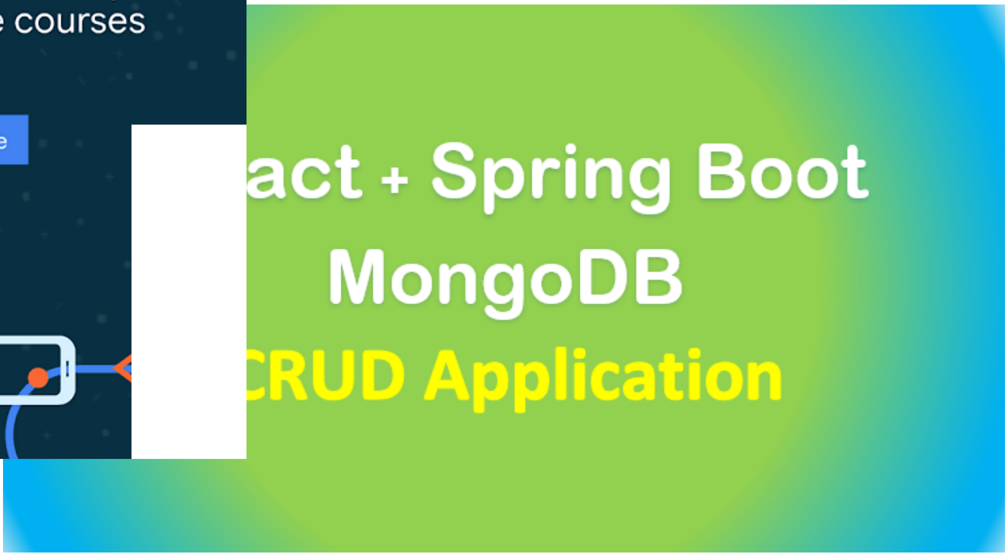


Spring Boot, React & MongoDB example: Build a CRUD Application

Last modified: May 13, 2020 (https://bezkoder.com/react-spring-boot-mongodb/) bezkoder (https://bezkoder.com/author/bezkoder/) Full Stack (https://bezkoder.com/category/full-stack/), MongoDB (https://bezkoder.com/category/mongodb/), React (https://bezkoder.com/category/react/), Spring (https://bezkoder.com/category/spring/)

In this tutorial, we will learn how to build a full stack CRUD App example using Spring Boot, React and MongoDB. The back-end server uses Spring Boot with Spring Web MVC for REST APIs and Spring Data MongoDB. The front-end side will be made with React, React Router, Axios & Bootstrap.

Related Post:
– Spring Boot + React: Login example with JWT Authentication & Spring Security (https://bezkoder.com/spring-boot-react-jwt-auth/)
– React Upload/Download Files to/from Spring Boot Rest Apis (https://bezkoder.com/react-file-upload-spring-boot/)

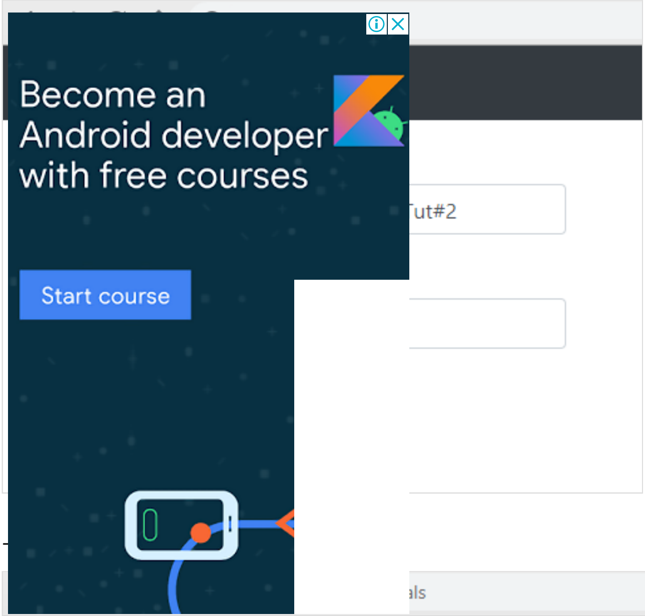
Contents [hide]

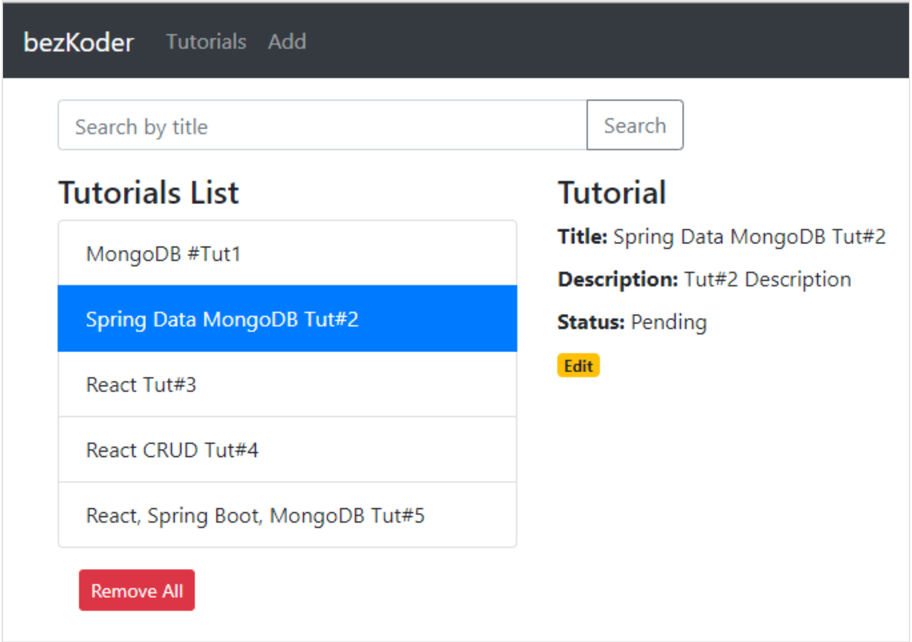
Spring Boot, React, MongoDB example Overview

We will build a full-stack Tutorial Application in that:

- Each Tutorial has id, title, description, published status.
- We can create, retrieve, update, delete Tutorials.
- We can also find Tutorials by title.

– Create a Tutorial:





Tutorials List

- MongoDB #Tut1
- Spring Data MongoDB Tut#2
- React Tut#3
- React CRUD Tut#4
- React, Spring Boot, MongoDB Tut#5

Remove All

Tutorial

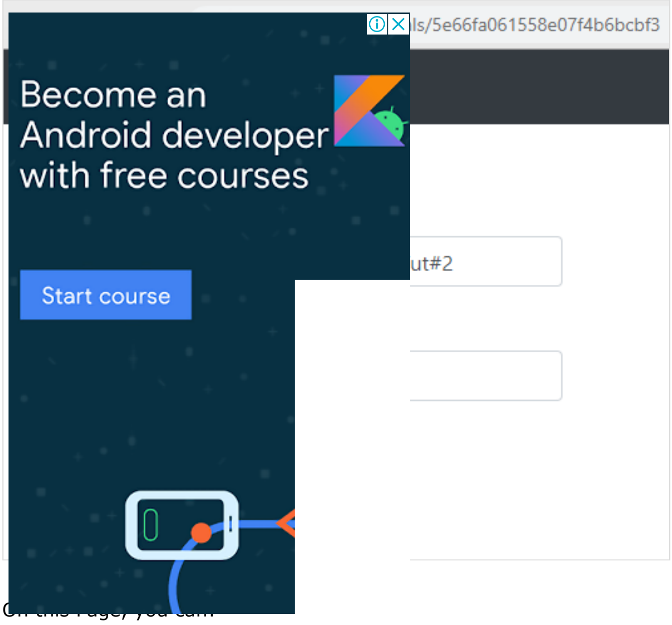
Title: Spring Data MongoDB Tut#2

Description: Tut#2 Description

Status: Pending

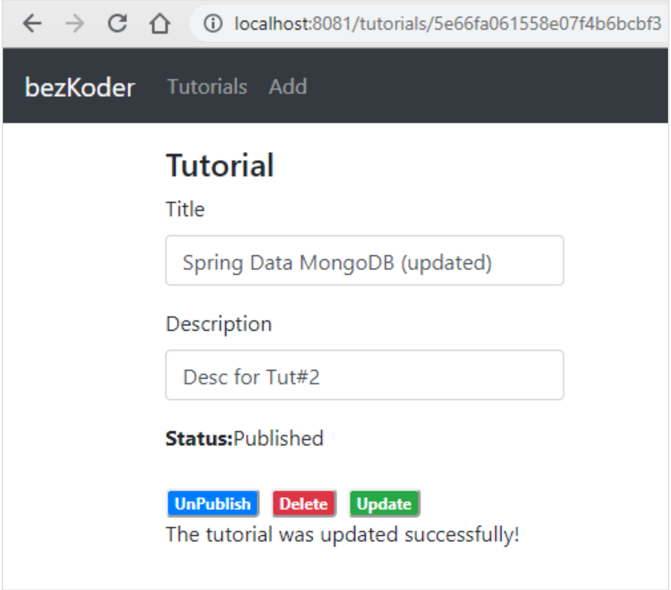
Edit

– Click on **Edit** button to view a Tutorial:



On this Page, you can:

- change status to **Published** using **Publish** button
- remove the Tutorial from Database using **Delete** button
- update the Tutorial details on Database with **Update** button



Tutorial

Title

Spring Data MongoDB (updated)

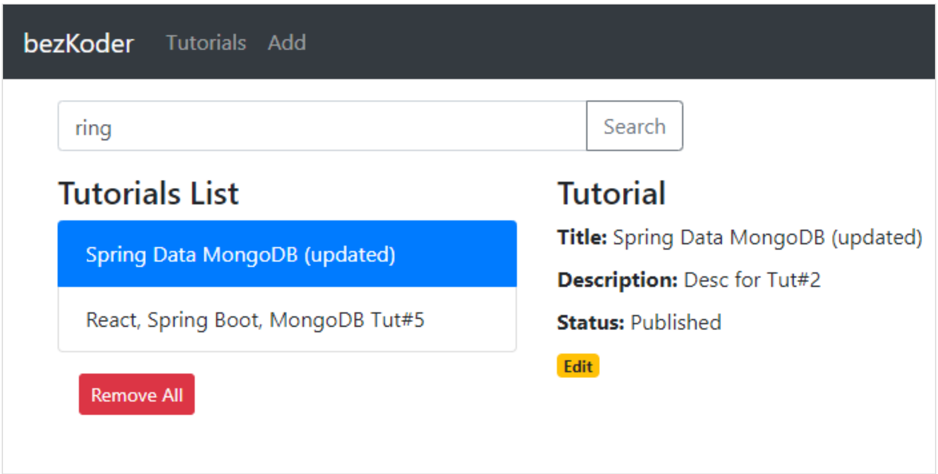
Description

Desc for Tut#2

Status: Published

The tutorial was updated successfully!

– Search Tutorials by title:



Tutorials List

Spring Data MongoDB (updated)

React, Spring Boot, MongoDB Tut#5

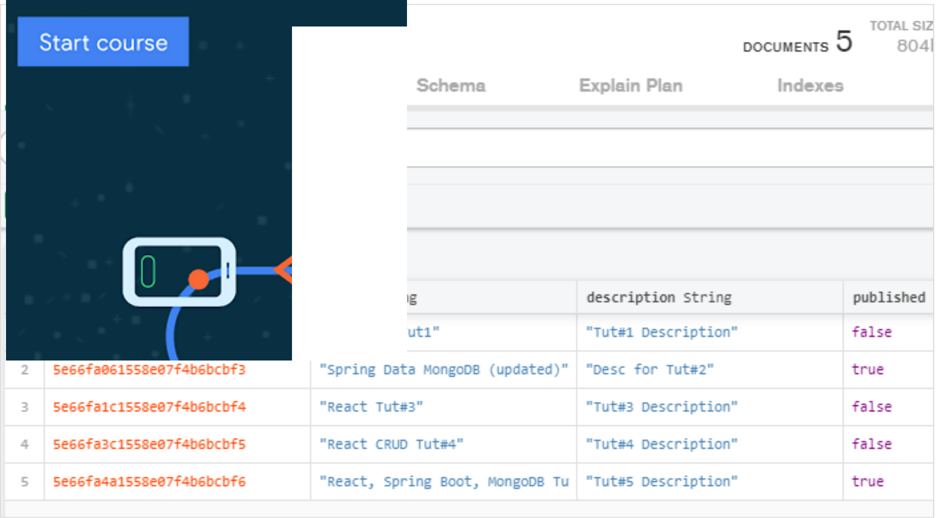
Tutorial

Title: Spring Data MongoDB (updated)

Description: Desc for Tut#2

Status: Published

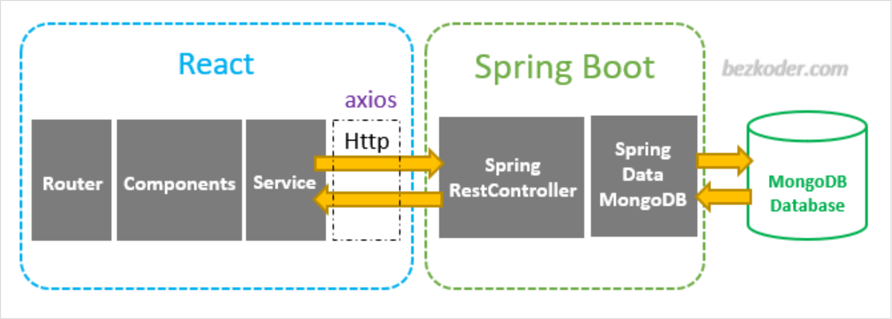
– Check MongoDB Database for all items:



	_id	g	description String	published
		ut1"	"Tut#1 Description"	false
2	5e66fa061558e07f4b6bcbf3	"Spring Data MongoDB (updated)"	"Desc for Tut#2"	true
3	5e66fa1c1558e07f4b6bcbf4	"React Tut#3"	"Tut#3 Description"	false
4	5e66fa3c1558e07f4b6bcbf5	"React CRUD Tut#4"	"Tut#4 Description"	false
5	5e66fa4a1558e07f4b6bcbf6	"React, Spring Boot, MongoDB Tu	"Tut#5 Description"	true

Spring Boot, React and MongoDB Architecture

We're gonna build the system with following architecture:



– Spring Boot exports REST Apis using Spring Web MVC & interacts with MongoDB Database using Spring Data MongoDB
– React Client sends HTTP Requests and retrieve HTTP Responses using **axios**, shows data on the components. We also use React Router for navigating to pages.

Spring Boot Back-end

Overview

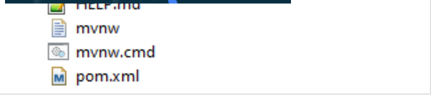
These are APIs that Spring Boot App will export:

Methods	Urls	Actions
POST	/api/tutorials	create new Tutorial
GET	/api/tutorials	retrieve all Tutorials
GET	/api/tutorials/:id	retrieve a Tutorial by `:id`
PUT	/api/tutorials/:id	update a Tutorial by `:id`
DELETE	/api/tutorials/:id	delete a Tutorial by `:id`
DELETE	/api/tutorials	delete all Tutorials
GET	/api/tutorials?title=[keyword]	find all Tutorials which title contains `keyword`

We make CRUD operations & finder methods with Spring Data MongoDB's `MongoRepository` .

Technology

Spring Data MongoDB)



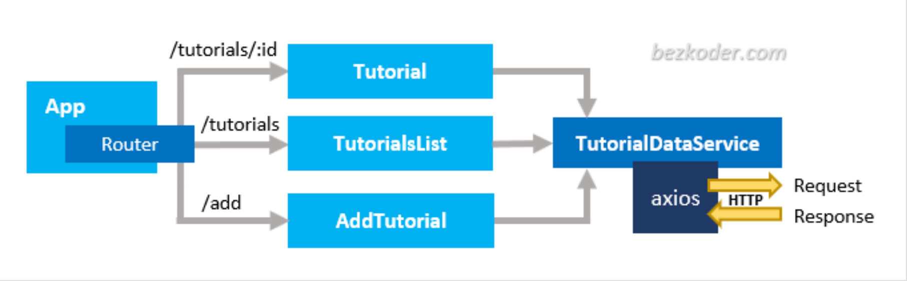
- `Tutorial` data model class corresponds to entity and table *tutorials*.
- `TutorialRepository` is an interface that extends MongoRepository (https://docs.spring.io/spring-data/mongodb/docs/current/api/org/springframework/data/mongodb/repository/MongoRepository.html) for CRUD methods and custom finder methods. It will be autowired in `TutorialController` .
- `TutorialController` is a RestController (https://docs.spring.io/spring/docs/current/javadoc-api/org/springframework/web/bind/annotation/RestController.html) which has request mapping methods for RESTful requests such as: *getAllTutorials*, *createTutorial*, *updateTutorial*, *deleteTutorial*, *findByPublished*…
- Configuration for Spring Data MongoDB is in **application.properties**.
- **pom.xml** contains dependencies for Spring Boot Web MVC and Spring Data MongoDB.

Implementation

You can find step by step to implement this Spring Boot Server Application in the post:
Spring Boot with MongoDB CRUD example using Spring Data (https://bezkoder.com/spring-boot-mongodb-crud/)

React.js Front-end

Overview



– The `App` component is a container with React `Router` . It has `navbar` that links to routes paths.

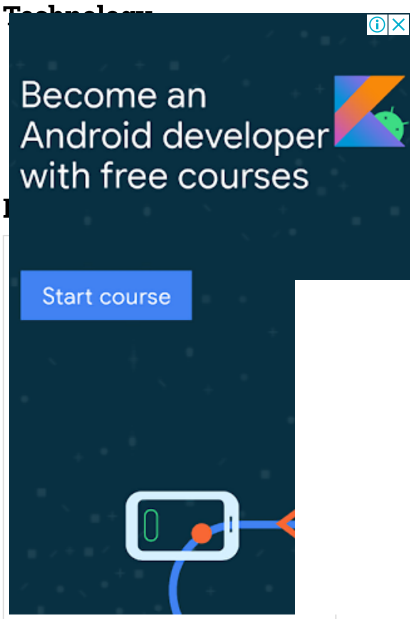
– `TutorialsList` component gets and displays Tutorials.

– `Tutorial` component has form for editing Tutorial’s details based on `:id` .

– `AddTutorial` component has form for submission new Tutorial.

– These Components call `TutorialDataService` methods which use `axios` to make HTTP requests and receive responses.

Technology



– **package.json** contains 4 main modules: `react` , `react-router-dom` , `axios` & `bootstrap` .
– `App` is the container that has `Router` & navbar.
– There are 3 components: `TutorialsList` , `Tutorial` , `AddTutorial` .
– **http-common.js** initializes axios with HTTP base Url and headers.
– `TutorialDataService` has methods for sending HTTP requests to the Apis.
– **.env** configures *port* for this React CRUD App.

Implementation

You can find step by step to implement this React App in the post:
– React.js CRUD example to consume Web API (https://bezkoder.com/react-crud-web-api/)
– or React Hooks CRUD example to consume Web API (https://bezkoder.com/react-hooks-crud-axios-api/)

Further Reading

- Spring Data MongoDB (https://docs.spring.io/spring-data/mongodb/docs/current/reference/html/#reference)
- Spring Web MVC (https://docs.spring.io/spring/docs/current/spring-framework-reference/web.html)
- React Component (https://reactjs.org/docs/react-component.html)
- https://www.npmjs.com/package/axios (https://www.npmjs.com/package/axios)

Conclusion

Now we have an overview of our Spring Boot React MongoDB CRUD example.

We also take a look at client-server architecture for REST API using Spring Web MVC & Spring Data MongoDB, as well as React project structure for building a front-end app to make HTTP requests and consume responses.

Next tutorials show you more details about how to implement the system (including source code):
– Back-end (https://bezkoder.com/spring-boot-mongodb-crud/)
– Front-end:

- Using React Components (https://bezkoder.com/react-crud-web-api/)
- Using React Hooks (https://bezkoder.com/react-hooks-crud-axios-api/)

Happy learning, see you again!

//bezkoder.com/tag/react/)   react router (https://bezkoder.com/tag/react-router/)

rest api (https://bezkoder.com/tag/rest-api/)   spring boot (https://bezkoder.com/tag/spring-boot/)   spring data mongodb (https://bezkoder.com/tag/spring-data-mongodb/)

2 thoughts to “Spring Boot, React & MongoDB example: Build a CRUD Application”

Sampada
July 24, 2020 at 11:58 pm (https://bezkoder.com/react-spring-boot-mongodb/#comment-3932)

Where can I find source code for this tutorial?

REPLY

bezkoder
July 25, 2020 at 1:14 am (https://bezkoder.com/react-spring-boot-mongodb/#comment-3934)

Hi, please visit the next tutorials in Conclusion section.

REPLY

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Name *

Email *

Website

□

Save my name, email, and website in this browser for the next time I comment.

POST COMMENT

< © /bezkoder.com/spring-boot-restcontrolleradvice/)

React + Node.js + Express + MySQL example: Build a CRUD App > (https://bezkoder.com/react-node-express-mysql/)

CATEGORIES

- Android (https://bezkoder.com/category/android/)
- Angular (https://bezkoder.com/category/angular/)
- Dart (https://bezkoder.com/category/dart/)
- Deployment (https://bezkoder.com/category/deployment/)
- Django (https://bezkoder.com/category/django/)
- Firebase (https://bezkoder.com/category/firebase/)
- Flutter (https://bezkoder.com/category/flutter/)
- Full Stack (https://bezkoder.com/category/full-stack/)
- GraphQL (https://bezkoder.com/category/graphql/)
- Java (https://bezkoder.com/category/java/)
- Kotlin (https://bezkoder.com/category/kotlin/)
- MongoDB (https://bezkoder.com/category/mongodb/)
- Node.js (https://bezkoder.com/category/node-js/)
- Python (https://bezkoder.com/category/python/)
- React (https://bezkoder.com/category/react/)
- Security (https://bezkoder.com/category/security/)
- Spring (https://bezkoder.com/category/spring/)
- Testing (https://bezkoder.com/category/testing/)
- Uncategorized (https://bezkoder.com/category/uncategorized/)
- Vue.js (https://bezkoder.com/category/vue/)

Hamed Bakayok

FOLLOW US

We use cookies to improve your experience with the site. To find out more, you can read the full

Privacy & Policy (https://bezkoder.com/privacy-policy/) Accept

(https://www.youtube.com/channel/UCp0mx9RH0JxaFsmMvKXA86Q)

(https://facebook.com/bezkoder)

(https://github.com/bezkoder)

TOOLS

Json Formatter (https://bezkoder.com/json-formatter/)

Home (https://bezkoder.com/) Privacy Policy (https://bezkoder.com/privacy-policy/) Contact Us (https://bezkoder.com/contact-us/)

About Us (https://bezkoder.com/about/)

BezKoder 2019