```
<html ng-app="myApp">
<head>
  <title>BookStore</title>
  <link rel="stylesheet" href="bower_components/bootstrap/dist/css/bootstrap.css">
    <link rel="stylesheet" href="css/style.css">

</head><body>

  <nav class="navbar navbar-default">
      <div class="container">
        <div class="navbar-header">
          <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-
target="#navbar" aria-expanded="false" aria-controls="navbar">
            <span class="sr-only">Toggle navigation</span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
          </button>
          <a class="navbar-brand" href="#">Bookstore</a>
        </div>
        <div id="navbar" class="collapse navbar-collapse">
          <ul class="nav navbar-nav navbar-right">
            <li><a href="#/books/add">Add Book</a></li>
          </ul>
        </div><!--/.nav-collapse -->
      </div>
    </nav>

    <div class="container">
      <div class="row">
        <div class="col-md-12">
          <div ng-view></div>
        </div>
      </div>
     </div>

    </div><!-- /.container -->
  <script src="/bower_components/jquery/dist/jquery.js"></script>
  <script src="/bower_components/angular/angular.js"></script>
  <script src="/bower_components/angular-route/angular-route.js"></script>
  <script src="/bower_components/bootstrap/dist/js/bootstrap.js"></script>
  <script src="/app.js"></script>
  <script src="/controllers/books.js"></script>
  <script src="/controllers/genres.js"></script>
</body>
</html>
```

APP.JS

```
const express = require('express');
const app = express();
```

```
const bodyParser = require('body-parser');
const mongoose = require('mongoose');

app.use(express.static(__dirname+'/client'));
app.use(bodyParser.json());

Genre =require('./models/genre');
Book =require('./models/book');

// Connect to Mongoose
mongoose.connect('mongodb://localhost/bookstore');
var db = mongoose.connection;

app.get('/', (req, res) => {
  res.send('Please use /api/books or /api/genres');
});

app.get('/api/genres', (req, res) => {
  Genre.getGenres((err, genres) => {
    if(err){
      throw err;
    }
    res.json(genres);
  });
});

app.post('/api/genres', (req, res) => {
  var genre = req.body;
  Genre.addGenre(genre, (err, genre) => {
    if(err){
      throw err;
    }
    res.json(genre);
  });
});

app.put('/api/genres/:_id', (req, res) => {
  var id = req.params._id;
  var genre = req.body;
  Genre.updateGenre(id, genre, {}, (err, genre) => {
    if(err){
      throw err;
    }
    res.json(genre);
  });
});

app.delete('/api/genres/:_id', (req, res) => {
```

```
  var id = req.params._id;
  Genre.removeGenre(id, (err, genre) => {
    if(err){
      throw err;
    }
    res.json(genre);
  });
});

app.get('/api/books', (req, res) => {
  Book.getBooks((err, books) => {
    if(err){
      throw err;
    }
    res.json(books);
  });
});

app.get('/api/books/:_id', (req, res) => {
  Book.getBookById(req.params._id, (err, book) => {
    if(err){
      throw err;
    }
    res.json(book);
  });
});

app.post('/api/books', (req, res) => {
  var book = req.body;
  Book.addBook(book, (err, book) => {
    if(err){
      throw err;
    }
    res.json(book);
  });
});

app.put('/api/books/:_id', (req, res) => {
  var id = req.params._id;
  var book = req.body;
  Book.updateBook(id, book, {}, (err, book) => {
    if(err){
      throw err;
    }
    res.json(book);
  });
});
```

```
app.delete('/api/books/:_id', (req, res) => {
  var id = req.params._id;
  Book.removeBook(id, (err, book) => {
    if(err){
      throw err;
    }
    res.json(book);
  });
});

app.listen(3000);
console.log('Running on port 3000...');
```

PACKAGE.JSON

```
{
  "name": "bookstore",
  "version": "1.0.0",
  "description": "Simple bookstore app",
  "main": "app.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "dependencies":{
    "express":"*",
    "body-parser":"*",
    "mongoose":"*"
  },
  "author": "Brad Traversy",
  "license": "ISC"
}
```

BOOK.JSON

```
const mongoose = require('mongoose');

// Book Schema
const bookSchema = mongoose.Schema({
  title:{
    type: String,
    required: true
  },
  genre:{
    type: String,
    required: true
  },
  description:{
    type: String
  },
  author:{
    type: String,
```

```
    required: true
  },
  publisher:{
    type: String
  },
  pages:{
    type: String
  },
  image_url:{
    type: String
  },
  buy_url:{
    type: String
  },
  create_date:{
    type: Date,
    default: Date.now
  }
});

const Book = module.exports = mongoose.model('Book', bookSchema);

// Get Books
module.exports.getBooks = (callback, limit) => {
  Book.find(callback).limit(limit);
}

// Get Book
module.exports.getBookById = (id, callback) => {
  Book.findById(id, callback);
}

// Add Book
module.exports.addBook = (book, callback) => {
  Book.create(book, callback);
}

// Update Book
module.exports.updateBook = (id, book, options, callback) => {
  var query = {_id: id};
  var update = {
    title: book.title,
    genre: book.genre,
    description: book.description,
    author: book.author,
    publisher: book.publisher,
    pages: book.pages,
    image_url: book.image_url,
```

```
    buy_url: book.buy_url
  }
  Book.findOneAndUpdate(query, update, options, callback);
}

// Delete Book
module.exports.removeBook = (id, callback) => {
  var query = {_id: id};
  Book.remove(query, callback);
}
```

GENRE.JSON

```
const mongoose = require('mongoose');

// Genre Schema
const genreSchema = mongoose.Schema({
  name:{
    type: String,
    required: true
  },
  create_date:{
    type: Date,
    default: Date.now
  }
});

const Genre = module.exports = mongoose.model('Genre', genreSchema);

// Get Genres
module.exports.getGenres = (callback, limit) => {
  Genre.find(callback).limit(limit);
}

// Add Genre
module.exports.addGenre = (genre, callback) => {
  Genre.create(genre, callback);
}

// Update Genre
module.exports.updateGenre = (id, genre, options, callback) => {
  var query = {_id: id};
  var update = {
    name: genre.name
  }
  Genre.findOneAndUpdate(query, update, options, callback);
}


// Delete Genre
```

```
module.exports.removeGenre = (id, callback) => {
  var query = {_id: id};
  Genre.remove(query, callback);
}
```