

# Build a simple app using Node JS and MySQL.

 [dev.to/achowba/build-a-simple-app-using-node-js-and-mysql-19me](https://dev.to/achowba/build-a-simple-app-using-node-js-and-mysql-19me)

Atauba Prince

Hello world!. In this tutorial, we are going to build a simple CRUD application using Node JS and MySQL.

## What we will build?



The image above shows the app. It is an application that lets you add players to a database and also display their details from the database. You can also delete and edit player details.

## Prerequisites

Before you join this tutorial it is assumed that you meet the requirements listed below:

- Node JS installed on your PC.
- Basic understanding of Node JS and Express JS.
- Knowledge of SQL, you should know and understand how to query a database.
- phpmyadmin installed on your PC. I recommend installing xampp as it already contains phpmyadmin in it.
- Understand how to use templating engines -- we are going to be using ejs in this tutorial).
- A text editor or IDE of your choice.

## Folder Structure

---

This is how the project will be structured.

```
├─ node-mysql-crud-app (main directory)
  │
  ├─ node_modules
  │
  ├─ public
  │   │
  │   └─ assets
  │       └─ img
  │
  ├─ routes
  │   │
  │   ├─ index.js
  │   └─ player.js
  │
  ├─ views
  │   │
  │   ├─ partials
  │   │   └─ header.ejs
  │   │
  │   ├─ index.ejs
  │   │
  │   ├─ add-player.ejs
  │   │
  │   └─ edit-player.ejs
  │
  └─ app.js
```

## Creating the directory for the project

---

Open the command prompt in a suitable directory and type the following command:

```
mkdir node-mysql-crud-app
```

then change to the directory by typing the following command

```
cd node-mysql-crud-app
```

## Initialize the Project

---

Open your command prompt in your project directory and type the command below:

```
npm init
```

## Install required modules.

---

The following modules are going to be needed to successfully build the app.

Type the following command to install the first 7 modules as dependencies.

```
npm install express express-fileupload body-parser mysql ejs req-flash --save
```

Then type the following command to install the last module globally on your PC.

```
npm install nodemon -g
```

## Creating the database for the app

---

Copy the command below and navigate to your phpmyadmin dashboard and execute the following query in the console (usually found at the bottom of the page) in order to create database and table for the app.

```
CREATE DATABASE socka;
CREATE TABLE IF NOT EXISTS `players` (
  `id` int(5) NOT NULL AUTO_INCREMENT,
  `first_name` varchar(255) NOT NULL,
  `last_name` varchar(255) NOT NULL,
  `position` varchar(255) NOT NULL,
  `number` int(11) NOT NULL,
  `image` varchar(255) NOT NULL,
  `user_name` varchar(20) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1;
```

## Adding the views

---

### header.ejs

---

The **header.ejs** file is going to be in the */views/partials* folder where it is going to be included in the rest of the project.

```

<!doctype html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport"
    content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-
scale=1.0, minimum-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
crossorigin="anonymous">
  <!--<link rel="stylesheet" href="/assets/css/custom.css">-->
  <title><%= title %></title>
</head>
<style>
  .table-wrapper {
    margin-top: 50px;
  }

  .player-img {
    width: 40px;
    height: 40px;
  }

  .add-player-form {
    margin-top: 50px;
  }
</style>
<body>
<div class="page-wrapper">
  <nav class="navbar navbar-light bg-light">
    <span class="navbar-brand mb-0 h1" ><a href="/">Socka Players</a></span>
    <a class="float-right" href="/add" title="Add a New Player">Add a Player</a>
  </nav>

```

## index.ejs

---

This is the homepage of the app which contains a table to display a list of all the players.

```

<% include partials/header.ejs %>
<div class="table-wrapper">
  <% if (players.length > 0) {%>
    <table class="table table-hovered">
      <thead class="thead-dark">
        <tr>
          <th scope="col">ID</th>
          <th scope="col">Image</th>
          <th scope="col">First Name</th>
          <th scope="col">Last Name</th>
          <th scope="col">Position</th>
          <th scope="col">Number</th>
          <th scope="col">Username</th>
          <th scope="col">Action</th>
        </tr>
      </thead>
      <tbody>
        <% players.forEach((player, index) => { %>
          <tr>
            <th scope="row"><%= player.id %></th>
            <td></td>
            <td><%= player.first_name %></td>
            <td><%= player.last_name %></td>
            <td><%= player.position %></td>
            <td><%= player.number %></td>
            <td><%= player.user_name %></td>
            <td>
              <a href="/edit/<%= player.id %>" target="_blank"
rel="noopener" class="btn btn-sm btn-success">Edit</a>
              <a href="/delete/<%= player.id %>" class="btn btn-sm
btn-danger">Delete</a>
            </td>
          </tr>
        <% }) %>
      </tbody>
    </table>
  <% } else { %>
    <p class="text-center">No players found. Go <a href="/add" >here</a> to
add players.</p>
  <% } %>
</div>
</div>
</body>
</html>

```

## add-player.ejs

---

This page contains the form to add a new player to the database.

```

<% include partials/header.ejs %>
<div class="container">
  <% if (message != '') { %>
    <p class="text-center text-danger"><%= message %></p>
  <% } %>
  <form class="add-player-form" action="" method="post" enctype="multipart/form-
data">
    <div class="form-row">
      <div class="form-group col-md-4">
        <input type="text" class="form-control" name="first_name"
id="first-name" placeholder="First Name" required>
      </div>
      <div class="form-group col-md-4">
        <input type="text" class="form-control" name="last_name" id="last-
name" placeholder="Last Name" required>
      </div>
      <div class="form-group col-md-4">
        <input type="text" class="form-control" name="username"
id="username" placeholder="Username" required>
      </div>
    </div>
    <div class="form-row">
      <div class="form-group col-md-6">
        <input type="number" class="form-control" name="number" id="number"
placeholder="Number" required>
      </div>
      <div class="form-group col-md-6">
        <select id="position" name="position" class="form-control"
required>
          <option selected disabled>Choose position</option>
          <option>Goalkeeper</option>
          <option>Defender</option>
          <option>Midfielder</option>
          <option>Forward</option>
        </select>
      </div>
      <div class="col-md-12">
        <label for="player-img"><b>Player Image</b></label><br>
        <input type="file" name="image" id="player-img" class="" required>
      </div>
    </div>
    <button type="submit" class="btn btn-primary float-right">Add
Player</button>
  </form>
</div>
</div>
</body>
</html>

```

edit-player.ejs

---

This page contains the form to edit a player added to the database.

```
<% include partials/header.ejs %>
<div class="container">
  <% if (message) { %>
    <p class="text-center text-danger"><%= message %></p>
  <% } %>

  <% if (player) { %>
    <form class="add-player-form" action="" method="post" enctype="multipart/form-
data">
      <div class="form-row">
        <div class="form-group col-md-4">
          <label for="first-name">First Name</label>
          <input type="text" class="form-control" name="first_name"
id="first-name" value="<%= player.first_name %>" required>
        </div>
        <div class="form-group col-md-4">
          <label for="last-name">Last Name</label>
          <input type="text" class="form-control" name="last_name" id="last-
name" value="<%= player.last_name %>" required>
        </div>
        <div class="form-group col-md-4">
          <label for="username">Username</label>
          <input type="text" class="form-control" name="username"
id="username" value="<%= player.user_name %>" required disabled title="Username cannot
be edited.">
        </div>
      </div>
      <div>
        <div class="form-row">
          <div class="form-group col-md-6">
            <label for="number">Number</label>
            <input type="number" class="form-control" name="number" id="number"
placeholder="Number" value="<%= player.number %>" required>
          </div>
          <div class="form-group col-md-6">
            <label for="position">Position</label>
            <select id="position" name="position" class="form-control"
required>
              <option selected><%= player.position %></option>
              <option>Goalkeeper</option>
              <option>Centre Back</option>
              <option>Right Back</option>
              <option>Left Back</option>
              <option>Defensive Midfielder</option>
              <option>Central Midfielder</option>
              <option>Attacking Midfielder</option>
              <option>Right Wing Forward</option>
              <option>Left Wing Forward</option>
              <option>Striker</option>
            </select>
          </div>
        </div>
      </div>
    </form>
  <% } %>
</div>
```

```

        <button type="submit" class="btn btn-success float-right">Update
Player</button>
    </form>
    <% } else { %>
        <p class="text-center">Player Not Found. Go <a href="/add">here</a> to add
players.</p>
    <% } %>
</div>
</div>
</body>
</html>

```

## Working on the server.

---

### App.js

---

```

const express = require('express');
const fileUpload = require('express-fileupload');
const bodyParser = require('body-parser');
const mysql = require('mysql');
const path = require('path');
const app = express();

// const {getHomePage} = require('./routes/index');
// const {addPlayerPage, addPlayer, deletePlayer, editPlayer, editPlayerPage} =
require('./routes/player');
const port = 5000;

// create connection to database
// the mysql.createConnection function takes in a configuration object which contains
host, user, password and the database name.
const db = mysql.createConnection ({
    host: 'localhost',
    user: 'root',
    password: '',
    database: 'socka'
});

// connect to database
db.connect((err) => {
    if (err) {
        throw err;
    }
    console.log('Connected to database');
});
global.db = db;

// configure middleware
app.set('port', process.env.port || port); // set express to use this port
app.set('views', __dirname + '/views'); // set express to look in this folder to
render our view
app.set('view engine', 'ejs'); // configure template engine

```



```

app.use(bodyParser.urlencoded({ extended: false }));
app.use(bodyParser.json()); // parse form data client
app.use(express.static(path.join(__dirname, 'public'))); // configure express to use
public folder
app.use(fileUpload()); // configure fileupload

// routes for the app
/*
app.get('/', getHomePage);
app.get('/add', addPlayerPage);
app.get('/edit/:id', editPlayerPage);
app.get('/delete/:id', deletePlayer);
app.post('/add', addPlayer);
app.post('/edit/:id', editPlayer);
*/

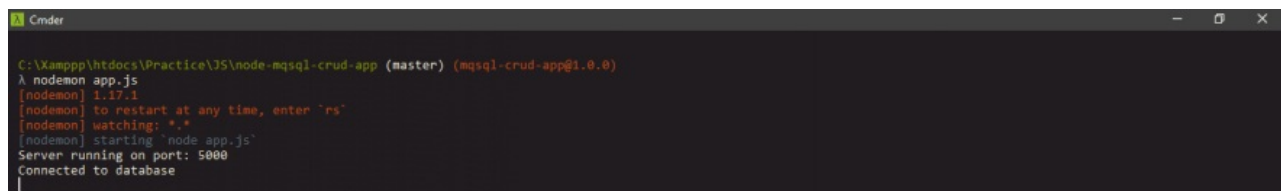
// set the app to listen on the port
app.listen(port, () => {
  console.log(`Server running on port: ${port}`);
});

```

In the above code, the modules are required and then a connection to the database is created. The **mysql.createConnection** function takes in an object which contains the configuration of the database being connected to. In the next statement, the database is being connected. Run the code below on the command prompt to run the server.

nodemon app.js

Your console should show the result below:



```

C:\xampp\htdocs\Practice\JS\node-mysql-crud-app (master) (mysql-crud-app@1.0.0)
A nodemon app.js
[nodemon] 1.19.1
[nodemon] to restart at any time, enter `rs`
[nodemon] watching: *.*
[nodemon] starting 'node app.js'
Server running on port: 5000
Connected to database

```

## Adding the routes

### index.js

Copy the code below into the index.js file in the **/routes** directory.

```

module.exports = {
  getHomePage: (req, res) => {
    let query = "SELECT * FROM `players` ORDER BY id ASC"; // query database to
    get all the players

    // execute query
    db.query(query, (err, result) => {
      if (err) {
        res.redirect('/');
      }
      res.render('index.ejs', {
        title: Welcome to Socka | View Players
        ,players: result
      });
    });
  },
};

```

The **db.query** function queries the database. It takes in the query and string and a callback which takes in two parameters, if the query is successful, the result is passed to the view in the **res.render** function.

## player.js

---

The **player.js** file is going to contain all the routes for the players page such as adding a player, updating a player's details and deleting a player.

```

const fs = require('fs');

module.exports = {
  addPlayerPage: (req, res) => {
    res.render('add-player.ejs', {
      title: Welcome to Socka | Add a new player
      ,message: ''
    });
  },
  addPlayer: (req, res) => {
    if (!req.files) {
      return res.status(400).send("No files were uploaded.");
    }

    let message = '';
    let first_name = req.body.first_name;
    let last_name = req.body.last_name;
    let position = req.body.position;
    let number = req.body.number;
    let username = req.body.username;
    let uploadedFile = req.files.image;
    let image_name = uploadedFile.name;
    let fileExtension = uploadedFile.mimetype.split('/')[1];
    image_name = username + '.' + fileExtension;

```

```

let usernameQuery = "SELECT * FROM `players` WHERE user_name = '" + username +
""";

db.query(usernameQuery, (err, result) => {
  if (err) {
    return res.status(500).send(err);
  }
  if (result.length > 0) {
    message = 'Username already exists';
    res.render('add-player.ejs', {
      message,
      title: Welcome to Socka | Add a new player
    });
  } else {
    // check the filetype before uploading it
    if (uploadedFile.mimetype === 'image/png' || uploadedFile.mimetype ===
'image/jpeg' || uploadedFile.mimetype === 'image/gif') {
      // upload the file to the /public/assets/img directory
      uploadedFile.mv(`public/assets/img/${image_name}`, (err ) => {
        if (err) {
          return res.status(500).send(err);
        }
        // send the player's details to the database
        let query = "INSERT INTO `players` (first_name, last_name,
position, number, image, user_name) VALUES ('" +
          first_name + "', '" + last_name + "', '" + position + "',
'" + number + "', '" + image_name + "', '" + username + "')";
        db.query(query, (err, result) => {
          if (err) {
            return res.status(500).send(err);
          }
          res.redirect('/');
        });
      });
    } else {
      message = "Invalid File format. Only 'gif', 'jpeg' and 'png' images
are allowed.";
      res.render('add-player.ejs', {
        message,
        title: Welcome to Socka | Add a new player
      });
    }
  }
});
},
editPlayerPage: (req, res) => {
  let playerId = req.params.id;
  let query = "SELECT * FROM `players` WHERE id = '" + playerId + "' ";
  db.query(query, (err, result) => {
    if (err) {
      return res.status(500).send(err);
    }
  }
}

```

```

        res.render('edit-player.ejs', {
            title: Edit Player
            ,player: result[0]
            ,message: ''
        });
    });
},
editPlayer: (req, res) => {
    let playerId = req.params.id;
    let first_name = req.body.first_name;
    let last_name = req.body.last_name;
    let position = req.body.position;
    let number = req.body.number;

    let query = "UPDATE `players` SET `first_name` = '" + first_name + "',
`last_name` = '" + last_name + "', `position` = '" + position + "', `number` = '" +
number + "' WHERE `players`.`id` = '" + playerId + "'";
    db.query(query, (err, result) => {
        if (err) {
            return res.status(500).send(err);
        }
        res.redirect('/');
    });
},
deletePlayer: (req, res) => {
    let playerId = req.params.id;
    let getImageQuery = 'SELECT image from `players` WHERE id = "' + playerId +
    ""';
    let deleteUserQuery = 'DELETE FROM players WHERE id = "' + playerId + ""';

    db.query(getImageQuery, (err, result) => {
        if (err) {
            return res.status(500).send(err);
        }

        let image = result[0].image;

        fs.unlink(`public/assets/img/${image}`, (err) => {
            if (err) {
                return res.status(500).send(err);
            }
            db.query(deleteUserQuery, (err, result) => {
                if (err) {
                    return res.status(500).send(err);
                }
                res.redirect('/');
            });
        });
    });
},
};
};
};

```

This file handles all the post and get requests for the players page. The add player function contains a function that uploads the player's image to the */public/assets/img* directory and sends the player's details to the database.

## Connecting the routes and the views

Go to **app.js** and uncomment the following lines

on line 8 and 9

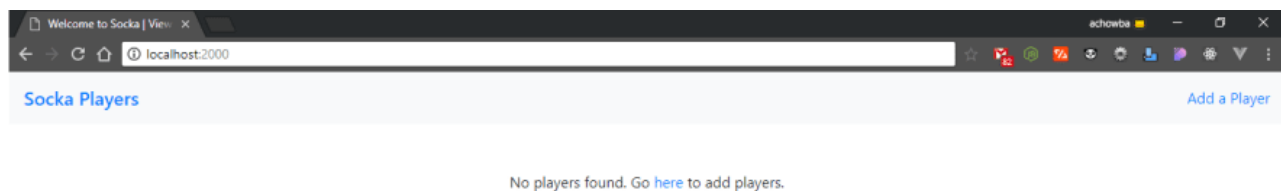
```
// const {getHomePage} = require('./routes/index');  
// const {addPlayerPage, addPlayer, deletePlayer, editPlayer, editPlayerPage} =  
require('./routes/player');
```

on line 40 - 47

```
/*  
app.get('/', getHomePage);  
app.get('/add', addPlayerPage);  
app.get('/edit/:id', editPlayerPage);  
app.get('/delete/:id', deletePlayer);  
app.post('/add', addPlayer);  
app.post('/edit/:id', editPlayer);  
*/
```

## Running the app

After removing the commented lines, check your command prompt to ensure your code has no errors, then head over to your browser and open <http://localhost:5000>. The index page will be shown and since no players have been added, the page will look similar to the one below:



Click the **Add a player** link on the page, the add player page will load and then fill the form to add a player. After adding a player the home page will display the players added in a table. Like this

Welcome to Socka | View

localhost:2000

socka

Socka Players

Add a Player

ID	Image	First Name	Last Name	Position	Number	Username	Action
1		John	Doe	Goalkeeper	8	@doe	<div>Edit</div> <div>Delete</div>

If errors are encountered, draw my attention in the comment section or check the repo of the project on [github](#).

The code is not perfect and please feel free to make it better and send a pull request.