGet = await getAllOrders()
60       const curOrder = {
61         id: responseGet.orders.length + 1,
62         order
63       }
64       const response = await addOrderItem(curOrder)
65       console.log(response)
66       Dispatch({ type: Types.setOrder, payload: curOrder.id })
67     } catch (error) {
68       Dispatch({ type: Types.error, payload: error.message })
69     }
70     Dispatch({ type: Types.changeLoader, payload: false })
71   }
72 }
73
74 // add new Actions - start ...

```

The screenshot shows the Visual Studio Code interface with two tabs open. The left tab is titled 'ActionCreator.js' and contains code for managing orders. The right tab is titled 'ActionType.js' and contains code for defining action types. Both files are part of a project structure under 'src/store'. The code includes imports from 'Types', exports for actions like 'getOrderById', 'resetUserOrder', and 'addCart', and logic for handling asynchronous operations using promises and dispatch.

```
JS ActionCreator.js U
1 // add new Actions - start //
2 export const getOrderById = (id) => {
3   return async dispatch => {
4     dispatch({ type: Types.changeloader, payload: true })
5     try {
6       const response = await getAllOrders()
7       const allOrders = response.orders
8       const userOrder = allOrders.find(order => order.id === id)
9       if (userOrder) {
10         dispatch({ type: Types.getorder, payload: userOrder })
11       } else {
12         throw new Error('Wrong order number')
13     }
14   } catch (error) {
15     dispatch({ type: Types.error, payload: error.message })
16   }
17   dispatch({ type: Types.changeLoader, payload: false })
18 }
19
20 export const resetUserOrder = () => {
21   return {
22     type: Types.resetorder
23   }
24 }
25
26 // add new Actions End - end //
27
28 export const addToCart = (id) => {
29   return {
30     type: Types.addItem,
31     payload: id
32   }
33 }
34
35 export const removeItemFromCart = (id) => {
36
37 }

JS ActionType.js U
1 const Types = {
2   changeloader: 'MenuList/ initial loading process',
3   getMenuId: 'MenuList/ menu items loaded',
4   error: 'MenuList/ request errors',
5   addItem: 'Cart/ add item to cart',
6   removeItem: 'Cart/ remove item from cart',
7   postOrder: 'Cart/ post order to db',
8   clearCart: 'Cart/ clear cart',
9   searchCategory: 'MenuList/ search Category by input value',
10  getOrders: 'Order/ orders items loaded', // add new Action
11  resetorder: 'Order/ reset current order' // add new Action
12 }
13
14 export default Types;
```

```
src > store > JS Reducerjs > ...
```

```
JS Reducerjs U x
```

```
1 import Types from './ActionType';
2
3 const init = {
4   isLoading: false,
5   menu: [],
6   error: null,
7   cart: [],
8   totalPrice: 0,
9   orderCount: 0,
10  searchCategory: '',
11  userOrder: null // add order array
12 }
13
14 export default function Reducer(state = init, { type, payload }) {
15   switch (type) {
16     case Types.getMenu: return _getMenu(state, payload)
17     case Types.changeLoader: return _changeLoader(state, payload)
18     case Types.error: return _error(state, payload)
19     case Types.addItem: return _addItem(state, payload)
20     case Types.removeItem: return _removeItem(state, payload)
21     case Types.setOrder: return _setOrder(state, payload)
22     case Types.clearCart: return _clearCart(state)
23     case Types.searchCategory: return {...state, searchCategory: payload}
24     case Types.getOrder: return _getOrder(state, payload) // add state for order array
25     case Types.resetOrder: return _resetOrder(state, payload) // add state for order array
26     default: return state
27   }
28 }
29
30 const _resetOrder = (state) => { // add state for order array
31   return {
32     ...state,
33     error: null,
34     userOrder: null
35   }
36 }
37
38 const _getOrder = (state, payload) => { // add state for order array
39   return {
40     ...state,
41     userOrder: payload
42   }
43 }
44
45 const _getMenu = (state, payload) => {
46   return {
47     ...state,
48     error: null,
49     menu: [...payload]
50   }
51 }
52
53 const _changeLoader = (state, payload) => {
54   return {
55     ...state,
56     isLoading: payload
57   }
58 }
59
60 const _error = (state, payload) => {
61   return {
62     ...state,
63     error: payload
64   }
65 }
```

The screenshot shows a code editor with a file named `Cart.js` open. The code defines a modal component for a shopping cart. It includes a close button with an 'onClick' event handler, a modal body with a heading and a spinner component, and a footer with a clear button. The code uses functional components and hooks like `useState` and `useEffect`.

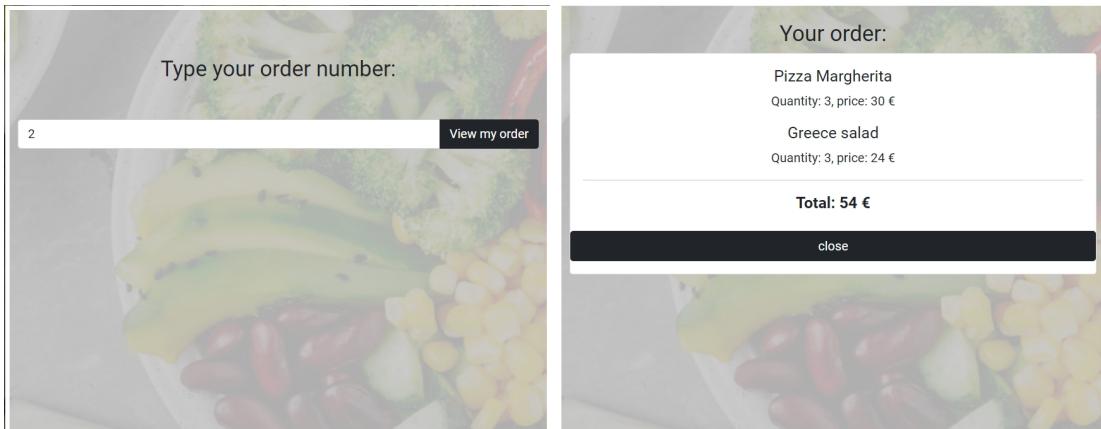
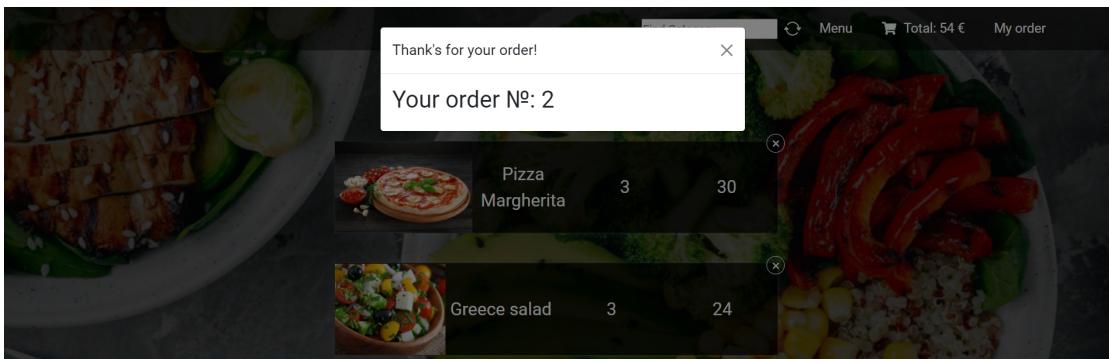
```
    aria-label="Close"
    onClick={clear}
  ></button>
</div>
<div className="modal-body">
  <h2>Your order:</h2>
  {isLoading ? <div className="spinner-border" role="status" /> : orderCount}
  <small>[" Add spinner"]</small>
</div>
</div>
</div>
</div>
</div>
);
}
);
};

const mapStateToProps = ({ cart, totalPrice, orderCount, isLoading }) => {
  return {
    cart,
    totalPrice,
    orderCount,
    isLoading, // for adding spinner
  }
}

export default Cart;
```

6 Visual Studio / modify your application for new actions

## 7 Visual Studio and Firebase / test your modified application and Cloud Firestore



A screenshot of the Firebase Cloud Firestore console. The left sidebar shows project settings and various services like Authentication, Firestore Database, Realtime Database, Storage, Hosting, Functions, Machine Learning, Analytics, Release & Monitor, and Engage. The main area shows the "orders" collection under the "menu" document in the "menudatabase" database. The data is displayed in a hierarchical tree view:

- orders
  - id: 2
    - order
      - 0
        - count: 3
        - price: 10
        - title: "Pizza Margherita"
      - 1
        - count: 3
        - price: 8
        - title: "Greece salad"

## 8 Firebase and Visual Studio / create Hosting for your application

The screenshot shows the Firebase Project Overview interface. On the left, a sidebar lists 'Build' categories: Authentication, Firestore Database, Realtime Database, Storage, Hosting (which is underlined in orange), Functions, and Machine Learning. The main content area is titled 'Hosting' with the sub-instruction: 'Deploy web and mobile web apps in seconds using a secure global content delivery network'. A large green Earth icon is visible on the right.

The screenshot shows the 'Set up Firebase Hosting' wizard. Step 1: 'Install Firebase CLI'. It includes instructions to run the command \$ npm install -g firebase-tools, a screenshot of the terminal, and a note about permissions. A 'Next' button is at the bottom. To the right is a blue smartphone icon connected to a laptop by a cable.

```
ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ ТЕРМИНАЛ КОНСОЛЬ ОТЛАДКИ
You can now view pizza-project in the browser.
Local: http://localhost:3002
On Your Network: http://192.168.1.21:3002
Note that the development build is not optimized.
To create a production build, use npm run build.
^Batchvergang abbrechen (J/N)?
vitch@LAPTOP-RVQOEHD6 MINGW64 /c/Hostingter/DONSK1_live/reactProjects/MenuOrder/menu_project_firebase (main)
$ npm install -g firebase-tools
```

```
ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ ТЕРМИНАЛ КОНСОЛЬ ОТЛАДКИ
+ firebase-tools@9.20.0
removed 4 packages and updated 2 packages in 64.20s
vitch@LAPTOP-RVQOEHD6 MINGW64 /c/Hostingter/DONSK1_live/reactProjects/MenuOrder/menu_project_firebase (main)
$ firebase projects:list
✓ Preparing the list of your Firebase projects
Project Display Name Project ID Project Number Resource Location ID
bookedb bookedb-a718a 80381107731 europe-west
MenuOrder menuorder-bdata 394094972173 europe-west
PersonalTodo personaltodo-bb6ef 562884729023 europe-west
3 project(s) total.
vitch@LAPTOP-RVQOEHD6 MINGW64 /c/Hostingter/DONSK1_live/reactProjects/MenuOrder/menu_project_firebase (main)
$ [ ]
```

The screenshot shows the 'Set up Firebase Hosting' wizard. Step 2: 'Initialize your project'. It includes instructions to open a terminal window and run \$ firebase login, followed by \$ firebase init. A 'Next' button is at the bottom. To the right is a blue smartphone icon connected to a laptop by a cable.

```
ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ ТЕРМИНАЛ КОНСОЛЬ ОТЛАДКИ
3 project(s) total.
vitch@LAPTOP-RVQOEHD6 MINGW64 /c/Hostingter/DONSK1_live/reactProjects/MenuOrder/menu_project_firebase (main)
$ firebase login
Already logged in as vitchynsky.y@gmail.com
vitch@LAPTOP-RVQOEHD6 MINGW64 /c/Hostingter/DONSK1_live/reactProjects/MenuOrder/menu_project_firebase (main)
$ [ ]
```

The screenshot shows a terminal window with the following content:

```
PROBLEMS   ВЫХОДНЫЕ ДАННЫЕ   ТЕРМИНАЛ   КОНСОЛЬ ОТЛАДКИ

Before we get started, keep in mind:
  * You are currently outside your home directory

? Are you ready to proceed? Yes
? Which Firebase features do you want to set up for this directory? Press Space to select features, then Enter to confirm your choices.
  ( ) Realtime Database: Configure a security rules file for Realtime Database and (optionally) provision default instance
  ( ) Firestore: Configure security rules and indexes files for Firestore
  ( ) Functions: Configure security rules and functions files for Cloud Functions
  (x) Hosting: Configure files for Firebase Hosting and (optionally) set up GitHub Action deploys
  ( ) Storage: Configure a security rules file for Cloud Storage
  ( ) Emulators: Set up local emulators for Firebase products
(Move up and down to reveal more choices)
```

At the top right of the terminal window, there are standard window control buttons (minimize, maximize, close) and a powershell icon.

```
ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ ТЕРМИНАЛ КОНСОЛЬ ОТЛАДКИ + \ ^ x
? Are you ready to proceed? Yes
? Which Firebase features do you want to set up for this directory? Press Space to select features, then Enter to confirm your choices. Hosting: Configure files for Firebase Hosting and (o
ptionally) set up GitHub Action deploys
== Project Setup

First, let's associate this project directory with a Firebase project.
You can create multiple project aliases by running firebase use --add,
but for now we'll just set up a default project.

? Please select an option: (Use arrow keys)
> use an existing project
Create a new project
Add Firebase to an existing Google Cloud Platform project
```

```
ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ ТЕРМИНАЛ КОНСОЛЬ ОТЛАДКИ + × ×
>Please select an option: Set an existing project
>Select a default Firebase project for this directory: menuorder-bdala (MenuOrder)
1 Using project menuorder-bdala (MenuOrder)

== Hosting Setup

Your public directory is the folder (relative to your project directory) that
will contain Hosting assets to be uploaded with firebase deploy. If you
have a build process for your assets, use your build's output directory.

? What do you want to use as your public directory? build
```

A screenshot of a terminal window titled 'Using project menuorder-bdata (menuOrder)'. The window shows the command 'cd menuorder' being run. Below it, the Firebase CLI is prompting for a public directory: 'What do you want to use as your public directory? build'. A question mark icon is next to the question. At the bottom, there's a configuration option: 'Configure as a single-page app (rewrite all urls to /index.html)? (y/N) y'. The terminal has a dark theme with light-colored text. There are some icons at the bottom left and a file browser sidebar on the right.

The screenshot shows the Firebase CLI interface with the following details:

- Project: menuunder-bdata (MenuUnder)
- Step: Hosting Setup
- Description: Your public directory is the folder (relative to your project directory) that will contain Hosting assets to be uploaded with `firebase deploy`. If you have a build process for your assets, use your build's output directory.
- Question: What do you want to use as your public directory? `build`
- Configuration: Configure as a single-page app (rewrite all urls to /index.html)? Yes
- Setting up automatic builds and deploys with GitHub? (y/n) `y`

```
ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ ТЕРМИНАЛ КОНСОЛЬ ОТЛАДКИ + v ^ x
? What do you want to use as your public directory? build
? Configure as a single-page app (rewrite all urls to /index.html)? Yes
? Set up automatic builds and deploys with GitHub? No
+ Wrote build/index.html

1 Writing configuration info to firebase.json...
1 Writing project information to .firebaserc...

+ Firebase initialization complete!

w1ch@APTOP-RVQ0EH06 MINGW64 /c/Hostinger/DONSKI_live/reactProjects/MenuOrder/menu_project_firebase (main)
$ npm run build
```

## × Set up Firebase Hosting

Install Firebase CLI

Initialize your project

Deploy to Firebase Hosting

When you're ready, deploy your web app  
Put your static files (e.g., HTML, CSS, JS) in your app's deploy directory (the default is "public").  
Then, run this command from your app's root directory:

```
$ firebase deploy
```

After deploying, view your app at [menuorder-bda1a.web.app](https://menuorder-bda1a.web.app).  
Need help? Check out the [Hosting docs](#).

[Continue to console](#)

```
vilchek@APTOR-RVNUEN16: MINGW64 /c/Hosting/00NSKI_1ive/reactProjects/MenuOrder/menu_project_firebase (main)
$ firebase deploy
== Deploying to "menuorder-bda1a"...
i: deploying hosting
i: hosting[menuorder-bda1a]: beginning deploy...
i: hosting[menuorder-bda1a]: found 17 files in build
i: hosting: adding files to version [0/17] (OK)
```

## 9 Firebase / test your Hosting

Firebase

Project Overview

Hosting

Dashboard Usage

menuorder-bda1a domains

Domain	Status
menuorder-bda1a.web.app	Default
menuorder-bda1a.firebaseioapp.com	Default

Add custom domain

