REACT +Axios+Spring Boot

GET,
POST,
DELETE

HOW TO:

- **Don’t try to code along with me! :)**
- **Watch the entire process all the way through.**
- **Take note of new terminology, interesting points, or things you don’t understand.**

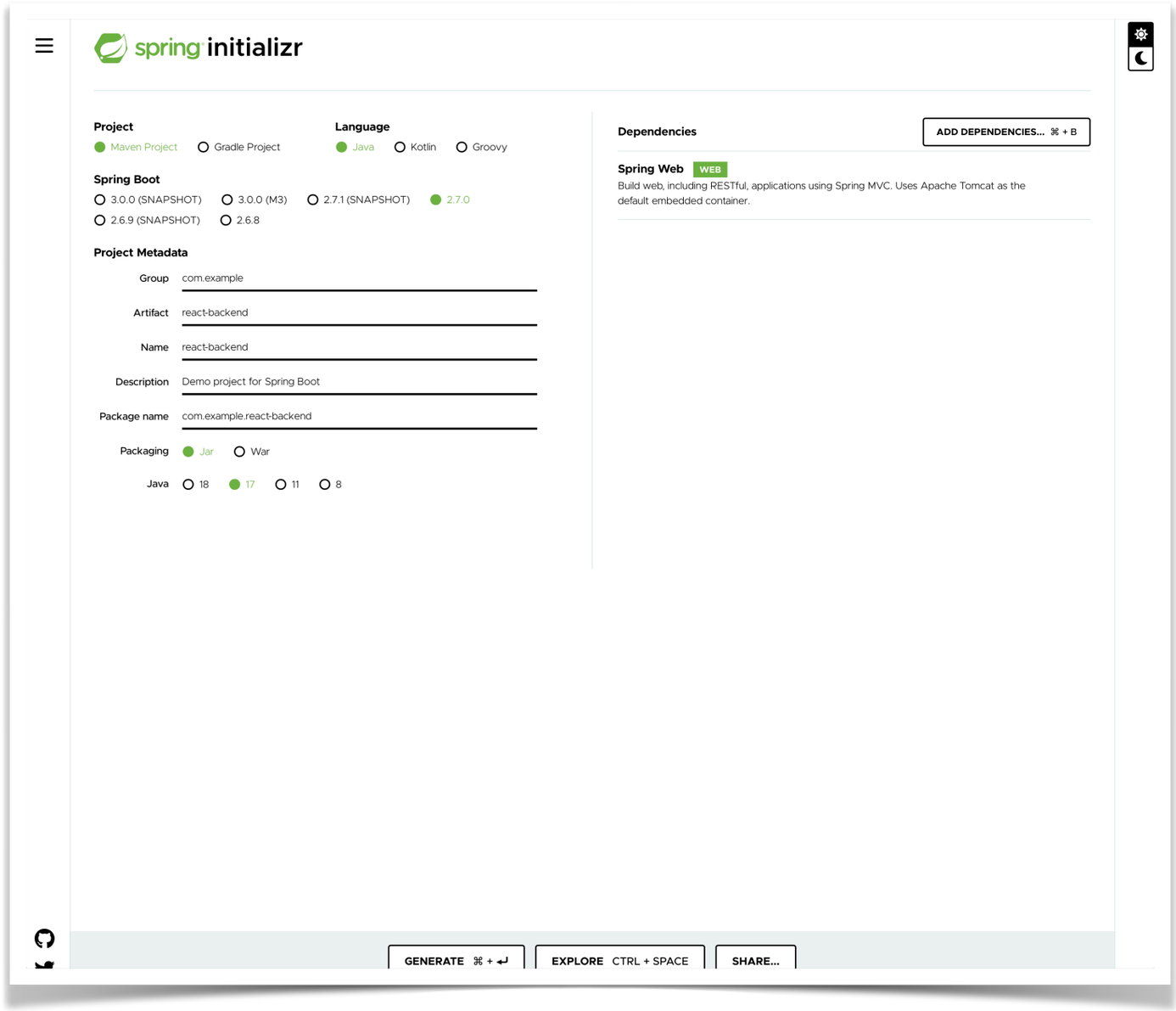
- **Code the example for yourself.**

- **Create your own project based on what you've learned.**

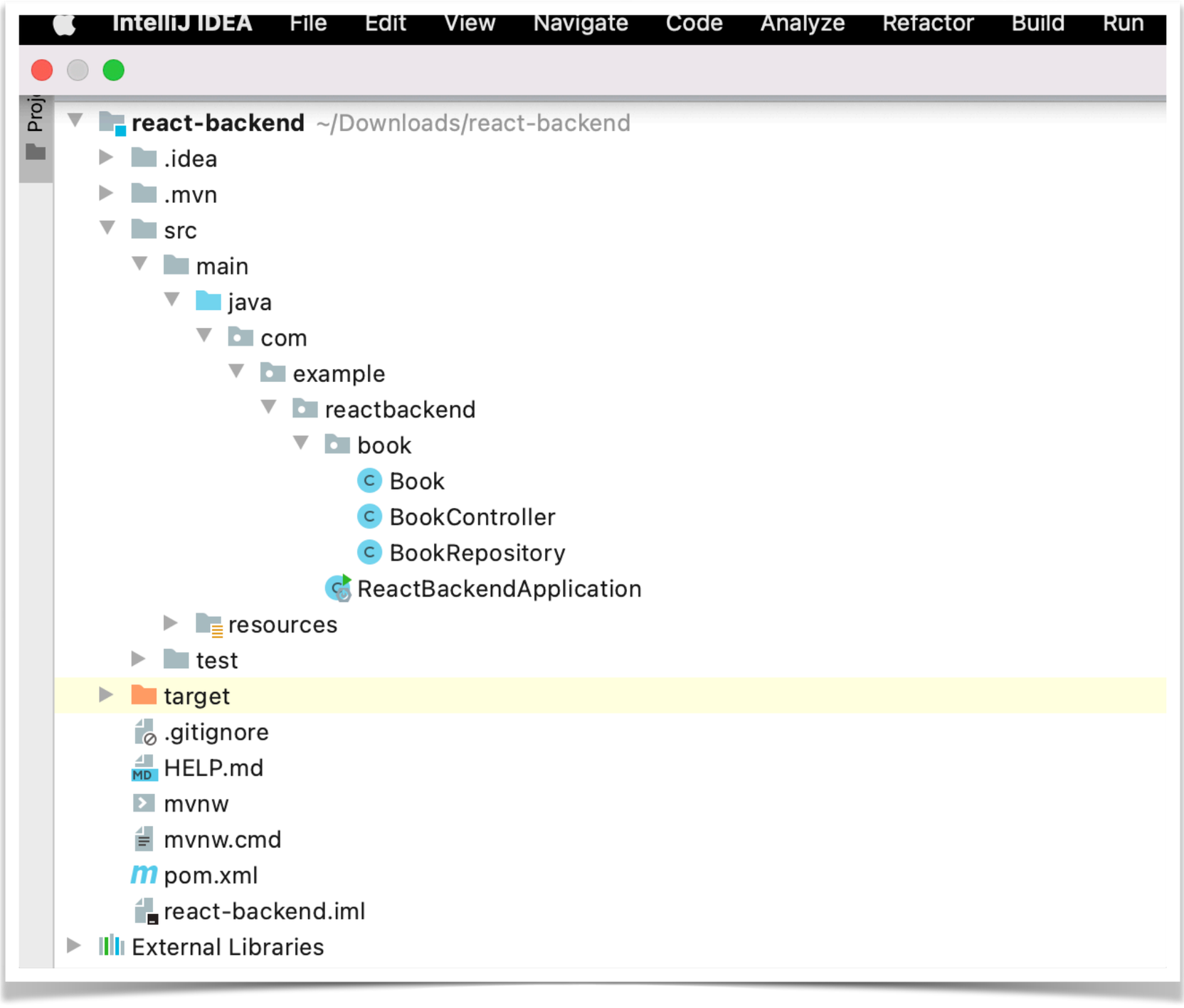
The backend: Spring Boot

- Spring Initializr
- Model+Controller [CORS!]+Repo

Spring Initializr : WEB



Project's classes



BookRepositoryTest

```java
class BookRepositoryTest {

    BookRepository bookRepository = new BookRepository();

    @Test
    void getAll() {
        assertEquals(4, bookRepository.getAll().size());
    }

    @Test
    void get() {
        assertEquals("The Dawn of Everything", bookRepository.get(2L).getTitle());
    }

    @Test
    void add() {
        assertEquals(4, bookRepository.getAll().size());
        var title = "Title";
        var author = "Author";
        bookRepository.add(new Book(1L, "Title", "Author"));
        assertAll(
                () -> assertEquals(5, bookRepository.getAll().size()),
                () -> assertEquals(title, bookRepository.get(5L).getTitle()),
                () -> assertEquals(author, bookRepository.get(5L).getAuthor())
        );
    }

    @Test
    void delete() {
        assertEquals(4, bookRepository.getAll().size());
        bookRepository.delete(1L);
        assertEquals(3, bookRepository.getAll().size());
    }

    @Test
    void deleteWithNonExistingId() {
        NoSuchElementException thrown = Assertions.assertThrows(NoSuchElementException.class, () -> bookRepository.delete(999L));
    }
}
```

Project's classes: Book

```java
package com.example.reactbackend.book;

public class Book {

    private Long id;
    private String title;
    private String author;

    public Book() {
    }

    public Book(Long id, String title, String author) {
        this.id = id;
        this.title = title;
        this.author = author;
    }

    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getTitle() {
        return title;
    }

    public void setTitle(String title) {
        this.title = title;
    }

    public String getAuthor() {
        return author;
    }

    public void setAuthor(String author) {
        this.author = author;
    }
}
```

Project's classes: BookController

```java
package com.example.reactbackend.book;

import org.springframework.web.bind.annotation.*;

import java.util.Collection;

@RestController
@RequestMapping("/books")
@CrossOrigin(origins = "http://localhost:3000")
public class BookController {

    private BookRepository bookRepository;

    public BookController(BookRepository bookRepository) {
        this.bookRepository = bookRepository;
    }


    @GetMapping
    Collection<Book> getAll() {
        return bookRepository.getAll();
    }

    @GetMapping("/{id}")
    Book get(@PathVariable Long id) {
        return bookRepository.get(id);
    }

    @PostMapping()
    void delete(@RequestBody Book book) {
        bookRepository.add(book);
    }
    @DeleteMapping("/{id}")
    void delete(@PathVariable Long id) {
        bookRepository.delete(id);
    }
}
```

Project's classes: BookRepository

```java
package com.example.reactbackend.book;

import org.springframework.stereotype.Repository;

import java.util.*;

@Repository
public class BookRepository {
    private Map<Long, Book> map = new HashMap();
    {
        map.put(1L, new Book(1l, "From Animals into Gods: A Brief History of Humankind", "Yuval Noah Harari"));
        map.put(2L,  new Book(2l, "The Dawn of Everything", "David Graeber & David Wengrow"));
        map.put(3L, new Book(3l, "How to Change Your Mind", "Michael Pollan"));
        map.put(4L, new Book(4l, "Intuition Pumps and Other Tools for Thinking", "Daniel C. Dennett"));
    }

    public Collection<Book> getAll() {
        return map.values();
    }

    public Book get(Long id){
        return map.get(id);
    }

    public void add(Book book){
        var nextId = Long.valueOf(map.size()+1);
        book.setId(nextId);
        map.put(nextId, book);
    }

    public void delete(Long id) {
        if(!map.containsKey(id)) {
            throw new NoSuchElementException("No element with given id: " + id);
        }
        map.remove(id);
    }

}
```

The frontend: React + Axios

Generate a React project & install Axios

```
npx create-react-app react-axios-example
cd react-axios-example/
npm install axios
```

Generate a React component: BookList

```
npx generate-react-cli component BookList
```

The first time

```
? Does this project use TypeScript? No
? Does this project use CSS modules? No
? Does this project use a CSS Preprocessor? css
? What testing library does your project use? None
? Set the default path directory to where your components will be generated in? src/components
? Would you like to create a corresponding stylesheet file with each component you generate? Yes
? Would you like to create a corresponding test file with each component you generate? Yes
? Would you like to create a corresponding story with each component you generate? No
? Would you like to create a corresponding lazy file (a file that lazy-loads your component out of the box an
d enables code splitting: https://reactjs.org/docs/code-splitting.html#code-splitting) with each component yo
u generate? No
```

Update BookList: useState, useEffect, axios

```
import { useState, useEffect } from 'react';
import axios from 'axios';

const BookList = () => {
  const [books, setBooks] = useState([]);

  useEffect(() => {
          axios.get('http://127.0.0.1:8080/books')
            .then(res => {
              const books = res.data;
              setBooks(books);
            }).catch((err) => console.log(err));
    }, []
  );
  return (
      <ul>
        {
            books.map(book =>
              <li key={book.id}>{`${book.title} | ${book.author}`}</li>
            )
        }
      </ul>
  );
}

export default BookList;
```

Update App: BookList

```
import BookList from "./components/BookList/BookList";

function App() {
  return (
    <BookList />
  );
}

export default App;
```

npx generate-react-cli component BookAdd
Update BookAdd

```
import { useState } from 'react';
import axios from 'axios';

const BookAdd = () => {
  const [title, setTitle] = useState('');
  const [author, setAuthor] = useState('');


  const handleTitleChange = event => setTitle(event.target.value);
  const handleAuthorChange = event => setAuthor(event.target.value);

  const handleSubmit = event => {
    event.preventDefault();

    const book = {
      "title": title,
      "author": author
    }

    axios.post('http://127.0.0.1:8080/books', book)
      .then(res => {
        console.log(res);
      }).catch((err) => console.log(err));
  }

  return (
    <div>
      <form onSubmit={handleSubmit}>
        <label>
          Title:
          <input type="text" name="title" onChange={handleTitleChange} />
        </label>
        <label>
          Author:
          <input type="text" name="author" onChange={handleAuthorChange} />
        </label>
        <button type="submit">Add</button>
      </form>
    </div>
  );

}

export default BookAdd;
```

Update App

```
import BookList from "./components/BookList/BookList";
import BookAdd from "./components/BookAdd/BookAdd";

function App() {
  return (
    <>
    <BookList />
    <BookAdd />
    </>
  );
}

export default App;
```

Axios vs Fetch

https://blog.logrocket.com/axios-vs-fetch-best-http-requests/

Spring Boot CORS

https://www.stackhawk.com/blog/spring-cors-guide/