INTRO TO  mongoDB

Fullstack Academy of Code

What is  mongoDB?

- Not (Only) relational Database (NoSQL)
  - Doesn't store data in Tables, but in collections as **JSON documents**
    - Actually BSON but you can think of it as JSON
    - {"name":"Nimit Maru", "location":"New York, NY"}
- Schema-less
- You can nest data, references…
  - { "foo": ["bar","baz"], "anObj": {} }

SQL to Mongo NoSQL

SQL Term	Mongo NoSQL Term
Database	*Database*
Table	*Collection*
Row	*Document*
Column	*Field*
Index	*Index*
Join	*Embedding & Linking*
Primary Key	*_id field*
Group By	*Aggregation*

Talking to Mongo

- Talk to Mongo using JavaScript functions
- The "db" object is the pathway to the DB
- Examples:

SQL	Mongo Statement
CREATE TABLE users (name varchar, age int)	*Done automatically while inserting*
INSERT INTO users (name, age) VALUES ("Nicole", 24)	*db.users.insert({"name":"Nicole", "age":24});*
*SELECT * FROM users;*	*db.users.find()*
*SELECT * FROM users WHERE age=24;*	*db.users.find({"age":24})*
*SELECT * FROM users WHERE name="Jay" ORDER BY name ASC;*	*db.users.find({"name":"Jay"}).sort("name":1)*

CRUD in Mongo

- Create

```
db.users.insert({
  firstName:   "Nimit",
  lastName:    "Maru",
  username:    "nimitmaru",
  password:    "74824h234b280412312fdgsdgae"
});
```

CRUD in Mongo

- Read

```
> db.users.find()
[{ "_id" : ObjectId("52f1014ace52721056688d7a"),
"firstName" : "Nimit", "lastName" : "Maru",
"username" : "nimitmaru", "password" :
"74824h234b280412312fdgsdgae" }]

> db.users.findOne()
{
  "_id" : ObjectId("52f1014ace52721056688d7a"),
  "firstName" : "Nimit",
  "lastName" : "Maru",
  "username" : "nimitmaru",
  "password" : "74824h234b280412312fdgsdgae"
}
```

CRUD in Mongo

- Update

```
> db.users.update(
     {"_id":ObjectId("52…0d7a”},
     {"username": "nimz"})

{
  "_id" : ObjectId("52f1014ace52721056688d7a"),
  "firstName" : "Nimit",
  "lastName" : "Maru",
  "password" : "74824h234b280412312fdgsdgae",
  "username" : "nimz"
}
```

CRUD in Mongo

- Delete

```
db.users.remove({_id:
ObjectId("52f1014ace52721056688d7a")})
> db.users.count()
0
```

What's Mongo Good At?

- Where you have a changing, flexible schema
- Where you may need scale
- Other specialized use cases:
  - Archiving/Event Logging
  - Documents/Content Management
  - Gaming
  - Mobile & Location Services
  - Agile Development
  - Real-Time Stats/Analysis

What's Mongo Bad At?

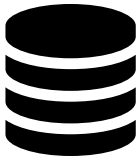
- Transactional Data like Banking or Accounting
  - Not ACID Compliant
  - ACID: Atomicity, Consistency, Isolation, Durability
- Where you may need or want SQL for whatever reason

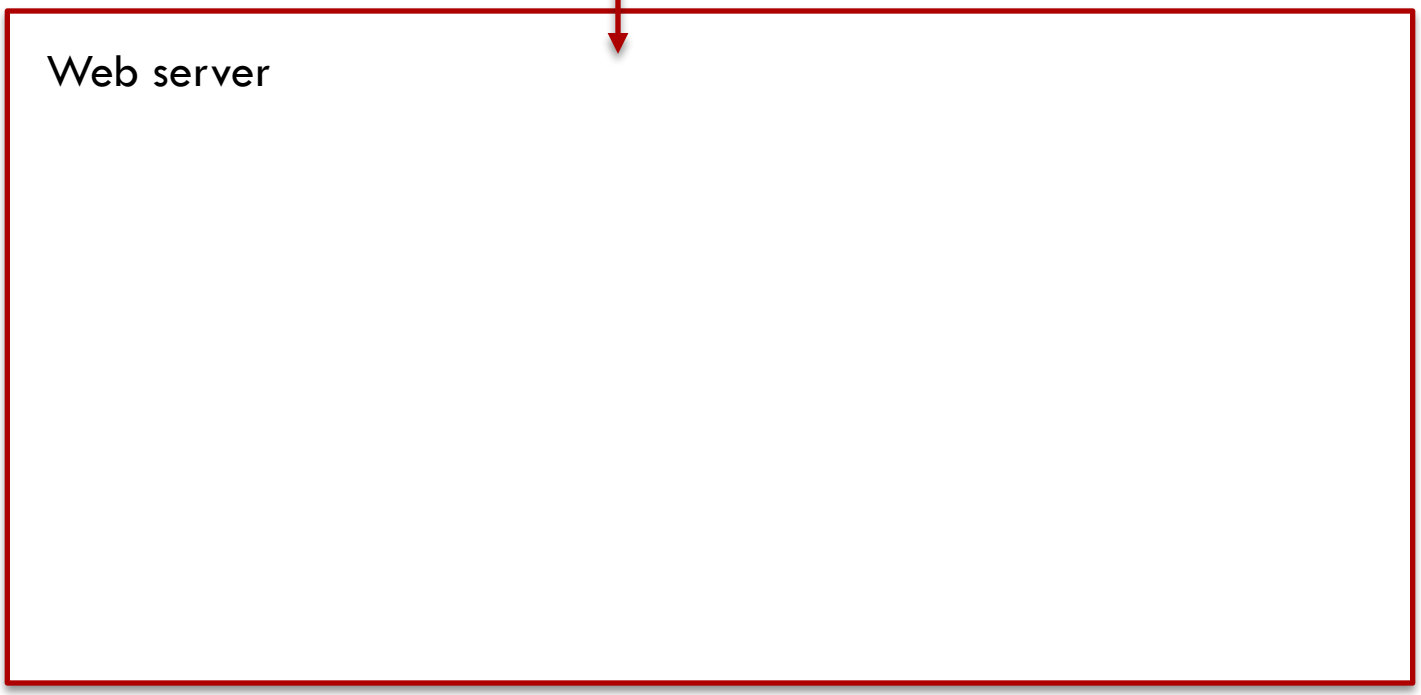
Lots of Features

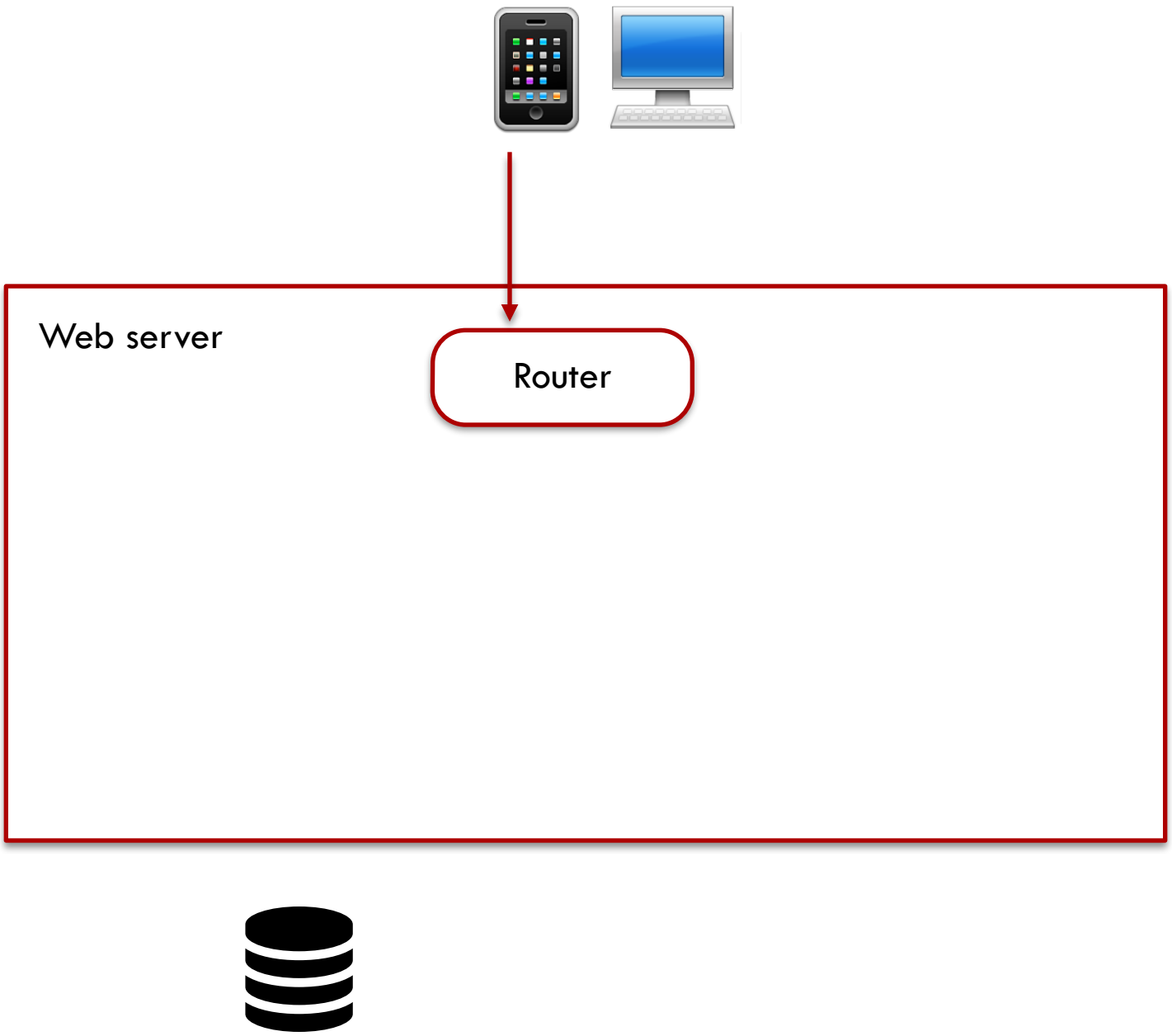
- Geo-spatial support
  - Spherical Searches
  - Inclusion search (for searches within a specified polygon)
- Easy horizontal-scaling (add and remove servers as needed)
- Aggregations (Map-Reduce)
- Stored functions
  - Let's you store JS functions on the server and run them later from MongoDB commands

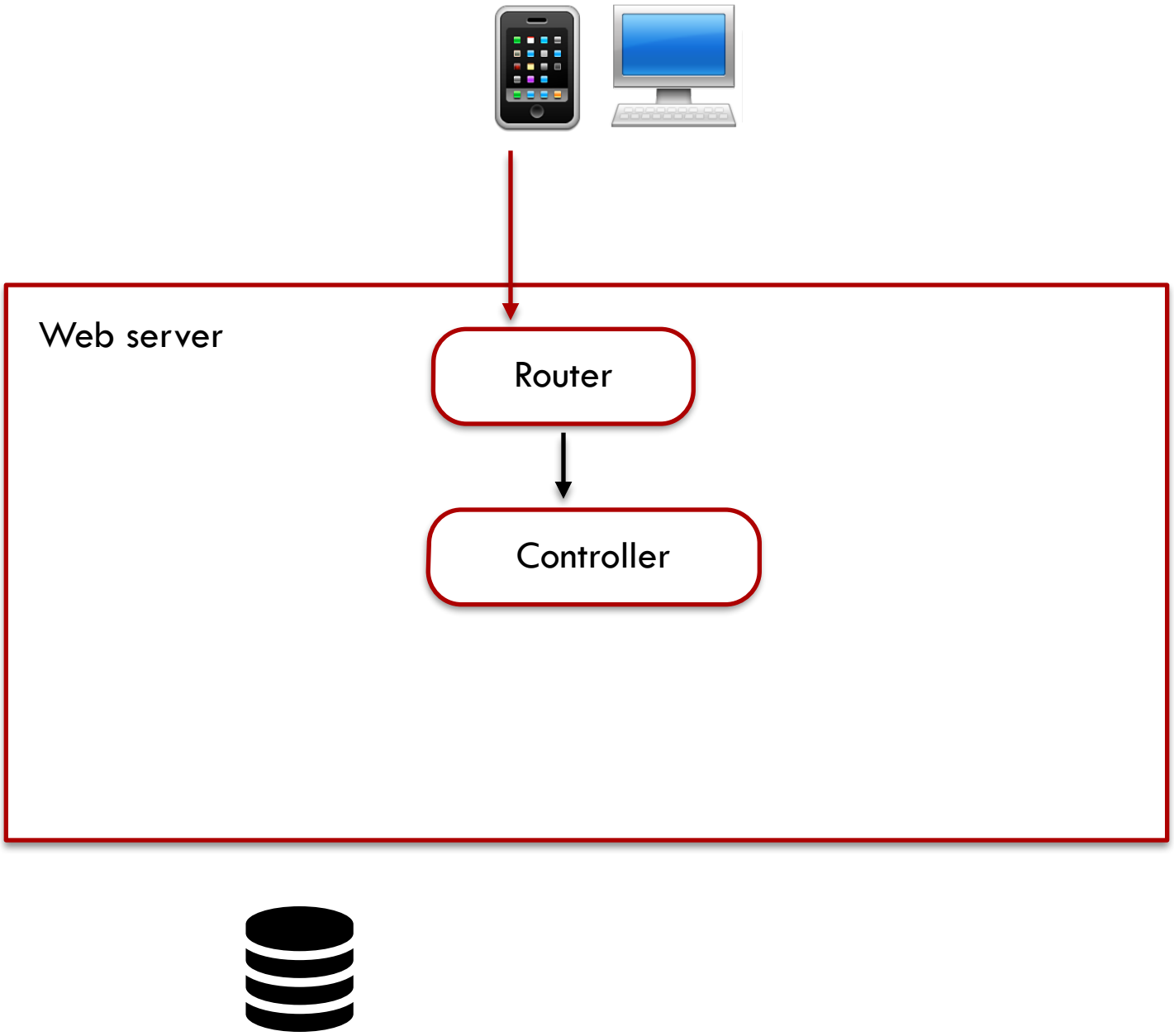
MongoDB with Node.js/Express

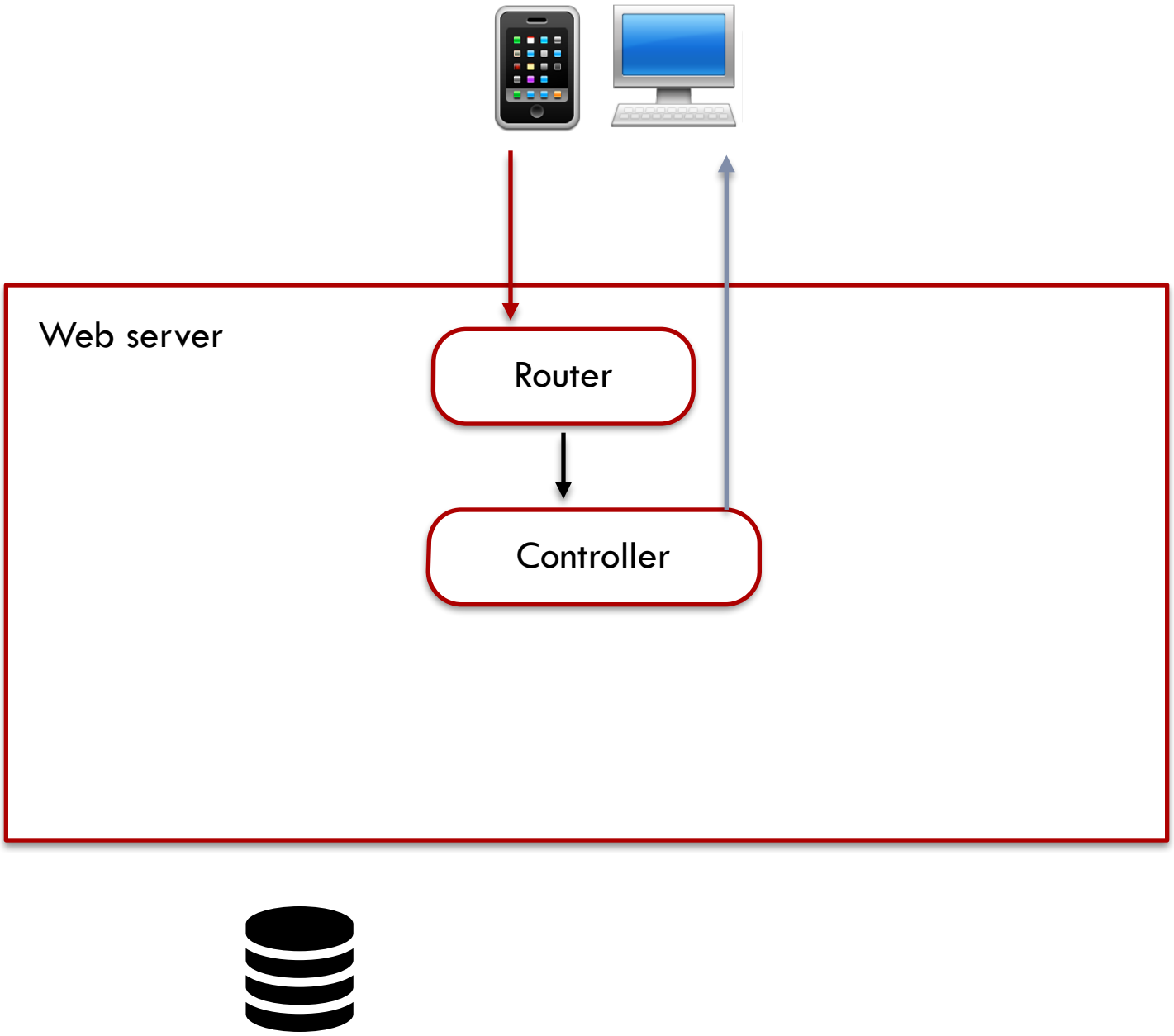
Using Mongoose

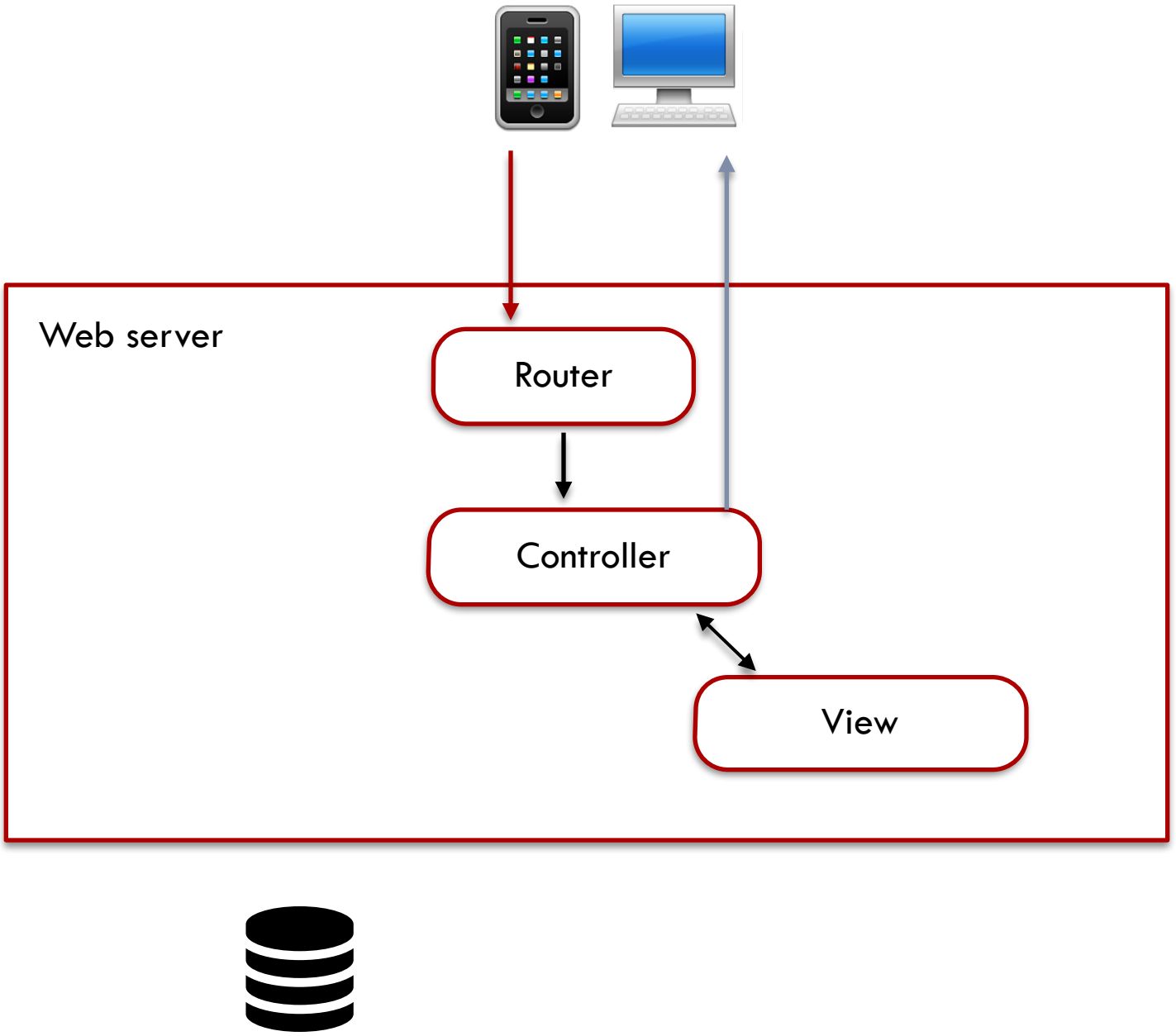


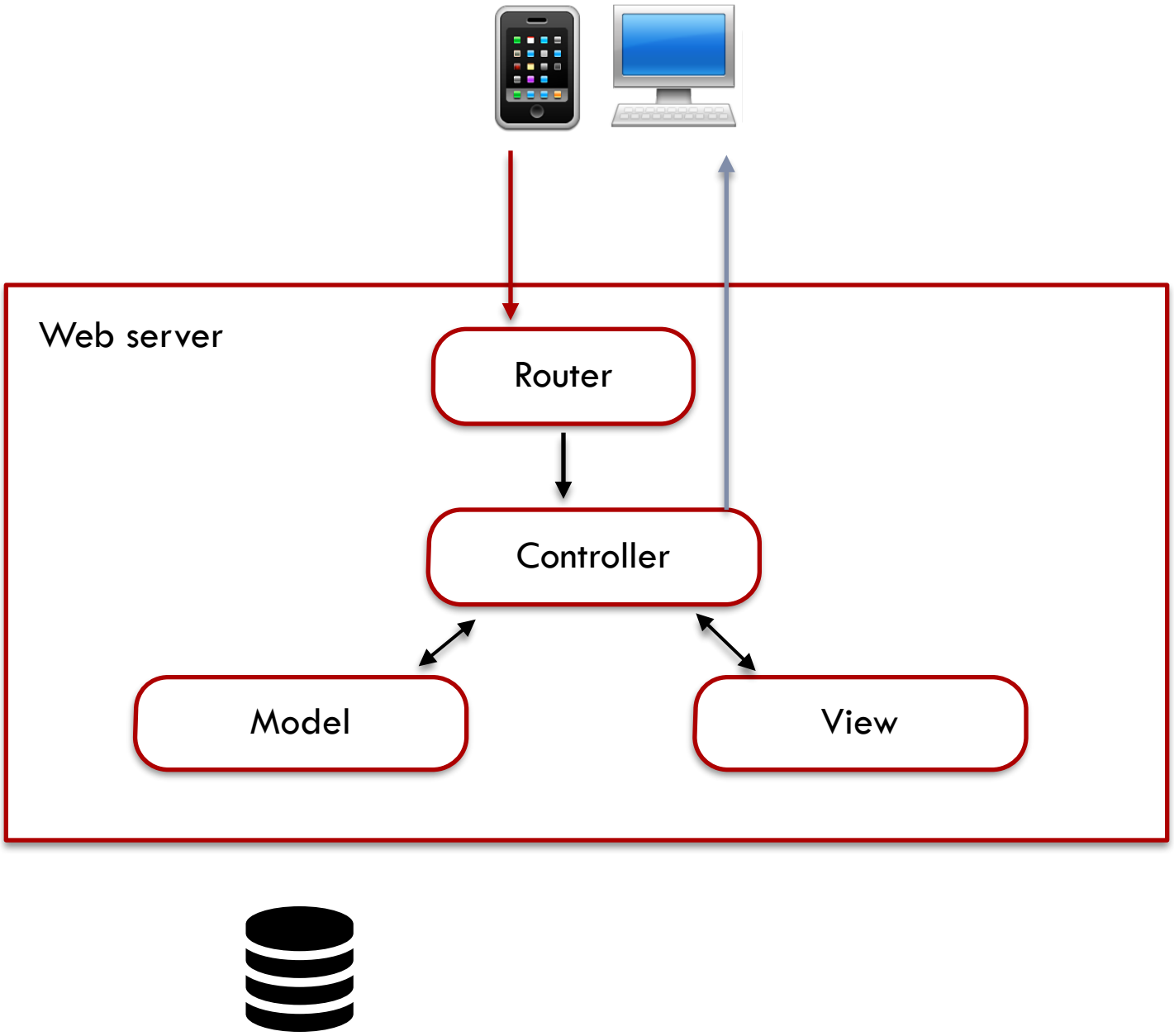

Web server

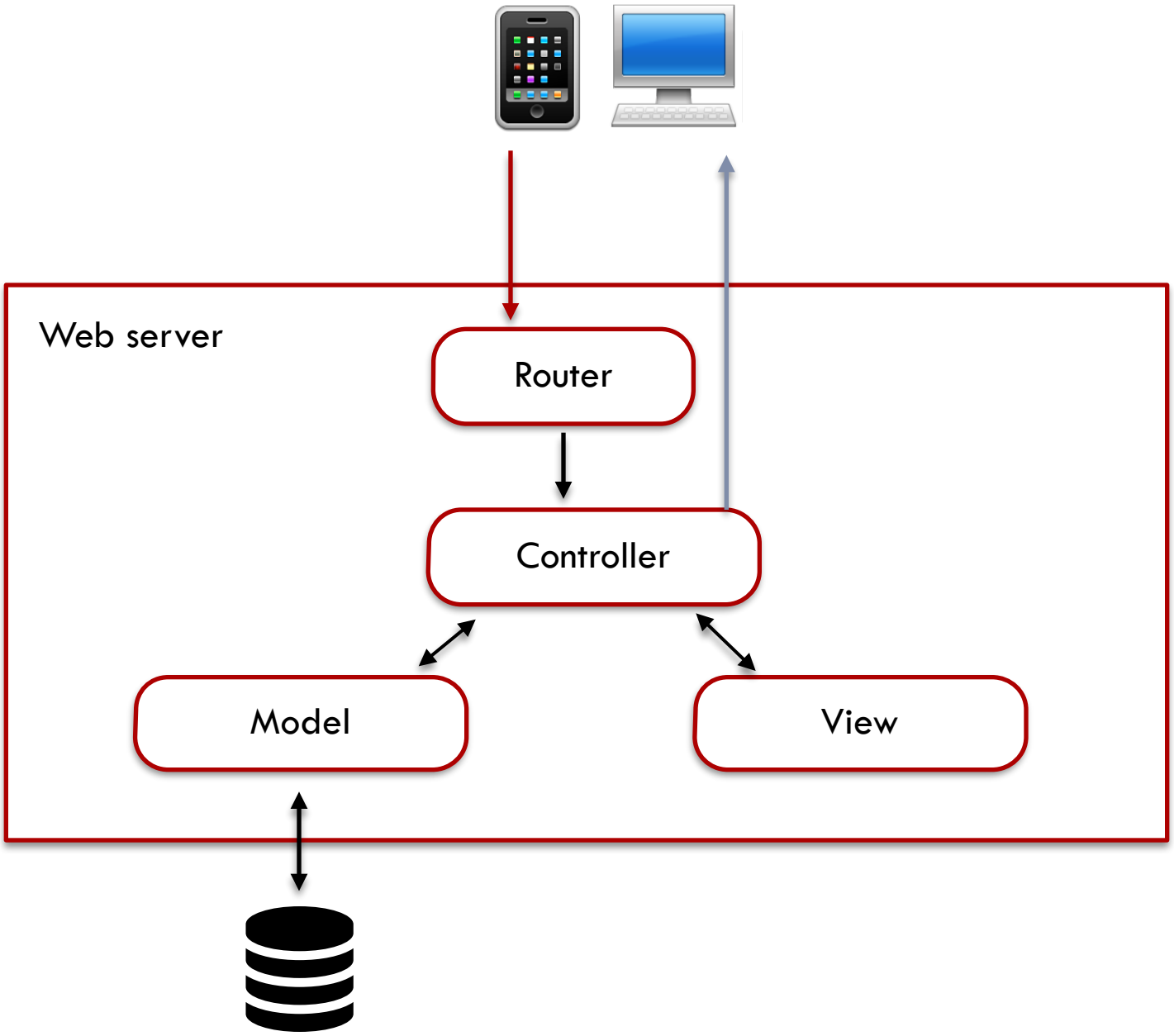

Web server
Router


Web server
Router
Controller


Web server
Router
Controller


Web server
Router
Controller
View


Web server
Router
Controller
Model
View


Web server
Router
Controller
Model
View

Data in a database…

```
{
  "_id" : ObjectId(“52f1014ace52721056688d7a"),
  "firstName" : "Nimit",
  "lastName" : "Maru",
  "username" : "nimitmaru",
  "password" : "74824h234b280412312fdgsdgae”,
  “email" : “nimit@fullstackacademy.com”
}
```

```
{
  "_id" : ObjectId("52f1014ace52721056688d7a"),
  "firstName" : "Nimit",
  "lastName" : "Maru",
  "username" : "nimitmaru",
  "password" : "74824h234b280412312fdgsdgae",
  "email" : "nimit@fullstackacademy.com"
}
```

```
{
  "_id" : ObjectId("52f1014ace52721056688d7a"),
  "firstName" : "Nimit",
  "lastName" : "Maru",
  "username" : "nimitmaru",
  "password" : "74824h234b280412312fdgsdgae",
  "email" : "nimit@fullstackacademy.com"
}
```

```
{
  "_id" : ObjectId("52f1014ace52721056688d7a"),
  "firstName" : "Nimit",
  "lastName" : "Maru",
  "username" : "nimitmaru",
  "password" : "74824h234b280412312fdgsdgae",
  "email" : "nimit@fullstackacademy.com",
  "hashPassword" : function() { … },
  "sendEmail" : function() { … },
  "fullName" : function() { return this.firstName + " " + … }
}
```

```
{
  "_id" : ObjectId("52f1014ace52721056688d7a"),
  "firstName" : "Nimit",
  "lastName" : "Maru",
  "username" : "nimitmaru",
  "password" : "74824h234b280412312fdgsdgae",
  "email" : "nimit@fullstackacademy.com"
}
```

Mongoose

- Mongoose is an ODM (Object-Document Mapper) that makes it easy to access MongoDB from JavaScript
- Features I love in Mongoose:
  - Schema Modeling/Validation
  - Hooks (before save, run x function; before delete, run y)
  - Define "classes" for collections (give docs methods)
  - Getters and Setters (first name, last name ⇛ “name”

Steps to using Mongoose

1. Create a Schema for a Collection
2. Convert it into a Mongoose Model
3. Use Model to work with Documents

1. Create a Schema

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

var blogSchema = new Schema({
  title:  String,
  author: String,
  body:   String,
  comments: [{ body: String, date: Date }],
  date: { type: Date, default: Date.now },
  hidden: Boolean,
  meta: {
    votes: Number,
    favs:  Number
  }
});
```

2. Convert Schema into Model

```
// convert into a Model
var Blog = mongoose.model('Blog', blogSchema);
```

3. Use Model to CRUD Documents

```
var myBlog = new Blog({
  title: "Nimit's Blog",
  author: "Nimit"
});

myBlog.save(function(err, myBlog) {
  if(err)
  myBlog.title
});

myBlog.find({ title: /Nimit/ });
```

Workshop

- WikiStack
  - Walk you through installing and using MongoDB
  - Wikipedia for Fullstackers
  - Application of everything we've learned so far
  - We will do intermediate reviews to ensure everyone is on track