**book-edit.component.html**

<div>

<h2>Book Edit</h2>

<div>

<label>Book Name:

<input [(ngModel)]="bookData.title" placeholder="Book Name"/>

</label><br>

<label>Book Price:

<input [(ngModel)]="bookData.price" placeholder="Book Price"/>

</label><br>

<label>Book Volume:

<input [(ngModel)]="bookData.volume" placeholder="Book Volume"/>

</label><br>

<label>Book Publish Date:

<input [(ngModel)]="bookData.publishDate" placeholder="Book Published Date"/>

</label><br>

</div>

<button (click)="updateBook()">Update</button>

</div>

**app.module.ts**

import { BrowserModule } from '@angular/platform-browser';

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

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

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

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

import { FormsModule } from '@angular/forms';

import { BookComponent } from './book/book.component';

import { BookAddComponent } from './book-add/book-add.component';

import { BookDetailComponent } from './book-detail/book-detail.component';

import { BookEditComponent } from './book-edit/book-edit.component';

const appRoutes: Routes = [

{

path: 'books',

component: BookComponent,

data: { title: 'Book List' }

},

{

path: 'books/:bookId',

component: BookDetailComponent,

data: { title: 'Book Details' }

},

{

path: 'book-add',

component: BookAddComponent,

data: { title: 'Book Add' }

},

{

path: 'book-edit/:id',

component: BookEditComponent,

data: { title: 'Book Edit' }

},

{ path: '',

redirectTo: '/books',

pathMatch: 'full'

}

];

@NgModule({

declarations: [

AppComponent,

BookComponent,

BookAddComponent,

BookDetailComponent,

BookEditComponent

],

imports: [

RouterModule.forRoot(appRoutes),

FormsModule,

BrowserModule,

HttpClientModule

],

providers: [],

bootstrap: [AppComponent]

})

export class AppModule { }

**rest.service.ts**

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

import { HttpClient, HttpHeaders, HttpErrorResponse } from '@angular/common/http';

import { Observable, of } from 'rxjs';

import { map, catchError, tap } from 'rxjs/operators';

const endpoint = 'http://localhost:8181/bookstore/';

const httpOptions = {

headers: new HttpHeaders({

'Content-Type': 'application/json'

})

};

@Injectable({

providedIn: 'root'

})

export class RestService {

constructor(private http: HttpClient) { }

private extractData(res: Response) {

let body = res;

return body || { };

}

getBooks(): Observable<any> {

return this.http.get(endpoint + 'books').pipe(

map(this.extractData));

}

getBook(id): Observable<any> {

return this.http.get(endpoint + 'books/' + id).pipe(

map(this.extractData));

}

addBook (book): Observable<any> {

console.log(book);

return this.http.post<any>(endpoint + 'books', JSON.stringify(book), httpOptions).pipe(

tap((book) => console.log(`added book w/ Name=${book.title}`)),

catchError(this.handleError<any>('addBook'))

);

}

updateBook (id, book): Observable<any> {

return this.http.put(endpoint + 'books/' + id, JSON.stringify(book), httpOptions).pipe(

tap(\_ => console.log(`updated book id=${id}`)),

catchError(this.handleError<any>('updateBook'))

);

}

deleteBook (id): Observable<any> {

return this.http.delete<any>(endpoint + 'books/' + id, httpOptions).pipe(

tap(\_ => console.log(`deleted book id=${id}`)),

catchError(this.handleError<any>('deleteBook'))

);

}

private handleError<T> (operation = 'operation', result?: T) {

return (error: any): Observable<T> => {

// TODO: send the error to remote logging infrastructure

console.error(error); // log to console instead

// TODO: better job of transforming error for user consumption

console.log(`${operation} failed: ${error.message}`);

// Let the app keep running by returning an empty result.

return of(result as T);

};

}

}

**app.component.html**

<router-outlet></router-outlet>

**book.component.ts**

import { Component, OnInit } from '@angular/core';

import { RestService } from '../rest.service';

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

@Component({

selector: 'app-book',

templateUrl: './book.component.html',

styleUrls: ['./book.component.css']

})

export class BookComponent implements OnInit {

books:any = [];

constructor(public rest:RestService, private route: ActivatedRoute, private router: Router) { }

ngOnInit() {

this.getBooks();

}

getBooks() {

this.books = [];

this.rest.getBooks().subscribe((data: {}) => {

console.log(data);

this.books = data;

});

}

add() {

this.router.navigate(['/book-add']);

}

delete(id) {

this.rest.deleteBook(id)

.subscribe(res => {

this.getBooks();

}, (err) => {

console.log(err);

}

);

}

}

**book.component.html**

<h2>Book List</h2>

<div>

<button (click)="add()">

Add

</button>

</div>

<ul class="books">

<li \*ngFor="let b of books; let i=index;">

<a routerLink="/book-edit/{{b.bookId}}">

<span class="badge">{{i+1}}</span> {{b.title}}

</a>

<button class="delete" title="delete book"

(click)="delete(b.bookId)">x</button>

</li>

</ul>

**book-add.component.ts**

import { Component, OnInit, Input } from '@angular/core';

import { RestService } from '../rest.service';

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

@Component({

selector: 'app-book-add',

templateUrl: './book-add.component.html',

styleUrls: ['./book-add.component.css']

})

export class BookAddComponent implements OnInit {

@Input() bookData = { title:'', price: 0, volume: 0, publishDate: '' };

constructor(public rest:RestService, private route: ActivatedRoute, private router: Router) { }

ngOnInit() {

}

addBook() {

this.rest.addBook(this.bookData).subscribe((result) => {

this.router.navigate(['/books']);

}, (err) => {

console.log(err);

});

}

}

**book-detail.component.ts**

import { Component, OnInit } from '@angular/core';

import { RestService } from '../rest.service';

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

@Component({

selector: 'app-book-detail',

templateUrl: './book-detail.component.html',

styleUrls: ['./book-detail.component.css']

})

export class BookDetailComponent implements OnInit {

book:any;

constructor(public rest:RestService, private route: ActivatedRoute, private router: Router) { }

ngOnInit() {

this.rest.getBook(this.route.snapshot.params['id']).subscribe((data: {}) => {

console.log(data);

this.book = data;

});

}

}

**book-detail.component.html**

<div \*ngIf="book" class="books">

<h2>{{book.title | uppercase}} Details</h2>

<div><span>Book Title: </span>{{book.title}}</div>

<div><span> Book Price: </span>{{book.price}}</div>

<div><span>Book Volume: </span>{{book.volume}}</div>

<div><span>Book Publish Date: </span>{{book.publishDate | date}}</div>

<div>

<button routerLink="/book-edit/{{book.bookId}}">

Edit

</button>

</div>

</div>

**book-edit.component.ts**

import { Component, OnInit, Input } from '@angular/core';

import { RestService } from '../rest.service';

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

@Component({

selector: 'app-book-edit',

templateUrl: './book-edit.component.html',

styleUrls: ['./book-edit.component.css']

})

export class BookEditComponent implements OnInit {

@Input() bookData:any = { title: '', price: 0, volume:0 , publishDate:''};

constructor(public rest:RestService, private route: ActivatedRoute, private router: Router) { }

ngOnInit() {

this.rest.getBook(this.route.snapshot.params['id']).subscribe((data: {}) => {

console.log(data);

this.bookData = data;

});

}

updateBook() {

this.rest.updateBook(this.route.snapshot.params['id'], this.bookData).subscribe((result) => {

this.router.navigate(['/books']);

}, (err) => {

console.log(err);

});

}

}

**book-edit.component.html**

<div>

<h2>Book Edit</h2>

<div>

<label>Book Name:

<input [(ngModel)]="bookData.title" placeholder="Book Name"/>

</label><br>

<label>Book Price:

<input [(ngModel)]="bookData.price" placeholder="Book Price"/>

</label><br>

<label>Book Volume:

<input [(ngModel)]="bookData.volume" placeholder="Book Volume"/>

</label><br>

<label>Book Publish Date:

<input [(ngModel)]="bookData.publishDate" placeholder="Book Published Date"/>

</label><br>

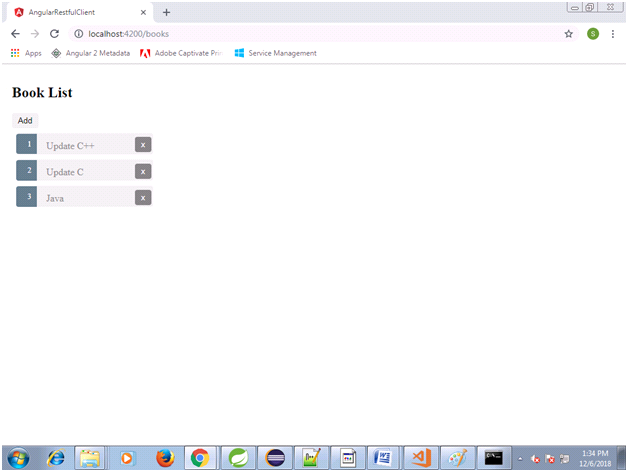
</div>

<button (click)="updateBook()"></button>

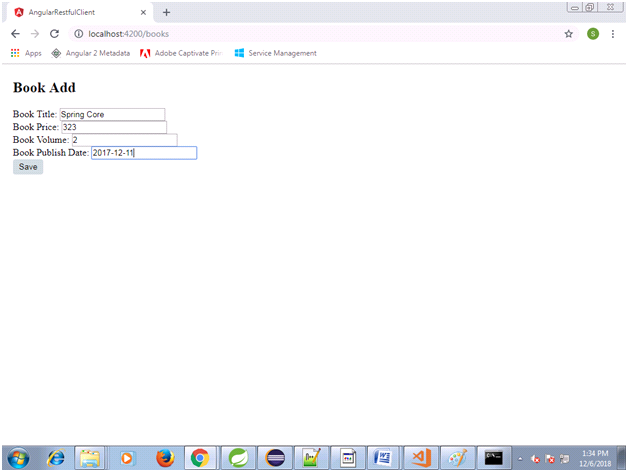
</div>

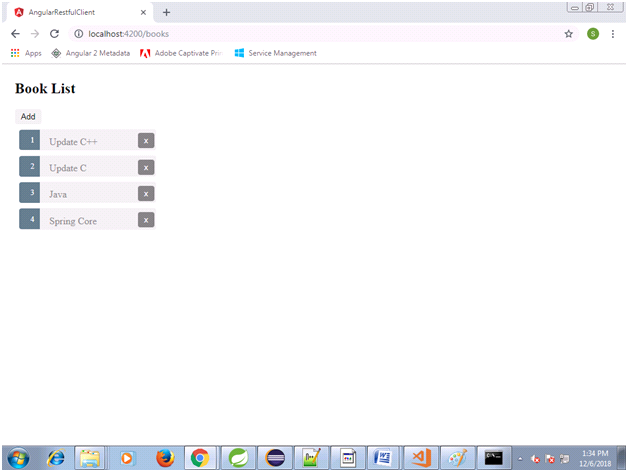
**Screenshots:**

**getBook() List:**

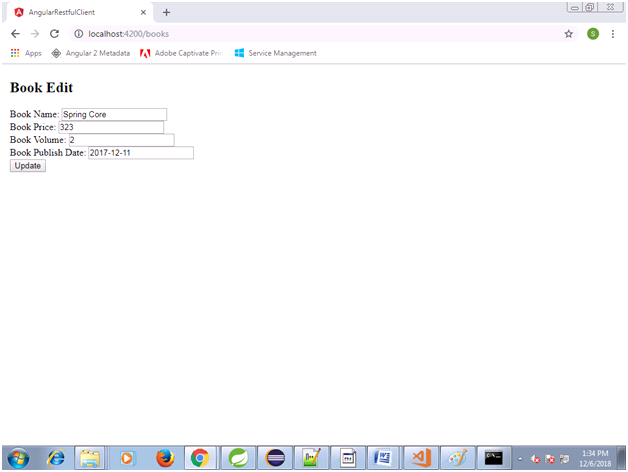
****

**Add Book**

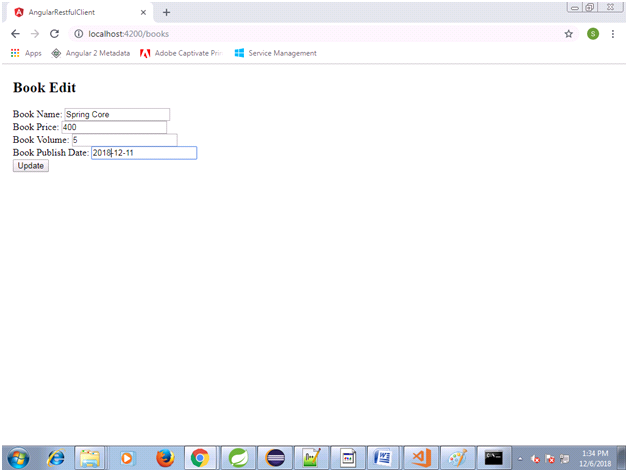
****

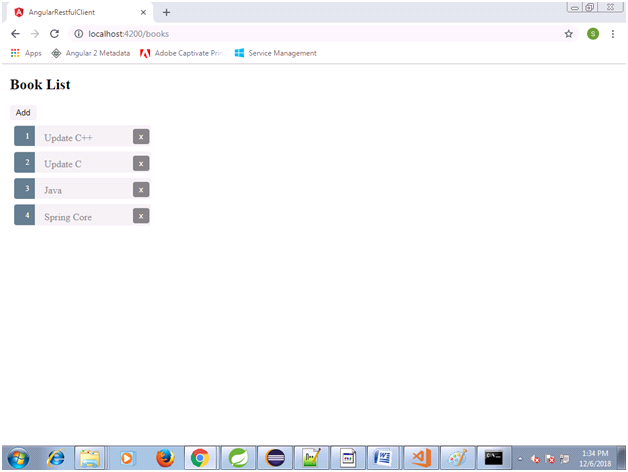
****

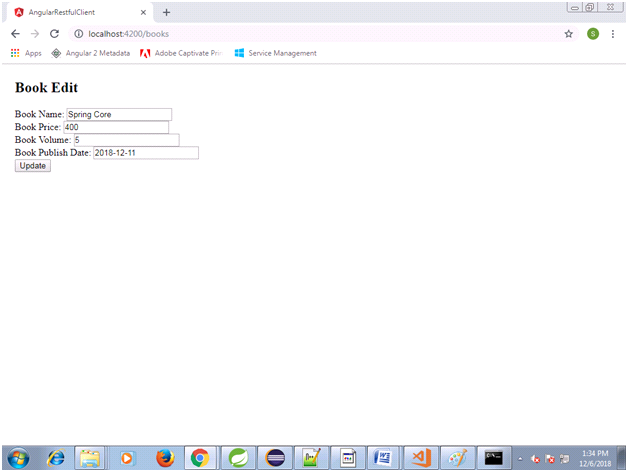
**getBook() Details**

****

**Edit Book:**

****

****

****