Penn Engineering

ONLINE LEARNING

Video 4.8

MongoDB: Advanced Queries

Chris Murphy

Review

- **MongoDB** is a NoSQL Database that stores **collections** of **documents**
- Once we've defined a **Schema** we can use the **`find`** function to select all documents in a collection, or pass a query object to select only certain ones
- Once we have a document, we can update it using the **`save`** function

```
{
   title: 'Introduction to Algorithms',
   year: 1990,
   authors: [
      {  name: 'Thomas Cormen', affiliation: 'Dartmouth' },
      {  name: 'Charles Leiserson', affiliation: 'MIT' },
      {  name: 'Ronald Rivest', affiliation: 'MIT' },
      {  name: 'Clifford Stein', affiliation: 'Columbia' }
   ]
}
```

```
{
   title: 'Principles of Compiler Design',
   year: 1977,
   authors: [
      {  name: 'Alfred Aho', affiliation: 'Bell Labs' },
      {  name: 'Jeffrey Ullman', affiliation: 'Princeton' }
   ]
}
```

```
{
   title: 'Introduction to Algorithms',
   year: 1990,
   authors: [
      {  name: 'Thomas Cormen', affiliation: 'Dartmouth' },
      {  name: 'Charles Leiserson', affiliation: 'MIT' },
      {  name: 'Ronald Rivest', affiliation: 'MIT' },
      {  name: 'Clifford Stein', affiliation: 'Columbia' }
   ]
}
```

```
{
   title: 'Principles of Compiler Design',
   year: 1977,
   authors: [
      {  name: 'Alfred Aho', affiliation: 'Bell Labs' },
      {  name: 'Jeffrey Ullman', affiliation: 'Princeton' }
   ]
}
```

```
{
   title: 'Introduction to Algorithms',
   year: 1990,
   authors: [
      {  name: 'Thomas Cormen', affiliation: 'Dartmouth' },
      {  name: 'Charles Leiserson', affiliation: 'MIT' },
      {  name: 'Ronald Rivest', affiliation: 'MIT' },
      {  name: 'Clifford Stein', affiliation: 'Columbia' }
   ]
}
```

```
{
   title: 'Principles of Compiler Design',
   year: 1977,
   authors: [
      {  name: 'Alfred Aho', affiliation: 'Bell Labs' },
      {  name: 'Jeffrey Ullman', affiliation: 'Princeton' }
   ]
}
```

```
{
   title: 'Introduction to Algorithms',
   year: 1990,
   authors: [
      {  name: 'Thomas Cormen', affiliation: 'Dartmouth' },
      {  name: 'Charles Leiserson', affiliation: 'MIT' },
      {  name: 'Ronald Rivest', affiliation: 'MIT' },
      {  name: 'Clifford Stein', affiliation: 'Columbia' }
   ]
}
```

```
{
   title: 'Principles of Compiler Design',
   year: 1977,
   authors: [
      {  name: 'Alfred Aho', affiliation: 'Bell Labs' },
      {  name: 'Jeffrey Ullman', affiliation: 'Princeton' }
   ]
}
```

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

mongoose.connect('mongodb://localhost:27017/myDatabase');

var Schema = mongoose.Schema;

var authorSchema = new Schema({
      name: String,
      affiliation: String
   });

var bookSchema = new Schema({
      title: {type: String, required: true, unique: true},
      year: Number,
      authors: [authorSchema]
   });


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

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

mongoose.connect('mongodb://localhost:27017/myDatabase');

var Schema = mongoose.Schema;

var authorSchema = new Schema({
      name: String,
      affiliation: String
   });

var bookSchema = new Schema({
      title: {type: String, required: true, unique: true},
      year: Number,
      authors: [authorSchema]
   });


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

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

mongoose.connect('mongodb://localhost:27017/myDatabase');

var Schema = mongoose.Schema;

var authorSchema = new Schema({
      name: String,
      affiliation: String
   });

var bookSchema = new Schema({
      title: {type: String, required: true, unique: true},
      year: Number,
      authors: [authorSchema]
   });


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

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

mongoose.connect('mongodb://localhost:27017/myDatabase');

var Schema = mongoose.Schema;

var authorSchema = new Schema({
     name: String,
     affiliation: String
   });

var bookSchema = new Schema({
     title: {type: String, required: true, unique: true},
     year: Number,
     authors: [authorSchema]
   });


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

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

mongoose.connect('mongodb://localhost:27017/myDatabase');

var Schema = mongoose.Schema;

var authorSchema = new Schema({
      name: String,
      affiliation: String
   });

var bookSchema = new Schema({
      title: {type: String, required: true, unique: true},
      year: Number,
      authors: [authorSchema]
   });


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

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

mongoose.connect('mongodb://localhost:27017/myDatabase');

var Schema = mongoose.Schema;

var authorSchema = new Schema({
      name: String,
      affiliation: String
   });

var bookSchema = new Schema({
      title: {type: String, required: true, unique: true},
      year: Number,
      authors: [authorSchema]
   });


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

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

mongoose.connect('mongodb://localhost:27017/myDatabase');

var Schema = mongoose.Schema;

var authorSchema = new Schema({
     name: String,
     affiliation: String
   });

var bookSchema = new Schema({
     title: {type: String, required: true, unique: true},
     year: Number,
     authors: [authorSchema]
   });


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

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

mongoose.connect('mongodb://localhost:27017/myDatabase');

var Schema = mongoose.Schema;

var authorSchema = new Schema({
      name: String,
      affiliation: String
   });

var bookSchema = new Schema({
      title: {type: String, required: true, unique: true},
      year: Number,
      authors: [authorSchema]
   });


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

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

mongoose.connect('mongodb://localhost:27017/myDatabase');

var Schema = mongoose.Schema;

var authorSchema = new Schema({
     name: String,
     affiliation: String
   });

var bookSchema = new Schema({
     title: {type: String, required: true, unique: true},
     year: Number,
     authors: [authorSchema]
   });


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

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

mongoose.connect('mongodb://localhost:27017/myDatabase');

var Schema = mongoose.Schema;

var authorSchema = new Schema({
      name: String,
      affiliation: String
   });

var bookSchema = new Schema({
      title: {type: String, required: true, unique: true},
      year: Number,
      authors: [authorSchema]
   });


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

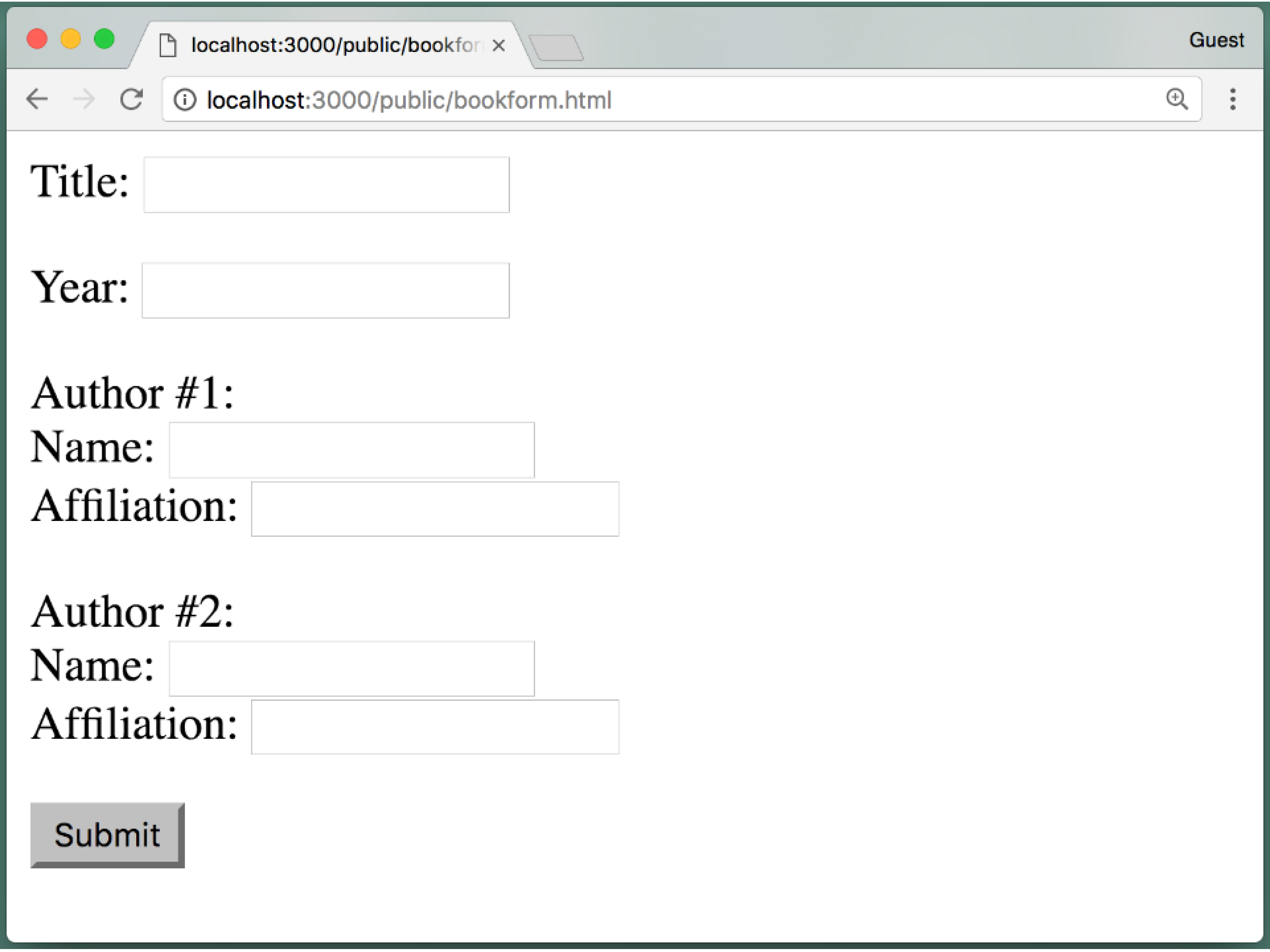
```
var mongoose = require('mongoose');

mongoose.connect('mongodb://localhost:27017/myDatabase');

var Schema = mongoose.Schema;

var authorSchema = new Schema({
     name: String,
     affiliation: String
   });

var bookSchema = new Schema({
     title: {type: String, required: true, unique: true},
     year: Number,
     authors: [authorSchema]
   });


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

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

mongoose.connect('mongodb://localhost:27017/myDatabase');

var Schema = mongoose.Schema;

var authorSchema = new Schema({
     name: String,
     affiliation: String
   });

var bookSchema = new Schema({
     title: {type: String, required: true, unique: true},
     year: Number,
     authors: [authorSchema]
   });


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

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

mongoose.connect('mongodb://localhost:27017/myDatabase');

var Schema = mongoose.Schema;

var authorSchema = new Schema({
     name: String,
     affiliation: String
   });

var bookSchema = new Schema({
     title: {type: String, required: true, unique: true},
     year: Number,
     authors: [authorSchema]
   });


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

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

mongoose.connect('mongodb://localhost:27017/myDatabase');

var Schema = mongoose.Schema;

var authorSchema = new Schema({
      name: String,
      affiliation: String
   });

var bookSchema = new Schema({
      title: {type: String, required: true, unique: true},
      year: Number,
      authors: [authorSchema]
   });


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

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

mongoose.connect('mongodb://localhost:27017/myDatabase');

var Schema = mongoose.Schema;

var authorSchema = new Schema({
     name: String,
     affiliation: String
   });

var bookSchema = new Schema({
     title: {type: String, required: true, unique: true},
     year: Number,
     authors: [authorSchema]
   });


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


localhost:3000/public/bookfor
Guest
localhost:3000/public/bookform.html

Title:	
Year:	

Author #1:

Name:	
Affiliation:	

Author #2:

Name:	
Affiliation:	

```
<form action='/createbook' method='post'>

Title: <input name='title'>
<p>

Year: <input name='year'>
<p>

Author #1: <br>
Name: <input name='authors[0][name]'> <br>
Affiliation: <input name='authors[0][affiliation]'>
<p>

Author #2: <br>
Name: <input name='authors[1][name]'> <br>
Affiliation: <input name='authors[1][affiliation]'>
<p>

<input type='submit' value='Submit'>

</form>
```

```
<form action='/createbook' method='post'>

Title: <input name='title'>
<p>

Year: <input name='year'>
<p>

Author #1: <br>
Name: <input name='authors[0][name]'> <br>
Affiliation: <input name='authors[0][affiliation]'>
<p>

Author #2: <br>
Name: <input name='authors[1][name]'> <br>
Affiliation: <input name='authors[1][affiliation]'>
<p>

<input type='submit' value='Submit'>

</form>
```

```
<form action='/createbook' method='post'>

Title: <input name='title'>
<p>

Year: <input name='year'>
<p>

Author #1: <br>
Name: <input name='authors[0][name]'> <br>
Affiliation: <input name='authors[0][affiliation]'>
<p>

Author #2: <br>
Name: <input name='authors[1][name]'> <br>
Affiliation: <input name='authors[1][affiliation]'>
<p>

<input type='submit' value='Submit'>

</form>
```

```
<form action='/createbook' method='post'>

Title: <input name='title'>
<p>

Year: <input name='year'>
<p>

Author #1: <br>
Name: <input name='authors[0][name]'> <br>
Affiliation: <input name='authors[0][affiliation]'>
<p>

Author #2: <br>
Name: <input name='authors[1][name]'> <br>
Affiliation: <input name='authors[1][affiliation]'>
<p>

<input type='submit' value='Submit'>

</form>
```

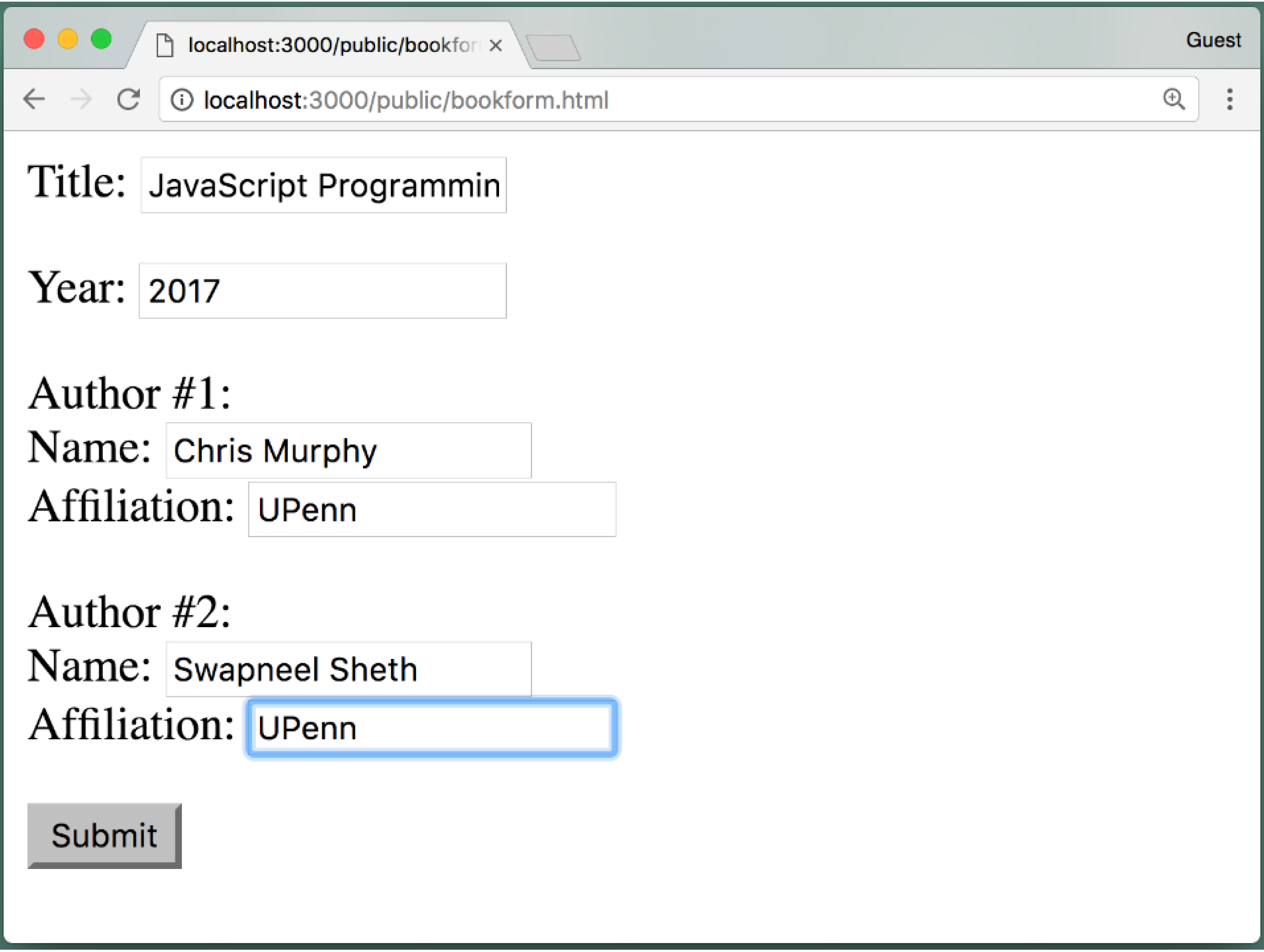
```
<form action='/createbook' method='post'>

Title: <input name='title'>
<p>

Year: <input name='year'>
<p>

Author #1: <br>
Name: <input name='authors[0][name]'> <br>
Affiliation: <input name='authors[0][affiliation]'>
<p>

Author #2: <br>
Name: <input name='authors[1][name]'> <br>
Affiliation: <input name='authors[1][affiliation]'>
<p>

<input type='submit' value='Submit'>

</form>
```

```
<form action='/createbook' method='post'>

Title: <input name='title'>
<p>

Year: <input name='year'>
<p>

Author #1: <br>
Name: <input name='authors[0][name]'> <br>
Affiliation: <input name='authors[0][affiliation]'>
<p>

Author #2: <br>
Name: <input name='authors[1][name]'> <br>
Affiliation: <input name='authors[1][affiliation]'>
<p>

<input type='submit' value='Submit'>

</form>
```

```
<form action='/createbook' method='post'>

Title: <input name='title'>
<p>

Year: <input name='year'>
<p>

Author #1: <br>
Name: <input name='authors[0][name]'> <br>
Affiliation: <input name='authors[0][affiliation]'>
<p>

Author #2: <br>
Name: <input name='authors[1][name]'> <br>
Affiliation: <input name='authors[1][affiliation]'>
<p>

<input type='submit' value='Submit'>

</form>
```

```
<form action='/createbook' method='post'>

Title: <input name='title'>
<p>

Year: <input name='year'>
<p>

Author #1: <br>
Name: <input name='authors[0][name]'> <br>
Affiliation: <input name='authors[0][affiliation]'>
<p>

Author #2: <br>
Name: <input name='authors[1][name]'> <br>
Affiliation: <input name='authors[1][affiliation]'>
<p>

<input type='submit' value='Submit'>

</form>
```

```
<form action='/createbook' method='post'>

Title: <input name='title'>
<p>

Year: <input name='year'>
<p>

Author #1: <br>
Name: <input name='authors[0][name]'> <br>
Affiliation: <input name='authors[0][affiliation]'>
<p>

Author #2: <br>
Name: <input name='authors[1][name]'> <br>
Affiliation: <input name='authors[1][affiliation]'>
<p>

<input type='submit' value='Submit'>

</form>
```


localhost:3000/public/bookfor
Guest
localhost:3000/public/bookform.html
Submit

Title: JavaScript Programmin

Year: 2017

Author #1:
Name: Chris Murphy
Affiliation: UPenn

Author #2:
Name: Swapneel Sheth
Affiliation: UPenn

```
var express = require('express');
var app = express();
app.set('view engine', 'ejs');
var bodyParser = require('body-parser');
app.use(bodyParser.urlencoded({ extended: true }));

var Book = require('./Book.js');

app.use('/createbook', (req, res) => {
   console.log(req.body);
   var newBook = new Book(req.body);
   newBook.save( (err) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('created', { book: newBook });
      }
   } );
});
```

```
var express = require('express');
var app = express();
app.set('view engine', 'ejs');
var bodyParser = require('body-parser');
app.use(bodyParser.urlencoded({ extended: true }));

var Book = require('./Book.js');

app.use('/createbook', (req, res) => {
   console.log(req.body);
   var newBook = new Book(req.body);
   newBook.save( (err) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('created', { book: newBook });
      }
   } );
});
```

```
var express = require('express');
var app = express();
app.set('view engine', 'ejs');
var bodyParser = require('body-parser');
app.use(bodyParser.urlencoded({ extended: true }));

var Book = require('./Book.js');

app.use('/createbook', (req, res) => {
   console.log(req.body);
   var newBook = new Book(req.body);
   newBook.save( (err) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('created', { book: newBook });
      }
   } );
});
```

```
var express = require('express');
var app = express();
app.set('view engine', 'ejs');
var bodyParser = require('body-parser');
app.use(bodyParser.urlencoded({ extended: true }));

var Book = require('./Book.js');

app.use('/createbook', (req, res) => {
  console.log(req.body);
  var newBook = new Book(req.body);
  newBook.save( (err) => {
    if (err) {
      res.type('html').status(500);
      res.send('Error: ' + err);
    }
    else {
      res.render('created', { book: newBook });
    }
  } );
});
```

```
var express = require('express');
var app = express();
app.set('view engine', 'ejs');
var bodyParser = require('body-parser');
app.use(bodyParser.urlencoded({ extended: true }));

var Book = require('./Book.js');

app.use('/createbook', (req, res) => {
    console.log(req.body);
    var newBook = new Book(req.body);
    newBook.save( (err) => {
        if (err) {
            res.type('html').status(500);
            res.send('Error: ' + err);
        }
        else {
            res.render('created', { book: newBook });
        }
    } );
});
```

```
var express = require('express');
var app = express();
app.set('view engine', 'ejs');
var bodyParser = require('body-parser');
app.use(bodyParser.urlencoded({ extended: true }));

var Book = require('./Book.js');

app.use('/createbook', (req, res) => {
    console.log(req.body);
    var newBook = new Book(req.body);
    newBook.save( (err) => {
        if (err) {
            res.type('html').status(500);
            res.send('Error: ' + err);
        }
        else {
            res.render('created', { book: newBook });
        }
    } );
});
```

```
var express = require('express');
var app = express();
app.set('view engine', 'ejs');
var bodyParser = require('body-parser');
app.use(bodyParser.urlencoded({ extended: true }));

var Book = require('./Book.js');

app.use('/createbook', (req, res) => {
   console.log(req.body);
   var newBook = new Book(req.body);
   newBook.save( (err) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('created', { book: newBook });
      }
   } );
});
```

```
{ title: 'JavaScript Programming',
  year: '2017',
  authors:
   [ { name: 'Chris Murphy', affiliation: 'UPenn' },
     { name: 'Swapneel Sheth', affiliation: 'UPenn' } ] }
{ title: 'JavaScript Programming',
  year: '2017',
  authors:
   [ { name: 'Chris Murphy', affiliation: 'UPenn' },
     { name: 'Swapneel Sheth', affiliation: 'UPenn' } ] }
```

```
var authorSchema = new Schema({
      name: String,
      affiliation: String
   });

var bookSchema = new Schema({
      title: {type: String, required: true, unique: true},
      year: Number,
      authors: [authorSchema]
   });
```

```
{ title: 'JavaScript Programming',
  year: '2017',
  authors:
   [ { name: 'Chris Murphy', affiliation: 'UPenn' },
     { name: 'Swapneel Sheth', affiliation: 'UPenn' } ] }
```

```
var authorSchema = new Schema({
      name: String,
      affiliation: String
   });

var bookSchema = new Schema({
      title: {type: String, required: true, unique: true},
      year: Number,
      authors: [authorSchema]
   });
```

```
{ title: 'JavaScript Programming',
  year: '2017',
  authors:
   [ { name: 'Chris Murphy', affiliation: 'UPenn' },
     { name: 'Swapneel Sheth', affiliation: 'UPenn' } ] }
```

```
var authorSchema = new Schema({
     name: String,
     affiliation: String
   });

var bookSchema = new Schema({
     title: {type: String, required: true, unique: true},
     year: Number,
     authors: [authorSchema]
   });
```

```
{ title: 'JavaScript Programming',
  year: '2017',
  authors:
   [ { name: 'Chris Murphy', affiliation: 'UPenn' },
     { name: 'Swapneel Sheth', affiliation: 'UPenn' } ] }
```

```
var authorSchema = new Schema({
      name: String,
      affiliation: String
   });

var bookSchema = new Schema({
      title: {type: String, required: true, unique: true},
      year: Number,
      authors: [authorSchema]
   });
```

```
{ title: 'JavaScript Programming',
  year: '2017',
  authors:
   [ { name: 'Chris Murphy', affiliation: 'UPenn' },
     { name: 'Swapneel Sheth', affiliation: 'UPenn' } ] }
```

```
var authorSchema = new Schema({
     name: String,
     affiliation: String
   });

var bookSchema = new Schema({
     title: {type: String, required: true, unique: true},
     year: Number,
     authors: [authorSchema]
   });
```

```
{ title: 'JavaScript Programming',
  year: '2017',
  authors:
   [ { name: 'Chris Murphy', affiliation: 'UPenn' },
     { name: 'Swapneel Sheth', affiliation: 'UPenn' } ] }
```

```
var authorSchema = new Schema({
      name: String,
      affiliation: String
   });

var bookSchema = new Schema({
      title: {type: String, required: true, unique: true},
      year: Number,
      authors: [authorSchema]
   });
```

```
{ title: 'JavaScript Programming',
  year: '2017',
  authors:
   [ { name: 'Chris Murphy', affiliation: 'UPenn' },
     { name: 'Swapneel Sheth', affiliation: 'UPenn' } ] }
```

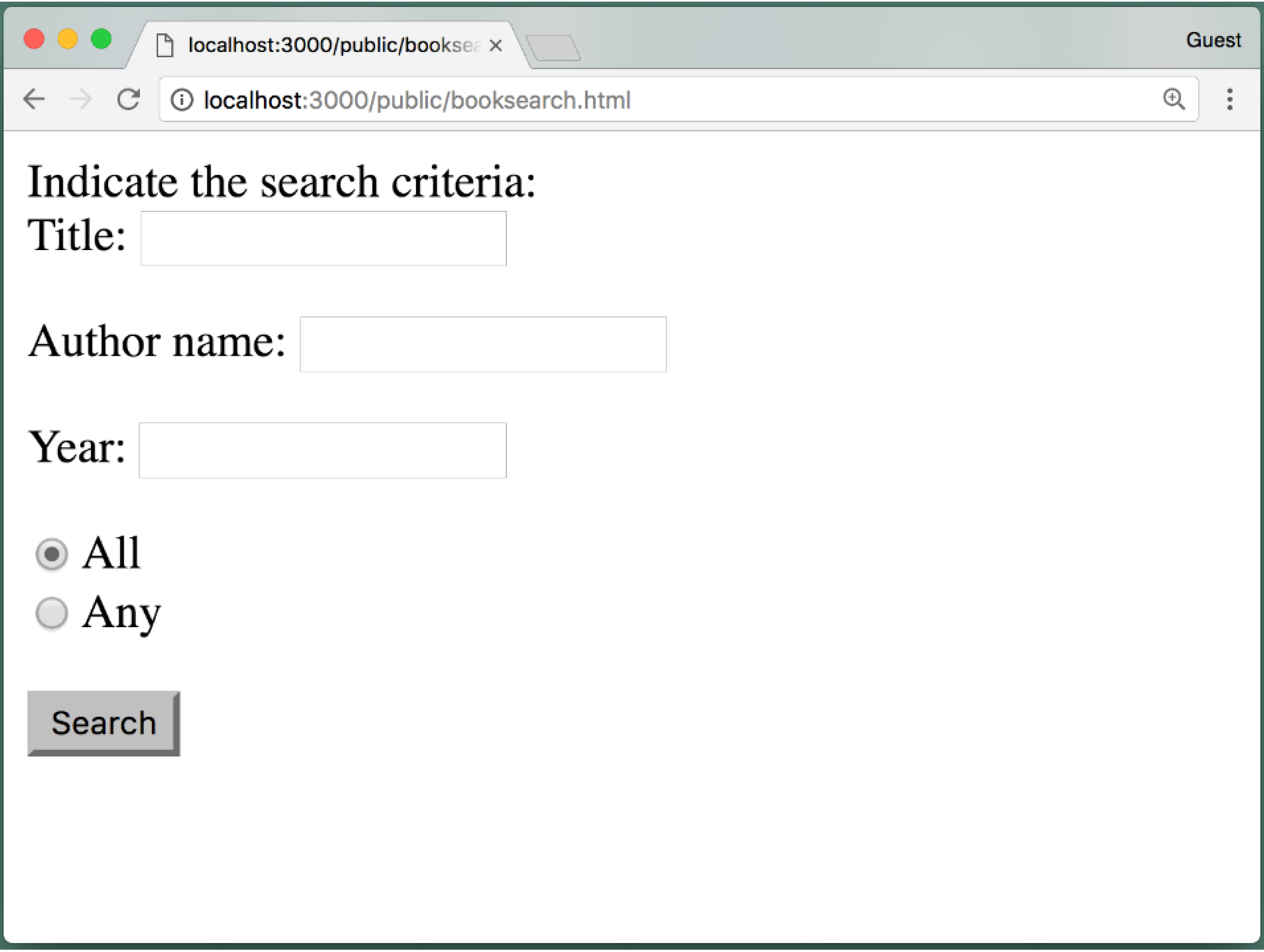
```
var authorSchema = new Schema({
      name: String,
      affiliation: String
   });

var bookSchema = new Schema({
      title: {type: String, required: true, unique: true},
      year: Number,
      authors: [authorSchema]
   });
```

```
var express = require('express');
var app = express();
app.set('view engine', 'ejs');
var bodyParser = require('body-parser');
app.use(bodyParser.urlencoded({ extended: true }));

var Book = require('./Book.js');

app.use('/createbook', (req, res) => {
   console.log(req.body);
   var newBook = new Book(req.body);
   newBook.save( (err) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('created', { book: newBook });
      }
   } );
});
```

```
var express = require('express');
var app = express();
app.set('view engine', 'ejs');
var bodyParser = require('body-parser');
app.use(bodyParser.urlencoded({ extended: true }));

var Book = require('./Book.js');

app.use('/createbook', (req, res) => {
   console.log(req.body);
   var newBook = new Book(req.body);
   newBook.save( (err) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('created', { book: newBook });
      }
   } );
});
```

```
var express = require('express');
var app = express();
app.set('view engine', 'ejs');
var bodyParser = require('body-parser');
app.use(bodyParser.urlencoded({ extended: true }));

var Book = require('./Book.js');

app.use('/createbook', (req, res) => {
    console.log(req.body);
    var newBook = new Book(req.body);
    newBook.save( (err) => {
        if (err) {
            res.type('html').status(500);
            res.send('Error: ' + err);
        }
        else {
            res.render('created', { book: newBook });
        }
    } );
});
```

```
var express = require('express');
var app = express();
app.set('view engine', 'ejs');
var bodyParser = require('body-parser');
app.use(bodyParser.urlencoded({ extended: true }));

var Book = require('./Book.js');

app.use('/createbook', (req, res) => {
    console.log(req.body);
    var newBook = new Book(req.body);
    newBook.save( (err) => {
        if (err) {
            res.type('html').status(500);
            res.send('Error: ' + err);
        }
        else {
            res.render('created', { book: newBook });
        }
    } );
});
```


localhost:3000/public/bookse
Guest
localhost:3000/public/booksearch.html

Indicate the search criteria:
Title:

Author name:

Year:

[✓ All]
Any

```
Indicate the search criteria:

<form action='/search' method='post'>

Title: <input name='title'>
<p>

Author name: <input name='name'>
<p>

Year: <input name='year'>
<p>

<input type='radio' name='which' value='all' checked>All
<br>
<input type='radio' name='which' value='any'>Any
<p>

<input type='submit' value='Search'>

</form>
```

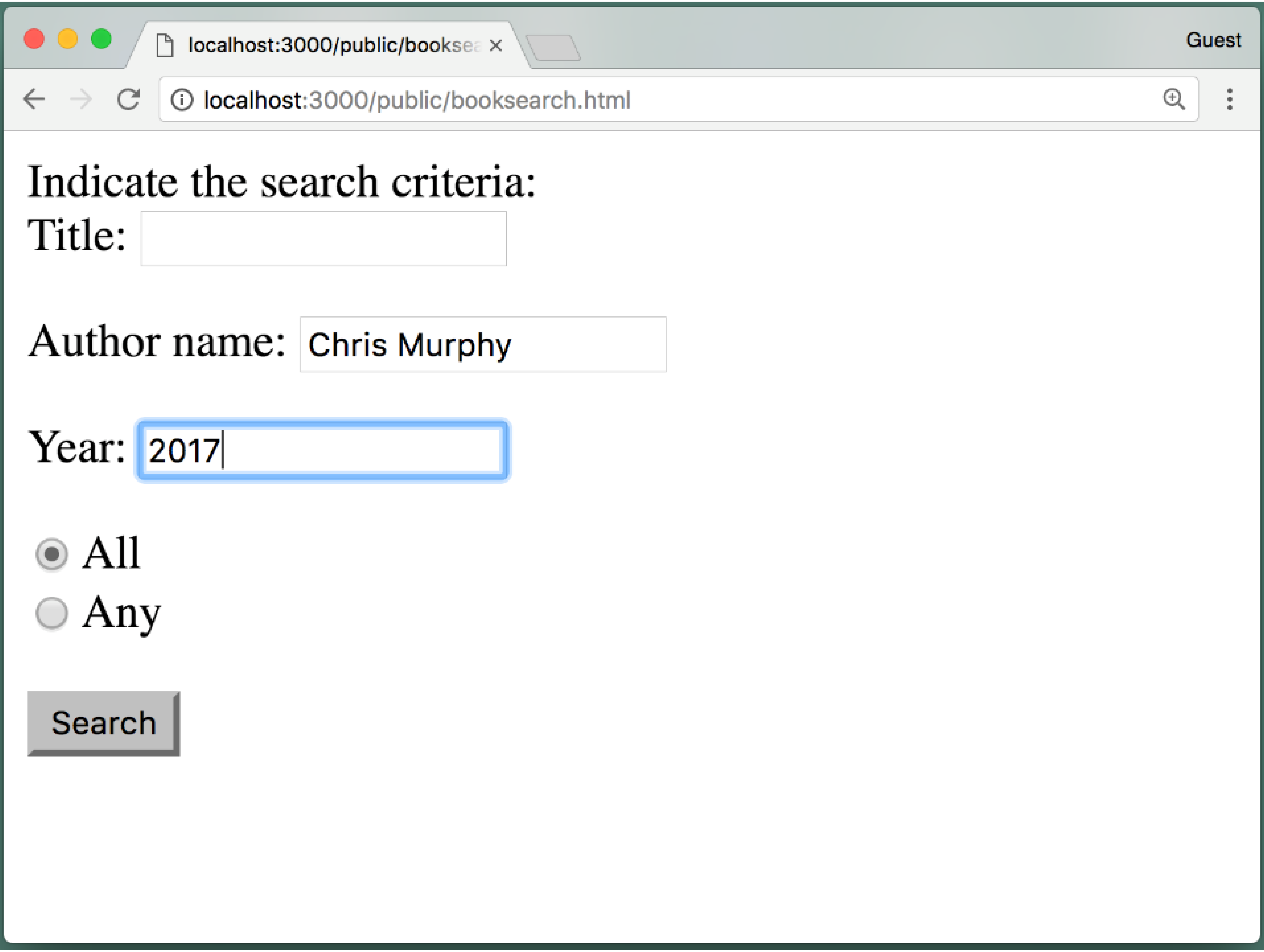
```
Indicate the search criteria:

<form action='/search' method='post'>

Title: <input name='title'>
<p>

Author name: <input name='name'>
<p>

Year: <input name='year'>
<p>

<input type='radio' name='which' value='all' checked>All
<br>
<input type='radio' name='which' value='any'>Any
<p>

<input type='submit' value='Search'>

</form>
```

```
Indicate the search criteria:

<form action='/search' method='post'>

Title: <input name='title'>
<p>

Author name: <input name='name'>
<p>

Year: <input name='year'>
<p>

<input type='radio' name='which' value='all' checked>All
<br>
<input type='radio' name='which' value='any'>Any
<p>

<input type='submit' value='Search'>

</form>
```

```
Indicate the search criteria:

<form action='/search' method='post'>

Title: <input name='title'>
<p>

Author name: <input name='name'>
<p>

Year: <input name='year'>
<p>

<input type='radio' name='which' value='all' checked>All
<br>
<input type='radio' name='which' value='any'>Any
<p>

<input type='submit' value='Search'>

</form>
```

```
Indicate the search criteria:

<form action='/search' method='post'>

Title: <input name='title'>
<p>

Author name: <input name='name'>
<p>

Year: <input name='year'>
<p>

<input type='radio' name='which' value='all' checked>All
<br>
<input type='radio' name='which' value='any'>Any
<p>

<input type='submit' value='Search'>

</form>
```

```
Indicate the search criteria:

<form action='/search' method='post'>

Title: <input name='title'>
<p>

Author name: <input name='name'>
<p>

Year: <input name='year'>
<p>

<input type='radio' name='which' value='all' checked>All
<br>
<input type='radio' name='which' value='any'>Any
<p>

<input type='submit' value='Search'>

</form>
```



localhost:3000/public/booksearch.html

Indicate the search criteria:

Title:

Author name: Chris Murphy

Year: 2017

[✓ All]

Any

```
app.use('/search', (req, res) => {

     if (req.body.which == 'all') {
         searchAll(req, res);
     }
     else if (req.body.which == 'any') {
         searchAny(req, res);
     }
     else {
         searchAll(req, res);
     }

  });
```

```
app.use('/search', (req, res) => {

        if (req.body.which == 'all') {
              searchAll(req, res);
        }
        else if (req.body.which == 'any') {
              searchAny(req, res);
        }
        else {
              searchAll(req, res);
        }

  });
```

```
app.use('/search', (req, res) => {

        if (req.body.which == 'all') {
              searchAll(req, res);
        }
        else if (req.body.which == 'any') {
              searchAny(req, res);
        }
        else {
              searchAll(req, res);
        }

  });
```

```
function searchAll(req, res) {

   var query = {};

   if (req.body.title) query.title = req.body.title;
   if (req.body.year) query.year = req.body.year;
   if (req.body.name) {
      query['authors.name'] = req.body.name;
    }

   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAll(req, res) {

   var query = {};

   if (req.body.title) query.title = req.body.title;
   if (req.body.year) query.year = req.body.year;
   if (req.body.name) {
      query['authors.name'] = req.body.name;
    }

   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAll(req, res) {

   var query = {};

   if (req.body.title) query.title = req.body.title;
   if (req.body.year) query.year = req.body.year;
   if (req.body.name) {
      query['authors.name'] = req.body.name;
    }

   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAll(req, res) {

   var query = {};

   if (req.body.title) query.title = req.body.title;
   if (req.body.year) query.year = req.body.year;
   if (req.body.name) {
      query['authors.name'] = req.body.name;
    }

   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAll(req, res) {

   var query = {};

   if (req.body.title) query.title = req.body.title;
   if (req.body.year) query.year = req.body.year;
   if (req.body.name) {
      query['authors.name'] = req.body.name;
   }

   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAll(req, res) {

    var query = {};

    if (req.body.title) query.title = req.body.title;
    if (req.body.year) query.year = req.body.year;
    if (req.body.name) {
        query['authors.name'] = req.body.name;
     }

    console.log(query);


    Book.find( query, (err, books) => {
        if (err) {
            res.type('html').status(500);
            res.send('Error: ' + err);
        }
        else {
            res.render('books', { books: books });
        }
    });
}
```

```
function searchAll(req, res) {

   var query = {};

   if (req.body.title) query.title = req.body.title;
   if (req.body.year) query.year = req.body.year;
   if (req.body.name) {
      query['authors.name'] = req.body.name;
   }

   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAll(req, res) {

   var query = {};

   if (req.body.title) query.title = req.body.title;
   if (req.body.year) query.year = req.body.year;
   if (req.body.name) {
      query['authors.name'] = req.body.name;
   }

   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAll(req, res) {

   var query = {};

   if (req.body.title) query.title = req.body.title;
   if (req.body.year) query.year = req.body.year;
   if (req.body.name) {
      query['authors.name'] = req.body.name;
   }

   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

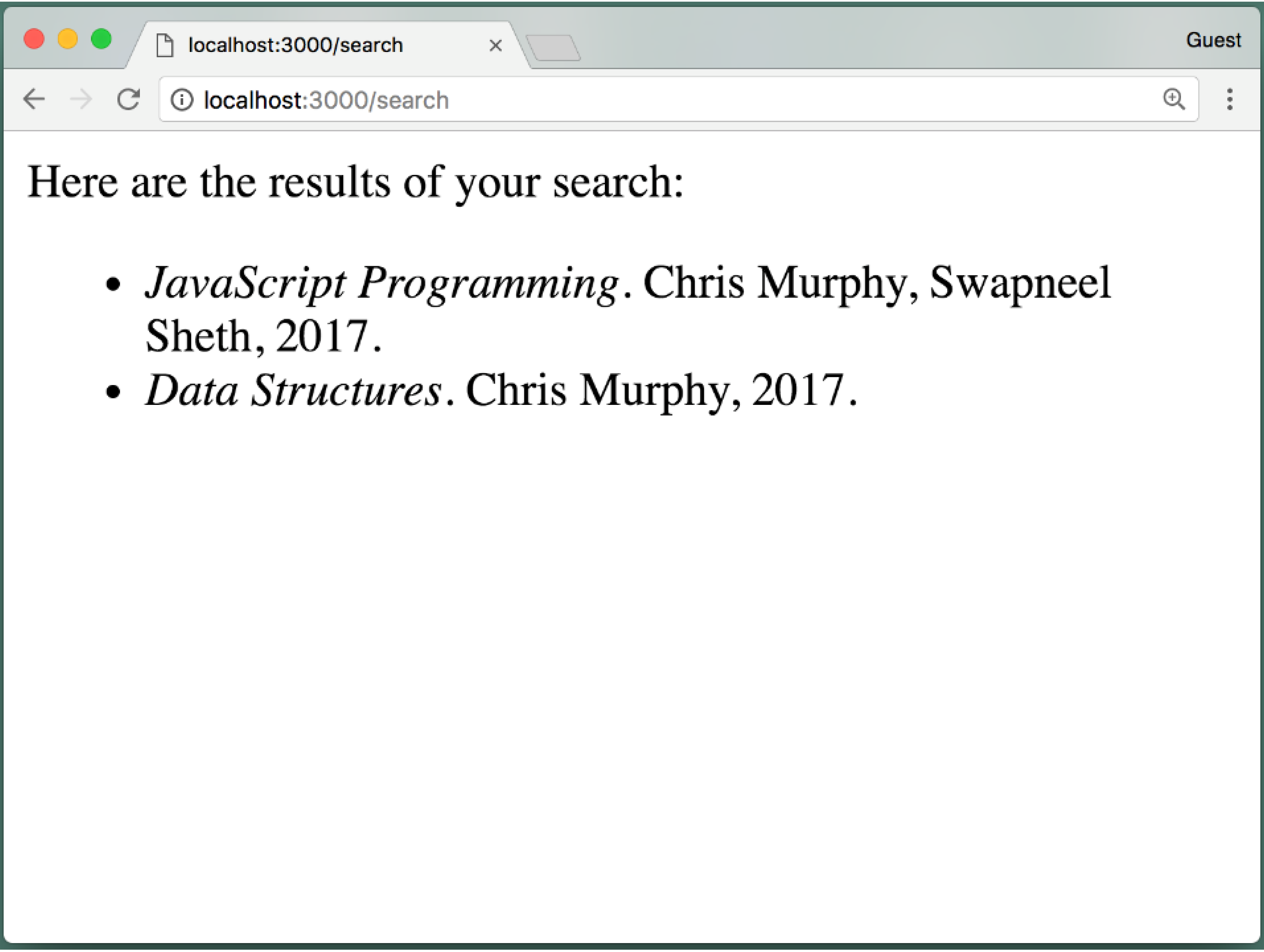
```
function searchAll(req, res) {

   var query = {};

   if (req.body.title) query.title = req.body.title;
   if (req.body.year) query.year = req.body.year;
   if (req.body.name) {
      query['authors.name'] = req.body.name;
    }

   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAll(req, res) {

   var query = {};

   if (req.body.title) query.title = req.body.title;
   if (req.body.year) query.year = req.body.year;
   if (req.body.name) {
      query['authors.name'] = req.body.name;
    }

   console.log(query);
   // { 'authors.name': 'Chris Murphy', year: '2017' }

   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAll(req, res) {

   var query = {};

   if (req.body.title) query.title = req.body.title;
   if (req.body.year) query.year = req.body.year;
   if (req.body.name) {
      query['authors.name'] = req.body.name;
   }

   console.log(query);
   // { 'authors.name': 'Chris Murphy', year: '2017' }

   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAll(req, res) {

  var query = {};

  if (req.body.title) query.title = req.body.title;
  if (req.body.year) query.year = req.body.year;
  if (req.body.name) {
    query['authors.name'] = req.body.name;
  }

  console.log(query);
  // { 'authors.name': 'Chris Murphy', year: '2017' }

  Book.find( query, (err, books) => {
    if (err) {
      res.type('html').status(500);
      res.send('Error: ' + err);
    }
    else {
      res.render('books', { books: books });
    }
  });
}
```

```
function searchAll(req, res) {

   var query = {};

   if (req.body.title) query.title = req.body.title;
   if (req.body.year) query.year = req.body.year;
   if (req.body.name) {
      query['authors.name'] = req.body.name;
    }

   console.log(query);
   // { 'authors.name': 'Chris Murphy', year: '2017' }

   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```



Here are the results of your search:

- *JavaScript Programming*. Chris Murphy, Swapneel Sheth, 2017.
- *Data Structures*. Chris Murphy, 2017.

```
<!-- This is views/books.ejs -->

Here are the results of your search:
<ul>

<% books.forEach( (book) => { %>

  <li>

   <i><%= book.title %></i>.

   <% book.authors.forEach( (author) => { %>

        <%= author.name %>,

   <% }); %>

   <%= book.year %>.

  </li>

<%  }); %>

</ul>
```

```
<!-- This is views/books.ejs -->

Here are the results of your search:
<ul>

<% books.forEach( (book) => { %>

  <li>

   <i><%= book.title %></i>.

   <% book.authors.forEach( (author) => { %>

        <%= author.name %>,

   <% }); %>

   <%= book.year %>.

  </li>

<%  }); %>

</ul>
```

```
<!-- This is views/books.ejs -->

Here are the results of your search:
<ul>

<% books.forEach( (book) => { %>

  <li>

   <i><%= book.title %></i>.

   <% book.authors.forEach( (author) => { %>

        <%= author.name %>,

   <% }); %>

   <%= book.year %>.

  </li>

<%  }); %>

</ul>
```

```
<!-- This is views/books.ejs -->

Here are the results of your search:
<ul>

<% books.forEach( (book) => { %>

  <li>

   <i><%= book.title %></i>.

   <% book.authors.forEach( (author) => { %>

        <%= author.name %>,

   <% }); %>

   <%= book.year %>.

  </li>

<%  }); %>

</ul>
```

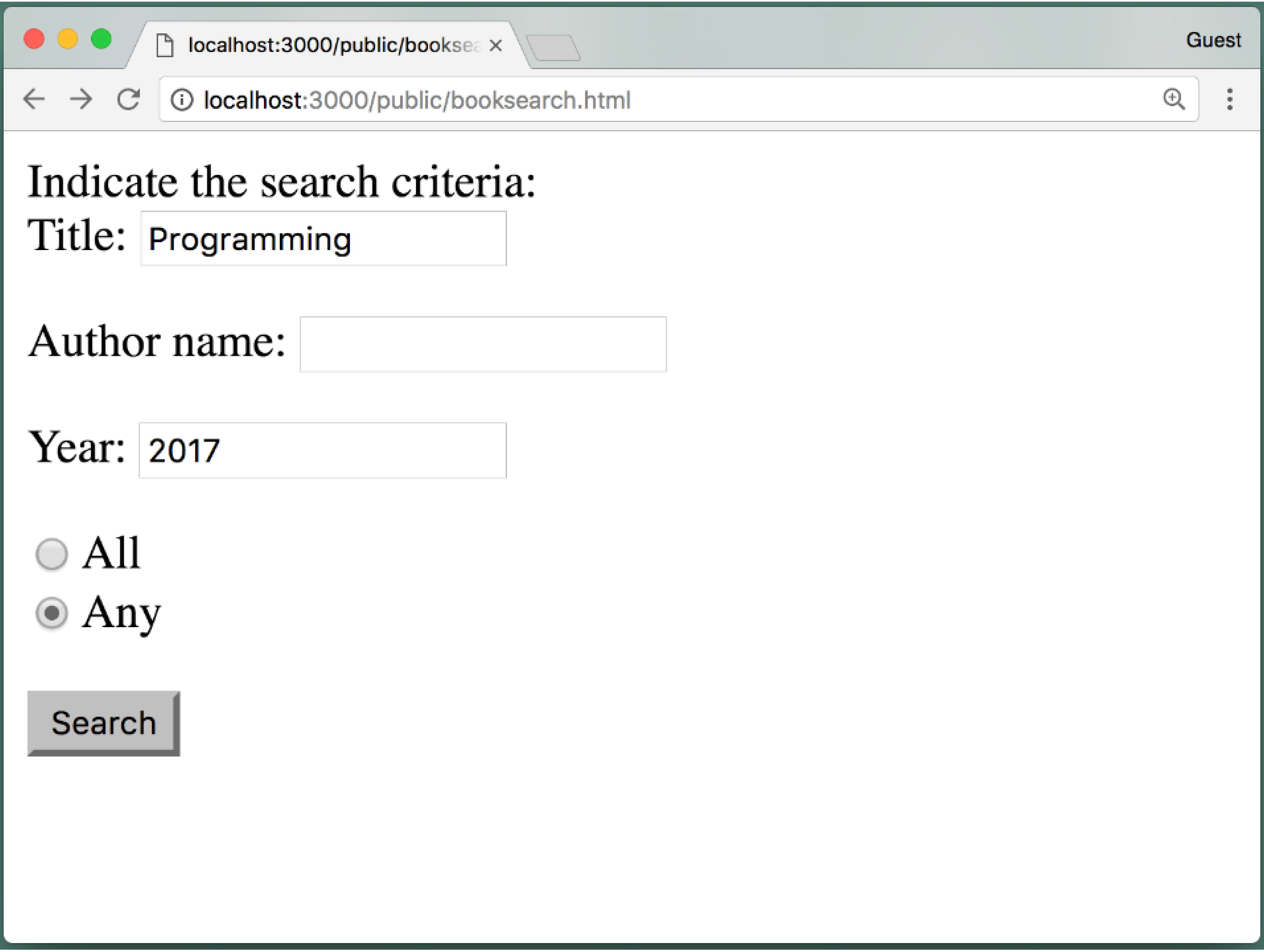
```
<!-- This is views/books.ejs -->

Here are the results of your search:
<ul>

<% books.forEach( (book) => { %>

  <li>

   <i><%= book.title %></i>.

   <% book.authors.forEach( (author) => { %>

        <%= author.name %>,

   <% }); %>

   <%= book.year %>.

  </li>

<%  }); %>

</ul>
```

```
<!-- This is views/books.ejs -->

Here are the results of your search:
<ul>

<% books.forEach( (book) => { %>

  <li>

   <i><%= book.title %></i>.

   <% book.authors.forEach( (author) => { %>

         <%= author.name %>,

   <% }); %>

   <%= book.year %>.

  </li>

<%  }); %>

</ul>
```

```
<!-- This is views/books.ejs -->

Here are the results of your search:
<ul>

<% books.forEach( (book) => { %>

  <li>

   <i><%= book.title %></i>.

   <% book.authors.forEach( (author) => { %>

        <%= author.name %>,

   <% }); %>

   <%= book.year %>.

  </li>

<%  }); %>

</ul>
```

```
<!-- This is views/books.ejs -->

Here are the results of your search:
<ul>

<% books.forEach( (book) => { %>

  <li>

   <i><%= book.title %></i>.

   <% book.authors.forEach( (author) => { %>

         <%= author.name %>,

   <% }); %>

   <%= book.year %>.

  </li>

<%  }); %>

</ul>
```

```
<!-- This is views/books.ejs -->

Here are the results of your search:
<ul>

<% books.forEach( (book) => { %>

  <li>

   <i><%= book.title %></i>.

   <% book.authors.forEach( (author) => { %>

        <%= author.name %>,

   <% }); %>

   <%= book.year %>.

  </li>

<%  }); %>

</ul>
```


localhost:3000/public/booksearch.html

Indicate the search criteria:

Title: Programming

Author name:

Year: 2017

All [✓ Any]

```
app.use('/search', (req, res) => {

      if (req.body.which == 'all') {
           searchAll(req, res);
      }
      else if (req.body.which == 'any') {
           searchAny(req, res);
      }
      else {
           searchAll(req, res);
      }

  });
```

```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: req.body.title });
   if (req.body.year)
      terms.push( { year: req.body.year });
   if (req.body.name)
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: req.body.title });
   if (req.body.year)
      terms.push( { year: req.body.year });
   if (req.body.name)
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: req.body.title });
   if (req.body.year)
      terms.push( { year: req.body.year });
   if (req.body.name)
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: req.body.title });
   if (req.body.year)
      terms.push( { year: req.body.year });
   if (req.body.name)
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: req.body.title });
   if (req.body.year)
      terms.push( { year: req.body.year });
   if (req.body.name)
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: req.body.title });
   if (req.body.year)
      terms.push( { year: req.body.year });
   if (req.body.name)
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: req.body.title });
   if (req.body.year)
      terms.push( { year: req.body.year });
   if (req.body.name)
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: req.body.title });
   if (req.body.year)
      terms.push( { year: req.body.year });
   if (req.body.name)
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: req.body.title });
   if (req.body.year)
      terms.push( { year: req.body.year });
   if (req.body.name)
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: req.body.title });
   if (req.body.year)
      terms.push( { year: req.body.year });
   if (req.body.name)
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: req.body.title });
   if (req.body.year)
      terms.push( { year: req.body.year });
   if (req.body.name)
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

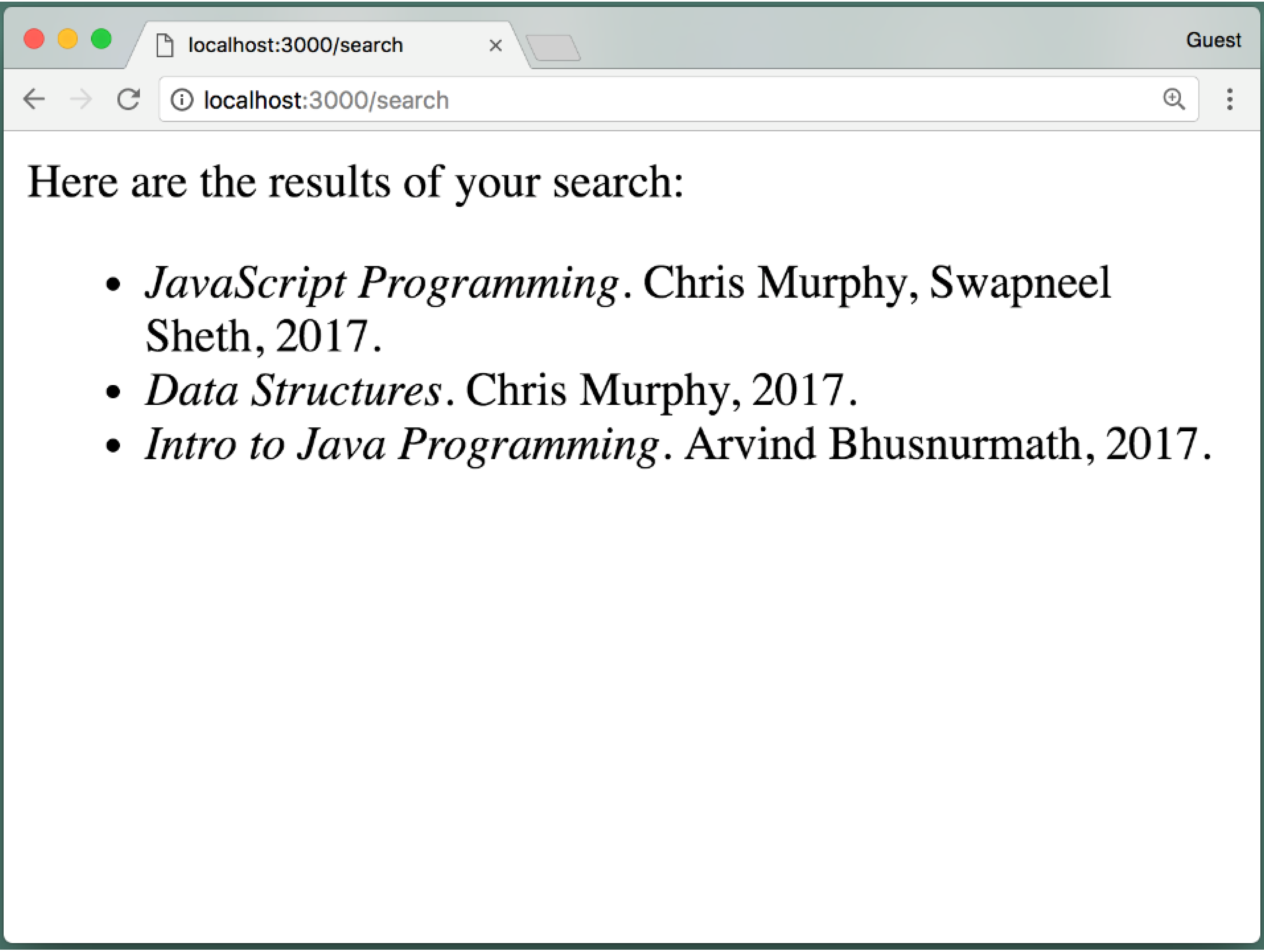
```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: req.body.title });
   if (req.body.year)
      terms.push( { year: req.body.year });
   if (req.body.name)
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: req.body.title });
   if (req.body.year)
      terms.push( { year: req.body.year });
   if (req.body.name)
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);


   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: req.body.title });
   if (req.body.year)
      terms.push( { year: req.body.year });
   if (req.body.name)
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);



   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: req.body.title });
   if (req.body.year)
      terms.push( { year: req.body.year });
   if (req.body.name)
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);
   // { '$or': [ {title: 'Programming'}, {year: '2017'} ] }

   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
}
```

```
function searchAny(req, res) {

    var terms = [];
    if (req.body.title)
        terms.push( { title: req.body.title });
    if (req.body.year)
        terms.push( { year: req.body.year });
    if (req.body.name)
        terms.push( { 'authors.name' : req.body.name });

    var query = { $or : terms };
    console.log(query);


    Book.find( query, (err, books) => {
        if (err) {
            res.type('html').status(500);
            res.send('Error: ' + err);
        }
        else {
            res.render('books', { books: books });
        }
    });
}
```



Here are the results of your search:

- *JavaScript Programming*. Chris Murphy, Swapneel Sheth, 2017.
- *Data Structures*. Chris Murphy, 2017.
- *Intro to Java Programming*. Arvind Bhusnurmath, 2017.

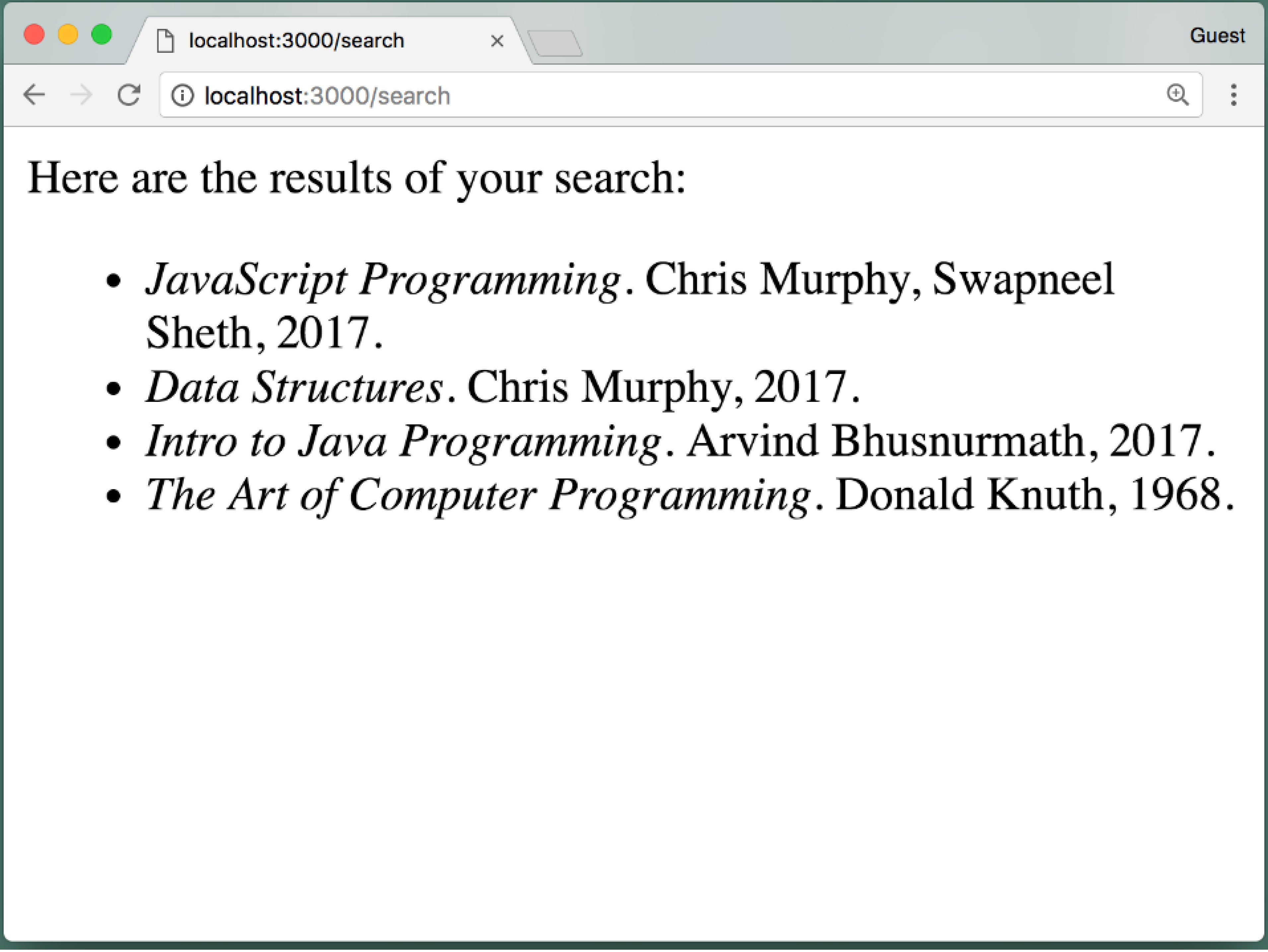
```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: req.body.title });
   if (req.body.year) {
      terms.push( { year: req.body.year });
   if (req.body.name) {
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);
   // { '$or': [ {title: 'Programming'}, {year: '2017'} ] }

   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
```

```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: req.body.title });
   if (req.body.year) {
      terms.push( { year: req.body.year });
   if (req.body.name) {
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);
   // { '$or': [ {title: 'Programming'}, {year: '2017'} ] }

   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
```

```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: req.body.title });
   if (req.body.year) {
      terms.push( { year: req.body.year });
   if (req.body.name) {
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);
   // { '$or': [ {title: 'Programming'}, {year: '2017'} ] }

   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
```

```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: { $regex: req.body.title } });
   if (req.body.year) {
      terms.push( { year: req.body.year });
   if (req.body.name) {
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);
   // { '$or': [ {title: 'Programming'}, {year: '2017'} ] }

   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
```

```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: { $regex: req.body.title } });
   if (req.body.year) {
      terms.push( { year: req.body.year });
   if (req.body.name) {
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);
   // { '$or': [ {title: [Object]}, {year: '2017'} ] }

   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
```

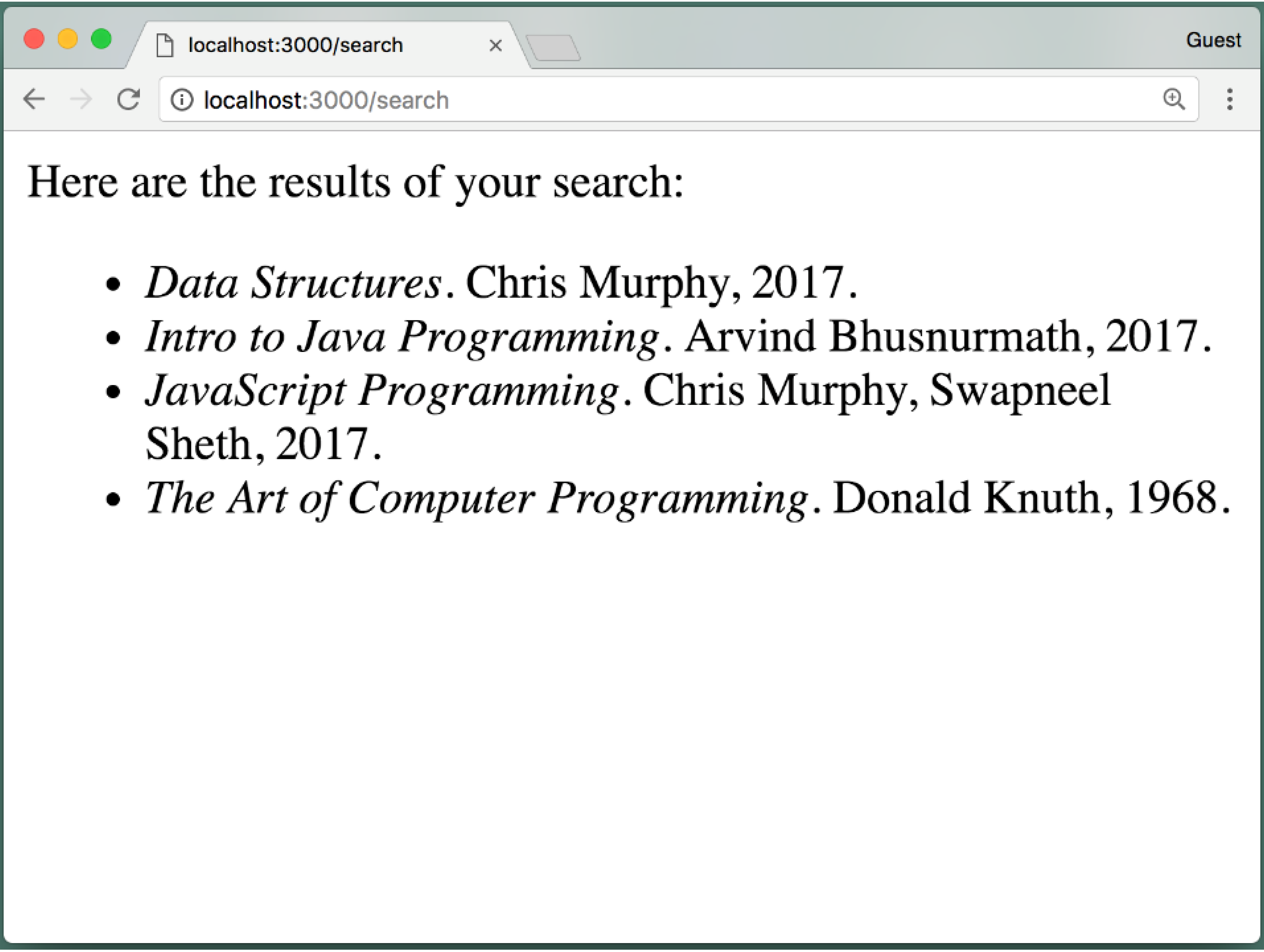


Here are the results of your search:

- *JavaScript Programming*. Chris Murphy, Swapneel Sheth, 2017.
- *Data Structures*. Chris Murphy, 2017.
- *Intro to Java Programming*. Arvind Bhusnurmath, 2017.
- *The Art of Computer Programming*. Donald Knuth, 1968.

```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: { $regex: req.body.title } });
   if (req.body.year) {
      terms.push( { year: req.body.year });
   if (req.body.name) {
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);
   // { '$or': [ {title: [Object]}, {year: '2017'} ] }

   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   });
```

```
function searchAny(req, res) {

   var terms = [];
   if (req.body.title)
      terms.push( { title: { $regex: req.body.title } });
   if (req.body.year) {
      terms.push( { year: req.body.year });
   if (req.body.name) {
      terms.push( { 'authors.name' : req.body.name });

   var query = { $or : terms };
   console.log(query);
   // { '$or': [ {title: [Object]}, {year: '2017'} ] }

   Book.find( query, (err, books) => {
      if (err) {
         res.type('html').status(500);
         res.send('Error: ' + err);
      }
      else {
         res.render('books', { books: books });
      }
   }).sort( { 'title' : 'asc' } );
```


localhost:3000/search

Guest

localhost:3000/search

Here are the results of your search:

- *Data Structures*. Chris Murphy, 2017.
- *Intro to Java Programming*. Arvind Bhusnurmath, 2017.
- *JavaScript Programming*. Chris Murphy, Swapneel Sheth, 2017.
- *The Art of Computer Programming*. Donald Knuth, 1968.

Summary

- MongoDB allows us to have a Schema in which one document contains other documents
- We can then do queries for documents using the properties of the documents they contain
- We can also do “all” and “any” queries by passing objects to the `find` function
- And sort the results using `sort`