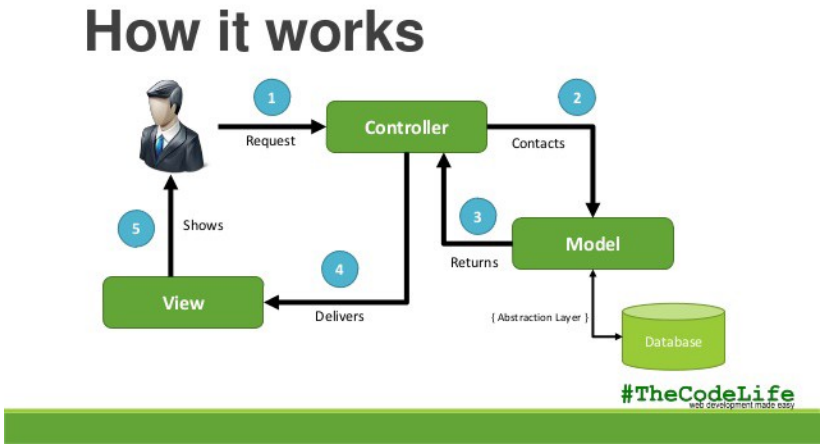
Unit 15.1 Highlights

MVC - (Model-View-Controller) Design pattern that separates an application into three main logical components: the model, the view, and the controller. Each of these components are built to handle specific development aspects of an application.



ORM - Object Relational Mapping - Programming technique for converting data between incompatible type systems using object-oriented programming languages. This creates, in effect, a "virtual object database" that can be used from within the programming language.
Sequelize (sequelizejs.com) - Promise-based Node v4+ ORM which supports dialects of PostgreSQL, MySQL, SQLite & MSSQL

Sequelize Directory Structure

```
.
├── app
│   └── config
│       └── connection.js
│   └── models
│       └── object.js
│   └── public
│       └── css
│       └── js
│       └── index.html
│   └── routes
│       └── api-routes.js
└── package.json
└── server.js
```

Ex. 1

```
//============ app/config/connection.js (MySQL config, connection statements)==============//
var Sequelize = require("sequelize");

// Creates mySQL connection using Sequelize, the empty string in the third argument spot is our password.
var sequelize = new Sequelize("database_name", "username", "password", {
  host: "localhost",
  port: 8889,
  dialect: "mysql",
  pool: {
    max: 5,
    min: 0,
    idle: 10000
  }
});

// Exports the connection for other files to use
module.exports = sequelize;
```

```
//============ app/models/school.js (object relational mapping)==============//

// Sequelize (capital) references the standard library
var Sequelize = require("sequelize");
// sequelize (lowercase) references my connection to the DB.
var sequelize = require("../config/connection.js");

// Creates a "School" model that matches up with DB
var School = sequelize.define("school", {
  name: Sequelize.STRING,
  location: Sequelize.STRING,
  region: Sequelize.STRING,
  code: Sequelize.INTEGER
});

// Syncs with DB
School.sync();

// Makes the School Model available for other files (will also create a table)
module.exports = School;


//============ app/routes/api-routes.js (entry point)==============//

var School = require("../models/school.js");


module.exports = function(app) {

  // Get all Schools
  app.get("/api/all", function(req, res) {

    // Finding all Schools, and then returning them to the user as JSON.
    // Sequelize queries are asynchronous, which helps with perceived speed.
    // If we want something to be guaranteed to happen after the query, we'll use
    // the .then function
    School.findAll({}).then(function(results) {
      // results are available to us inside the .then
      res.json(results);
    });

  });

  // Add a School
  app.post("/api/new", function(req, res) {

    console.log("School Data:");
    console.log(req.body);

   School.create({
      name: req.body.name,
      location: req.body.location,
      region: req.body.region,
      code: req.body.code
    }).then(function(results) {
      // `results` here would be the newly created school
      res.end();
    });

  });

};
```

```
//============ ./server.js (entry point)==============//
// Dependencies
var express = require("express");
var bodyParser = require("body-parser");

// Sets up the Express App
// =============================================================
var app = express();
var PORT = process.env.PORT || 8080;

// Sets up the Express app to handle data parsing

// parse application/x-www-form-urlencoded
app.use(bodyParser.urlencoded({ extended: true }));
// parse application/json
app.use(bodyParser.json());

// Static directory
app.use(express.static("app/public"));

// Routes
// =============================================================
require("./app/routes/api-routes.js")(app);
require("./app/routes/html-routes.js")(app);

// Starts the server to begin listening
// =============================================================
app.listen(PORT, function() {
  console.log("App listening on PORT " + PORT);
});
```

Sequelize CLI -Sequelize Command Line Interface

Ex. 2 (command line)

```
Install CLI globally with
npm install -g sequelize-cli

//run commands with sequelize
init:config //Initializes configuration
init:models //Initializes models
```