**Mastering full stack React web development**

**Create enterprise level highly scalable web applications using ReactJS, Redux, NodeJS, ExpressJS, MongoDB & FalcorJS by Tomasz Dyl, Kamil Przeorski and Maciej Czarnecki**

# **Audience**

*An audience of this book are developers familiar with building ReactJS client-side applications supported by FLUX architecture and want to scale it to enterprise-grade products using NodeJS, ExpressJS, MongoDB and FalcorJS stack. The book will also attract developers willing to learn Redux - predictable state container which evolves ideas of FLUX architecture and functional programming paradigms.*

# **Mission**

*Provide a full guideline for creating enterprise level applications from project planning and first architectural decisions through deployment process, promote React.js, Redux and FalcorJS as modern tools for building rich isomorphic applications with server side rendering using ExpressJS and NodeJS. Help product owners and lead developers with maintaining and updating already existing project by suggesting set of tools for monitoring and keeping the project alive with minimizing the risk of application downtime.*

# **Objectives and achievements**

*A reader after finishing this book should be able not only to create rich application, but also get it through deployment process, making use of modern virtualization tools like Docker containers.*

*The book provides a starter-kit project for a client-side starter-kit which will keep interested more advanced audience as the client-side stack will be mostly ready for a reader with basic knowledge in ReactJS. I also allows for using it as a reference in reader’s future projects*

*Each chapter is covering essential things in certain area of the expertise, containing detailed information about process of delivering web application - starting from environment preparation till deployment automation tools and continuous integration, allowing for further development of a product and picking right development methodology for the app.*

# **General structure**

1. “Hello world” - Configuring the full-stack with NodeJS, ExpressJS, MongoDB, Mongoose, Falcor and Redux
2. Creating a “simple login/register application” with the authentication & authorisation
3. Adding the server-side rendering to our “simple app”
4. Creating the Add/Edit/Publish articles on the front-end
5. Falcor.js - start using Falcor.browser.js on the client-side
6. NodeJS/ExpressJS with falcor-express - moving the Falcor’s model from client-side to backend side, using falcor-router
7. MongoDB/Mongoose - creating the structure of our noSQL db, preparing collections
8. Adding Mongoose in our express-falcor’s routes
9. Deployment with Amazon AWS
10. Continuous integration

# **Detailed outline**

# Chapter 1: “Hello world” - Configuring the full-stack with NodeJS, ExpressJS, MongoDB, Mongoose, Falcor and Redux - 50 pages

## Description

## Level

*MEDIUM*

## Topics covered

*What technical topics will the reader learn about during this chapter?*

* *Introduction to the project’s app*
* *installing npm, node, expressjs, mongodb*
* *Introduction & approaching the project*

## Skills learned

*A reader is able to create a simple node, express, mongodb server with an example API endpoint. User will be able to create a simple “hello world” application. He will also be able to get big picture idea about Redux*

# Chapter 2: Creating a “simple login/register application” with the authentication & authorisation - 50 pages

## Description

*Making from scratch the full-stack app that has login and register feature.*

## Level

*ADVANCED*

## Topics covered

* *Authentication with using express-jwt*
* *Registration and login feature*

## Skills learned

*A reader will create a simple app with login/registration feature*

# Chapter 3: Adding the server-side rendering to our “simple app” - 20 pages

## Description

*Adding the server-side rendering to our “simple app”*

## Level

*MEDIUM*

## Topics covered

* *server rendering with redux*

## Skills learned

On top of the simple login/registration app a reader will add pieces responsible for server-rendering with redux

# Chapter 4: Creating the Add/Edit/Publish articles on the front-end - 50 pages

## Description

*On top of the simple app (login/register) a reader will add add/edit/publish feature to the app.*

## Level

*MEDIUM*

## Topics covered

* Adding articles
* Editing articles
* Reading articles

## Skills learned

*A reader is able to create the client-side (without backend, yet) publishing app.*

# Chapter 5: Falcor.js - start using Falcor.browser.js on the client-side — 50 pages

## Description

*Starting coding on the client-side. We start with creating model’s for our app, then starting coding views and connecting them with the Falcor’s data model in order to create fully-functional client-side application.*

## Level

*ADVANCED*

## Topics covered

* *Falcor JS and it’s concepts*
* *Practical guide how to add Falcor’s on the client-side*

## Skills learned

*After finishing this chapter a reader will have the client-side publishing app that is using falcor.browser.js library (on the front-end only).*

# Chapter 6: NodeJS/ExpressJS with falcor-express - moving the Falcor’s model from client-side to backend side, using falcor-router - 50 pages

## Description

*Starting coding the backend and moving the Falcor’s model to the back-end. Guide to falcor-router and falcor-express.*

## Level

*ADVANCED*

## Topics covered

* NodeJS
* ExpressJS
* Creating Falcor’s Virtual JSON
* Falcor-express from npm
* Falcor-router from npm

## Skills learned

*After finishing this chapter, a reader will have client & backend side publishing app (without MongoDB/Mongoose yet).*

# Chapter 7: MongoDB/Mongoose - creating the structure of our noSQL db, preparing collections - 30 pages

## Description

*NodeJS/ExpressJS configuration related to MongoDB and Mongoose. Creating our publishing app collections in MongoDB.*

## Level

*ADVANCED*

## Topics covered

* Structure of MongoDB
* First Mongoose call on an example Falcor’s route

## Skills learned

*This chapter will prepare user to mix the Mongoose’s related code with falcor-router.*

# Chapter 8: Adding Mongoose in our express-falcor’s routes - 20 pages

## Description

Wrapping up the Mongoose with falcor-express library in order to finish our full-stack publishing app.

## Level

*ADVANCED*

## Topics covered

* Creating our routes in Falcor and making ready/edit’s directly on the MongoDB

## Skills learned

*This chapter will wrap-up everything together so we will have complete publishing app.*

# Chapter 9: Deployment with Docker & Amazon AWS - 50 pages

## Description

*A reader will learn how to provision, operate and manage distributed application systems on AWS with use of Docker.*

## Level

*ADVANCED*

## Topics covered

* Implement and manage continuous delivery systems and methodologies on AWS
* Understand, implement, and automate security controls, governance processes, and compliance validation
* Define and deploy monitoring, metrics, and logging systems on AWS
* Implement systems that are highly available, scalable, and self-healing on the AWS platform
* Design, manage, and maintain tools to automate operational processes
* Docker

## Skills learned

*A reader will have a complete guide and skills on how to deploy his full-stack app with use of Docker on the Amazon AWS.*

# Chapter 10: Continuous integration - 30 pages

## Description

*Approaching application state monitoring, updates and bug fixes. Description of concepts and tools for seamless version bumping, bug tracking and logging systems.*

## Level

*MEDIUM*

## Topics covered

* Monitoring tools
* Updating live application
* Logger systems

## Skills learned

*A reader will be able to successfully maintain the application, and handle it’s update process with minimizing the risk of possible application downtime.*