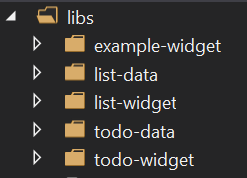
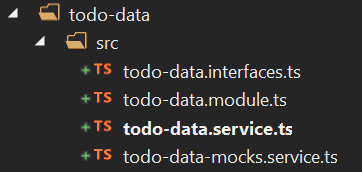
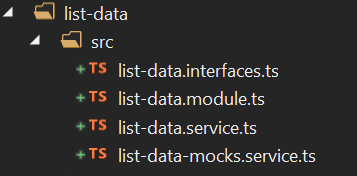
**Inter-widget communication with rest services**

1. For rest-service integration, create data-modules using RAML file.  
   The data module folder consist of Mock service file and Service file which contains the method (get/post/put).  
     
   Command to create data modules inside libs :  
   ng generate data-module --collection @bb-cli/schematics --name TodoData --ramlPath ./my-spec-folder/target/classes/api.raml  
     
   Generated Libs structure :  
     
     
   libs ->  
    ->list-data (data-module for list-widget)  
    ->list-widget  
    ->todo-data (data-module for todo-widget)  
    ->todo-widget  
     
   **Task to achieve** : (using appointment.raml to create data-module)  
   - list-widget will show lists of appointments (subjects), by clicking on individual appointment we need to show details regarding the particular appointment in todo-widget .  
     
   **Process:**   
   - To achieve given task we need to create two data-module from two raml files, list-data will create data-module for list-widget which consider only the list of appointments with date, name and id of appointments and todo-data will create data-module for todo-widget which have the details of the appointments.  
     
     
     
   list-data-mocks.service.ts / todo-data-mocks.service.ts: It has the data with url-pattern , which is generated using the raml and json file .  
   list.data.service.ts / todo.data.service.ts: It has the method for Http request and response .  
   list.data.module.ts / todo.data.module.ts: It has the dependency injections and declaration .  
   list-data.interface.ts / todo.data. interface.ts : It has the interface for response data that we need to pass to widget .
2. Integrate list-data to list-widget  
   libs/list-widget/src/list.service.ts  
     
   import { Injectable } from '@angular/core';

import { HttpResponse } from '@angular/common/http';

import { Observable, ReplaySubject } from 'rxjs';

import { map, switchMap } from 'rxjs/operators';

import {

ListDataService

} from '../../list-data/src/list-data.service';// import list-data service module into list.service

export interface ListItem {

id: string;

title: string;

dueDate: Date;

}

@Injectable()

export class ListService {

private readonly id = new ReplaySubject<string>();

constructor(private readonly data: ListDataService) {}  
 // Assign listdataservice to a private variable which will create a instance of listdataservice

readonly items: Observable<Array<any>> = this.data.getDataList()

.pipe(map((response: HttpResponse<any>): Array<any> => {

return response.body

? response.body

: [];

}));  
 // Calling listData method to get lists of appointments

getItem(id: string) {

this.id.next(id);

}

}

Method in list-data service :   
libs/list-data/src/list-data.service.ts (auto generated file)  
  
export class ListDataService {

constructor(private readonly http: HttpClient,

@Inject(LIST\_DATA\_CONFIG)

private readonly config: ServiceDataHttpConfig) { }

getDataList(params?: undefined, headers: HttpHeaders = new HttpHeaders({})): Observable<HttpResponse<any>> {

const uri = `${this.config.apiRoot}${this.config.servicePath}/${version}/dataList`;

return this.http.request<any>('get', uri, {

params: normalizeHttpParameters(params),

headers,

observe: 'response',

responseType: 'json',

withCredentials: false,

});

}

}

1. Integrate todo-data to todo-widget  
   libs/todo-widget/src/todo.service.ts  
     
   import { Injectable } from '@angular/core';

import { ActivatedRoute } from '@angular/router';

import { HttpResponse } from '@angular/common/http';

import { Observable, ReplaySubject } from 'rxjs';

import { map, switchMap, pluck, filter } from 'rxjs/operators';

import {

TodoDataService

} from '../../todo-data/src/todo-data.service'; // import todo-data service module into todo.service

export interface TodoItem {

id: string;

title: string;

dueDate: Date;

}

@Injectable()

export class TodoService {

private readonly id = new ReplaySubject<string>();

public items!: Observable<Array<any>>;

getItems(getid: string) {

return this.items = this.data.getAppointments(getid)

.pipe(map((response: HttpResponse<any>): Array<any> => {

return response.body

? response.body

: [];

}));

}  
 // Calling todoData method to get lists of appointments

getItem(id: string) {

this.id.next(id);

}

constructor(private readonly data: TodoDataService, private route: ActivatedRoute) { }  
 // Assign tododataservice to a private variable which will create a instance of tododataservice

}

Method in todo-data service :   
libs/todo-data/src/todo-data.service.ts (auto generated file)  
  
export class TodoDataService {

constructor(private readonly http: HttpClient,

@Inject(TODO\_DATA\_CONFIG)

private readonly config: ServiceDataHttpConfig) { }

getAppointments(todoId: string, params?: undefined, headers: HttpHeaders = new HttpHeaders({})): Observable<HttpResponse<any>> {

const uri = `${this.config.apiRoot}${this.config.servicePath}/${version}/appointments/${todoId}`;

return this.http.request<any>('get', uri, {

params: normalizeHttpParameters(params),

headers,

observe: 'response',

responseType: 'json',

withCredentials: false,

});

}

}

1. Showing list data in list-widget and emitting id to todo-widget .  
   libs/list-widget/src/list.widget.component.ts  
     
   import { Component, Input, Output, EventEmitter } from '@angular/core';

import { Router } from '@angular/router';

import { Observable } from 'rxjs';

import { ListItem, ListService } from './list.service';

import { RoutableContainer } from '@backbase/foundation-ang/core';

@RoutableContainer()

@Component({

selector: 'bb-todo-dataList',

template: `

<h1>List Of todos</h1>

<div \*ngFor="let todo of todos | async" [ngClass]="{'bg-warning':selectedId == todo.id}" class="row" style="padding:10px;">

<div class="col-md-6">

<p>{{todo.subject}}</p>

</div>

<div class="col-md-6">

<button type="button" class="btn btn-primary" (click)="selectAccount(todo.id)" role="button">View Appointment Details</button>

</div>

</div>

`,

providers: [ListService]

})

export class ListWidgetComponent {

todos: any;

selectedId: any;

@Output()

selectedAccount = new EventEmitter<string>();

selectAccount(id: string) {

this.selectedId = id;

this.selectedAccount.emit(id);//Emitting id

}

constructor(private router: Router, private todoService: ListService) {

this.todos = todoService.items;

}

}

1. Catching emitted id in todo-widget and showing appointment details accordingly  
   libs/todo-widget/src/todo.widget.component.ts  
     
   import { Component, Input } from '@angular/core';

import { ActivatedRoute } from '@angular/router';

import { TodoService, TodoItem } from './todo.service';

import { Observable, ReplaySubject } from 'rxjs';

import { map, switchMap, pluck, filter } from 'rxjs/operators';

import { RoutableContainer } from '@backbase/foundation-ang/core';

@RoutableContainer()

@Component({

selector: 'bb-todo-list',

template: `

<h1>List Of Appointments</h1>

<div \*ngIf="checkId == undefined">

<h2>No appointment selected</h2>

</div>

<div \*ngFor="let todo of todos | async">

<div>

<h2>

{{ todo.subject }}

</h2>

<table class="table">

<thead>

<tr>

<th scope="col">Full Name</th>

<th scope="col">Role</th>

</tr>

</thead>

<tbody>

<tr \*ngFor="let guest of todo.guests">

<td>{{guest.personalData.fullname}}</td>

<td>{{ guest.role }}</td>

</tr>

</tbody>

</table>

</div>

</div>

`,

providers: [TodoService]

})

export class TodoWidgetComponent {

todos: any;

checkId: any;

reciveData: Observable<{}> = this.route.params.pipe(

pluck('selectedAccount'),

map((selectedAccount) => {

return (selectedAccount)

})

) // Assigning selectedAccount (appointment id) to reciveData (generated variable)

constructor(private route: ActivatedRoute, private todoService: TodoService) {

this.reciveData.subscribe((data) => {

this.checkId = data;

if (this.checkId) {

this.todos = todoService.getItems(this.checkId); // Calling service function with appointment id , which will further call get method of data-module and get detail data for given appointment .

}

});

}

}

1. Dependency injection for list-widget   
   libs/list-widget/src/list-widget.module.ts  
     
   import { NgModule } from '@angular/core';

import { CommonModule } from '@angular/common';

import { RouterModule, Route } from '@angular/router';

import { BackbaseCoreModule } from '@backbase/foundation-ang/core';

import { TodoDataHttpModule } from '@backbase/reference-data-http-ang/todo';

import { ListDataModule } from '../../list-data/src/list-data.module';

import { ListWidgetComponent } from './List-widget.component';

import { ListService } from './list.service';

import { DataHttpModule } from '@backbase/foundation-ang/data-http';

export const todoWidgetRoutes: Route[] = [];

@NgModule({

imports: [

CommonModule,

RouterModule,

TodoDataHttpModule,

ListDataModule,

DataHttpModule.forRoot({

apiRoot: '/custom-api/',

}),

BackbaseCoreModule.withConfig({

classMap: { ListWidgetComponent }// mapping widget for routing interwidget communication

})

],

declarations: [ListWidgetComponent],

providers: [ListService],

entryComponents: [],

})

export class ListWidgetModule { }

1. Dependency injection for todo-widget  
   libs/todo-widget/src/todo-widget.module.ts  
     
   import { NgModule } from '@angular/core';

import { CommonModule } from '@angular/common';

import { RouterModule, Route } from '@angular/router';

import { BackbaseCoreModule } from '@backbase/foundation-ang/core';

import { TodoDataHttpModule } from '@backbase/reference-data-http-ang/todo';

import { TodoDataModule } from '../../todo-data/src/todo-data.module';

import { TodoWidgetComponent } from './todo-widget.component';

import { TodoService } from './todo.service';

import { DataHttpModule } from '@backbase/foundation-ang/data-http';

export const todoWidgetRoutes: Route[] = [];

@NgModule({

imports: [

CommonModule,

RouterModule,

TodoDataHttpModule,

TodoDataModule,

DataHttpModule.forRoot({

apiRoot: '/custom-api/',

}),

BackbaseCoreModule.withConfig({

classMap: { TodoWidgetComponent }// mapping widget for routing interwidget communication

})

],

declarations: [TodoWidgetComponent],

providers: [TodoService],

entryComponents: [],

})

export class TodoWidgetModule {}

1. Injecting both widget in app.module.ts   
   apps/my-example-app /src/app/app.module.ts  
     
   import { BrowserModule } from '@angular/platform-browser';

import { NgModule } from '@angular/core';

import { ContainersModule } from '@backbase/foundation-ang/containers';

import { AppComponent } from './app.component';

import { RouterModule } from '@angular/router';

import { BackbaseCoreModule } from '@backbase/foundation-ang/core';

import { environment } from '../environments/environment';

import { ExampleWidgetModule } from '@backbase/example-widget';

import { TodoWidgetModule } from '@backbase/todo-widget';

import { ListWidgetModule } from '@backbase/list-widget'; //Injecting widgets in app

import { ListDataModule } from '@backbase/list-data';

@NgModule({

declarations: [

AppComponent

],

imports: [

BrowserModule,

ContainersModule,

BackbaseCoreModule,

RouterModule.forRoot([], { initialNavigation: false, useHash: true }),

ExampleWidgetModule,

TodoWidgetModule,

ListWidgetModule, //Declaring dependency

ListDataModule,

BackbaseCoreModule.withConfig({

classMap: {}

}),

],

providers: [...environment.mockProviders],

bootstrap: [AppComponent]

})

export class AppModule { }

1. Declaring to access mock service in environment.ts  
   apps/my-example-app/src/environments/environment.ts  
     
   import { TodoDataMocksProvider } from '../../../../libs/todo-data/src/todo-data-mocks.service';

import { ListDataMocksProvider } from '../../../../libs/list-data/src/list-data-mocks.service';   
// Injecting mock service file for list-data and todo-data

import { Provider } from '@angular/core';

import { createMocksInterceptor } from '@backbase/data-http-ang';

export const environment = {

production: false,

mockProviders: [createMocksInterceptor(), TodoDataMocksProvider, ListDataMocksProvider],   
 // Declaring mock data services

};

1. Change dev-mock.js with following code :  
     
   window.document.addEventListener('DOMContentLoaded', () => {

const services = {};

const pageModel = {

name: 'bb-my-example-app-ang',

properties: {},

children: [{

name: 'deckContainer',

properties: {

classId: 'DeckContainerComponent'

},

children: [{

name: 'deck0',

properties: {

classId: 'panelContainerComponent',

},

children: [{

name: 'list-widget',

properties: {

classId: 'ListWidgetComponent', // mapping class declared in widget module

'output.selectedAccount': 'navigation:deck0' //emiting value trough navigation

}

},

{

name: 'todo-widget',

properties: {

classId: 'TodoWidgetComponent'

}

},

]

},

]

}]

};

window.BB.startSingleApp(services)

.then(app => app.bootstrap(pageModel));

});

1. Add preference in model.xml of list-widget ,   
   A preference is added to xml , as we can change the navigation container name once the widget is imported to cxp manager  
     
   <property name="output.selectedAccount">

<value>navigation:todo-widget</value>

</property>

1. Generating package  
   for creating package with mock services , we need to create one file in apps/my-example-app /src/environments/ named as environment.dev.ts.  
     
     
     
   - Copy the code of environment.ts to environment.dev.ts   
   - Change angular.json accordingly   
    "configurations": {

"production": {

"fileReplacements": [

{

"replace": "apps/my-example-app/src/environments/environment.ts",

"with": "apps/my-example-app/src/environments/environment.prod.ts"

}

],

"optimization": true,

"outputHashing": "all",

"sourceMap": false,

"extractCss": true,

"namedChunks": false,

"aot": true,

"extractLicenses": true,

"vendorChunk": false,

"buildOptimizer": true

},

"dev": {

"fileReplacements": [

{

"replace": "apps/my-example-app/src/environments/environment.ts",

"with": "apps/my-example-app/src/environments/environment.dev.ts"

}

],

"optimization": true,

"outputHashing": "all",

"sourceMap": false,

"extractCss": true,

"namedChunks": false,

"aot": true,

"extractLicenses": true,

"vendorChunk": false,

"buildOptimizer": true

}

}

},

"serve": {

"builder": "@angular-devkit/build-angular:dev-server",

"options": {

"browserTarget": "my-example-app:build"

},

"configurations": {

"production": {

"browserTarget": "my-example-app:build:production"

},

"dev": {

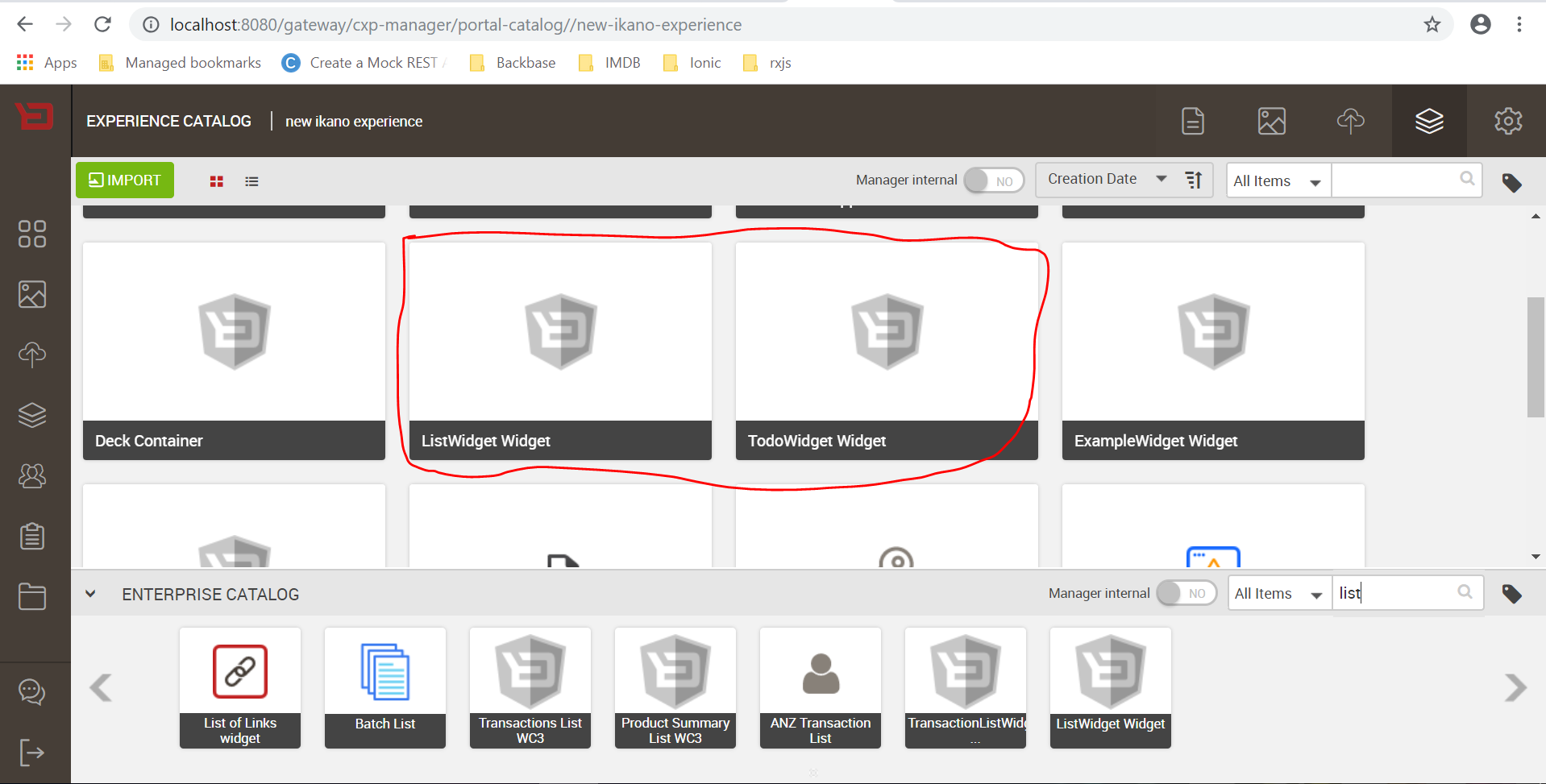
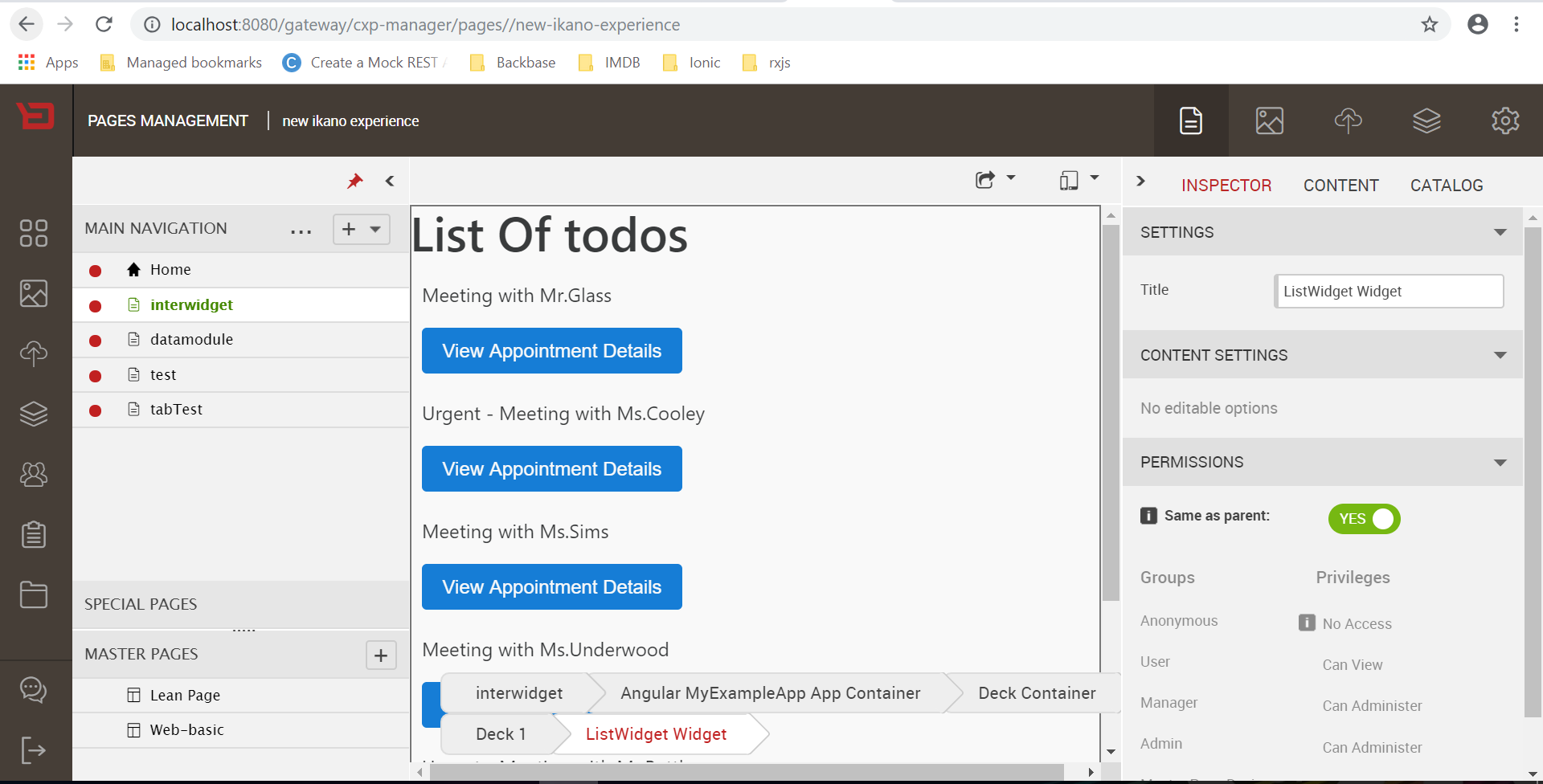
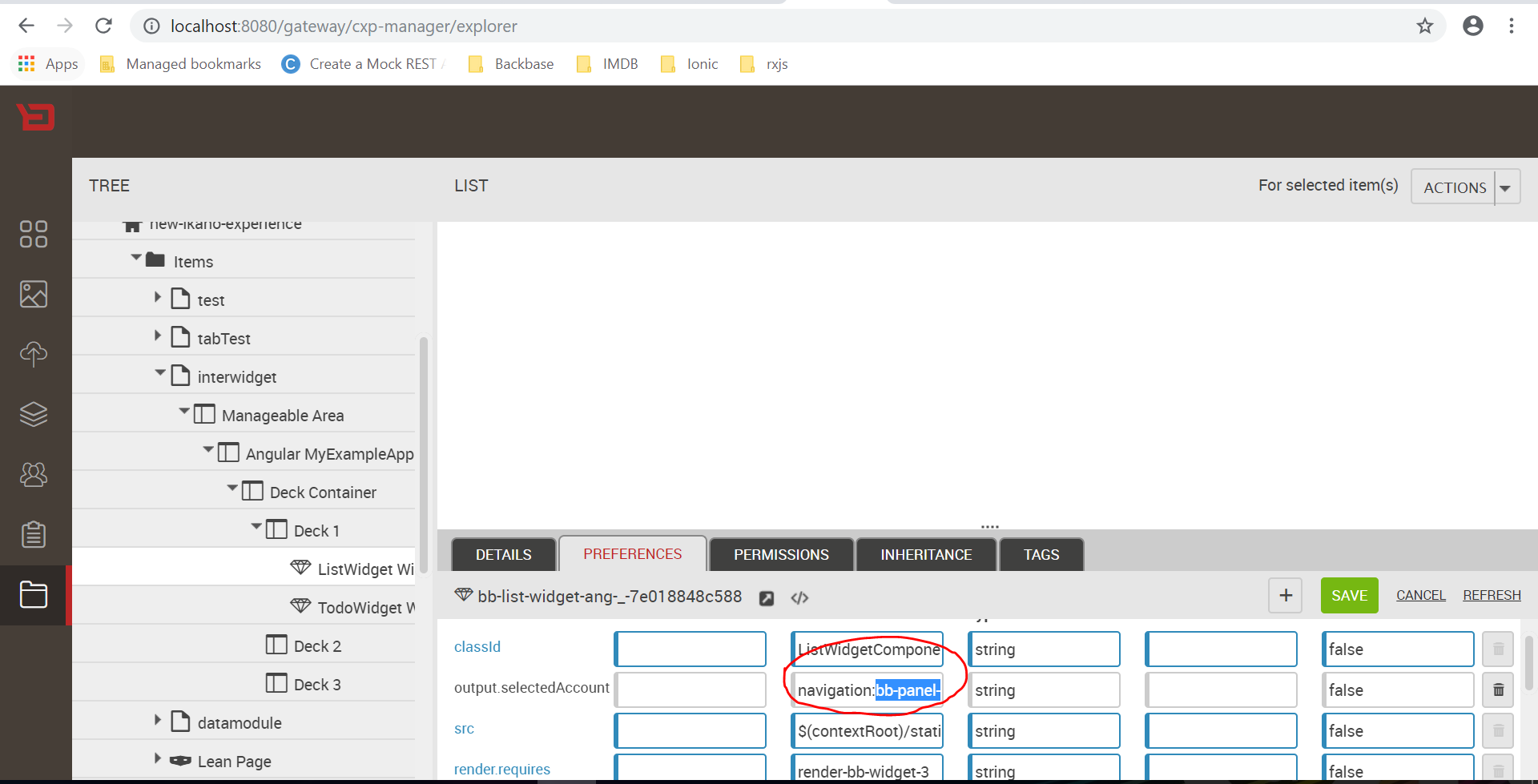
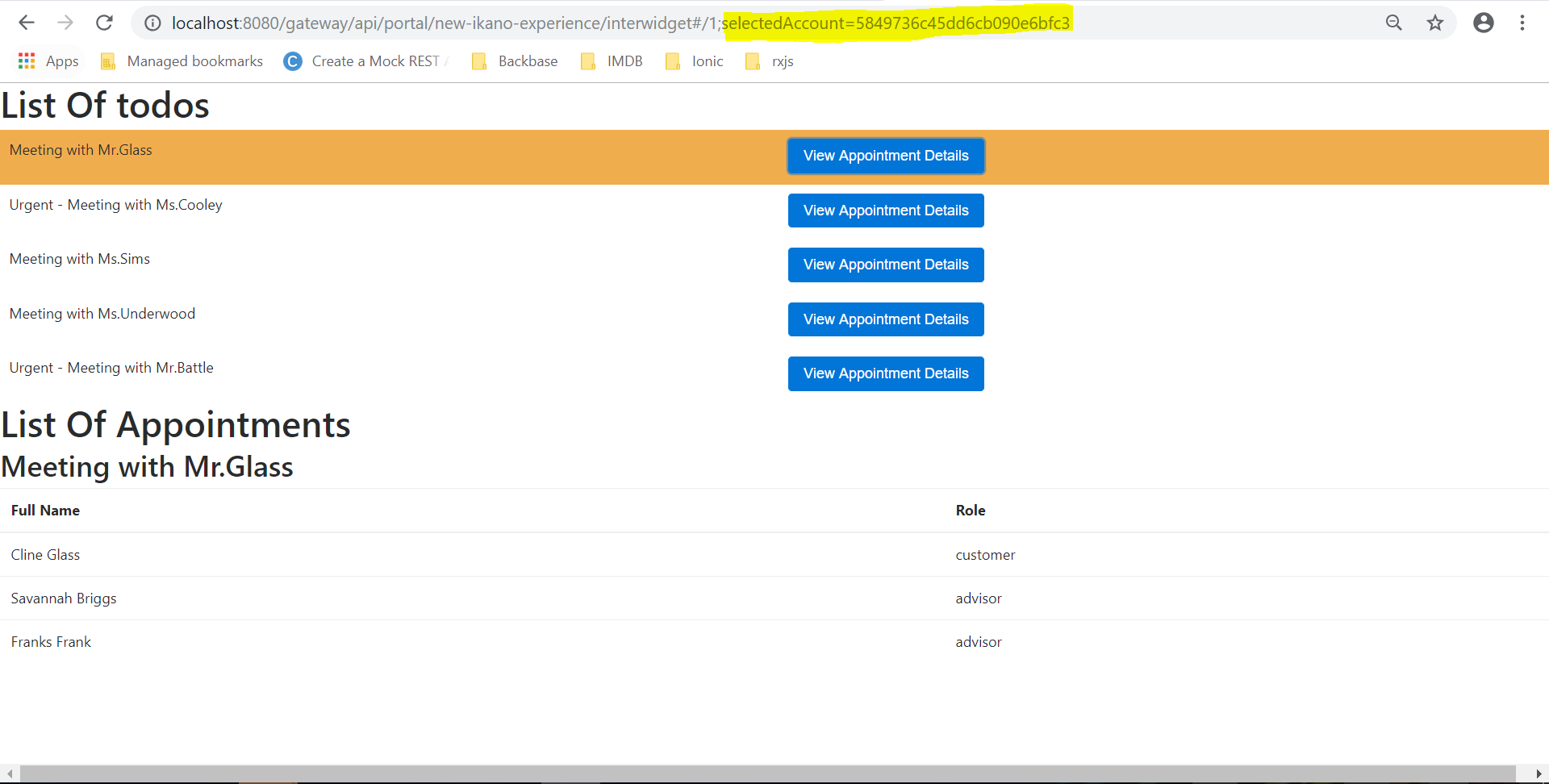
"browserTarget": "todo-app:build:dev"

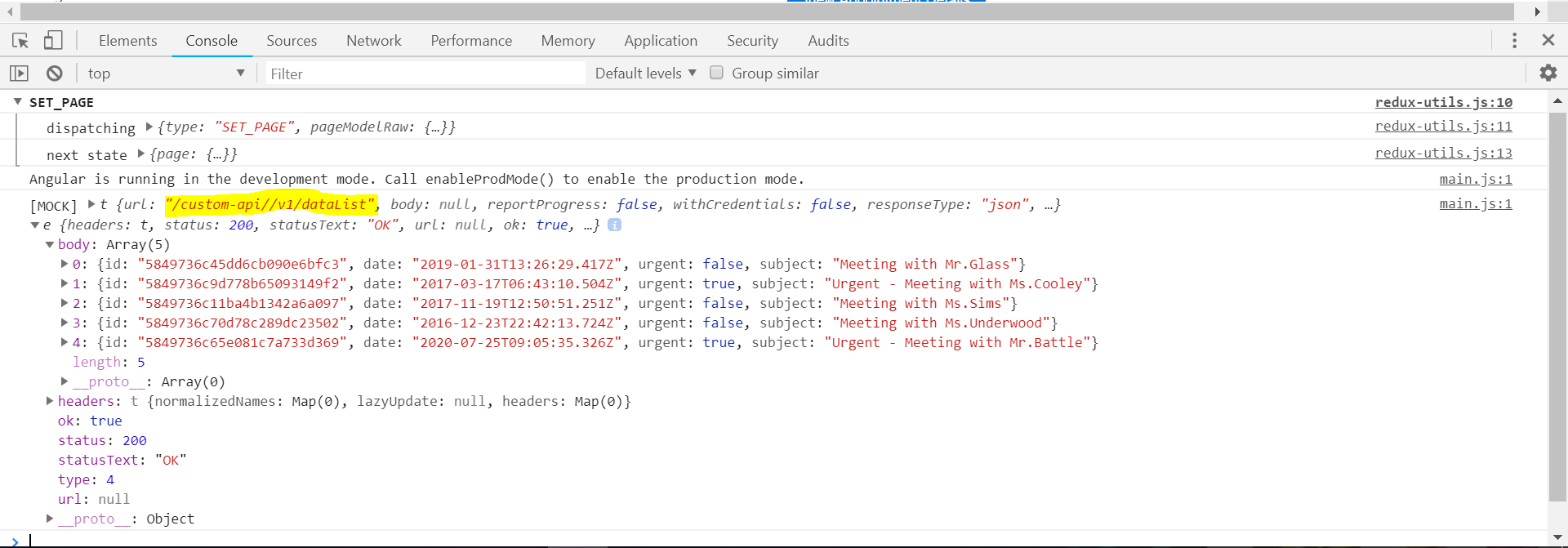
}

}

},  
generate package with following command :  
npm run package:apps --configuration=dev //--configuration=dev will create package using the “dev” configuration mentioned in angular.json

import package to cxp portal :  
bb-import package dist/provisioning-packages/my-example-app.zip --portal-port=8080 --portal-host=localhost --portal-username admin --portal-password admin --portal-context gateway/api --portal-auth-path gateway/api/auth/login --portal-version 6

1. Add todo-widget and list-widget from experience catalouge   
     
   
2. Drag and drop both widget to Deck1 in deck container   
     
   
3. Change preference name to Deckcontainer name from explorer :  
     
   
4. Result :  
     
   

you can see mock services url with response in console :  
  
list-widget service :  
  
  
  
todo-widget service once user click any appointment in list-widget :  
