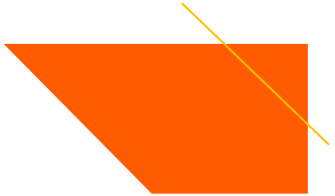
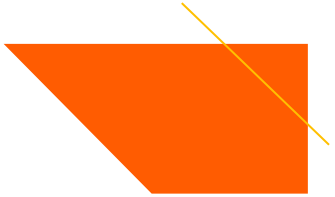
Web Application Development

COSI 152A



Main Point Preview

- In this lesson, you will learn to:
  - apply a more object-oriented approach to your data.
  - install the Mongoose package
  - convert the application data to fit a model structure.
  - build the first model and schema

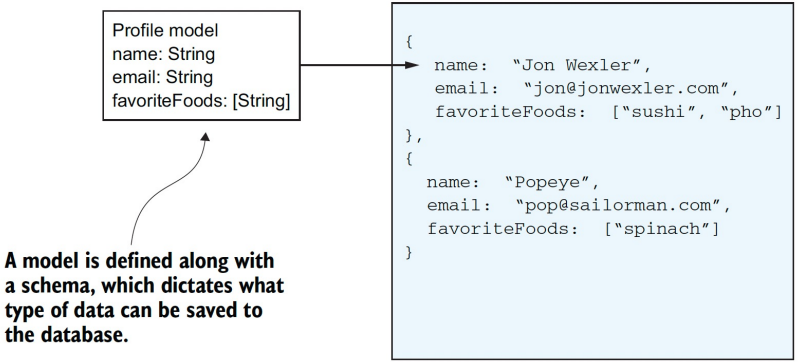


Setting Up Mongoose

- Mongoose package assists with the communication between a Node.js application and its database.
- Mongoose is an object-document mapper (ODM) that allows you to run MongoDB commands in a way that preserves the object-oriented structure of your application.
  - MongoDB may not keep one saved document consistent with the next.
  - Mongoose offers tools to build models with schemas deciding what type of data can be saved.

Model

- In MVC architecture, controllers communicate with both views and models to ensure that the correct data flows through the application.
- A model is like a class for a JavaScript object that Mongoose uses to organize your database queries.
- Install Mongoose and see what a model looks like in your application



Installing Mongoose

- To install Mongoose, run: `npm i mongoose -S`
- You no longer need to require mongodb or use any of the MongoDB code from the previous lesson.
- Require mongoose into the application file.
- Set up the application's connection to your MongoDB database.

```
const mongoose = require("mongoose");
mongoose.connect(
  "mongodb://localhost:27017/recipe_db",
  {useNewUrlParser: true}
);
const db = mongoose.connection;
```

Require mongoose.

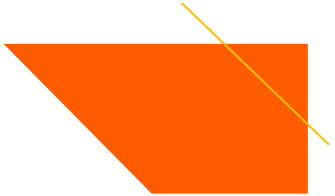
Set up the connection to your database.

Assign the database to the db variable.

Testing Mongoose

- You can log a message as soon as the database is connected:

The database connection runs the code in the callback function only upon receiving an "open" event from the database.



Creating a Schema

- A schema is like a class definition in some languages
  - Or a blueprint for how data are organized for specific objects in application.
- Create schemas to avoid inconsistent data
  - E.g., create a schema stating that all contact objects need to have an email field
  - E.g., create a schema for the subscriber to add a newsletter subscription form to your application,

Creating a Schema for Subscribers

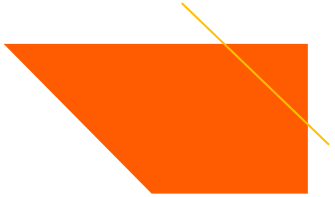
- mongoose.Schema offers a constructor that allows you to build a schema object with the given parameters.

```
const subscriberSchema = mongoose.Schema({
  name: String,
  email: String,
  zipCode: Number
});
```

Add schema properties.

Create a new schema with mongoose.Schema.

- According to this schema someone's name can't be a number.



Thank You!