

1. Server

For first step, please create database and table on phpmyadmin.

Database name: food, table name: users

The image consists of two screenshots. The top screenshot shows the phpMyAdmin interface with the 'food' database selected and the 'users' table created. The table structure is displayed with columns: first_name, last_name, email, password, id, month, day, year, phone, country, and zip. The bottom screenshot shows the VS Code editor with the database configuration in 'db.js' and the table schema in 'users.js'. Red arrows point from the table name 'users' in the phpMyAdmin table structure to the 'table name' in the VS Code 'db.js' file, and from the 'id' column in the table structure to the 'id' field in the 'users.js' file.

db.js

```
const Sequelize = require('sequelize');
const db = {};
const sequelize = new Sequelize('food', 'root', '', {
  host: 'localhost',
  dialect: 'mysql',
  operatorsAliases: false,
  pool: {
    max: 5,
    min: 0,
    acquire: 30000,
    idle: 10000
  }
});
```

users.js

```
const {
  Id,
  firstName,
  lastName,
  email,
  password,
  month,
  day,
  year,
  phone,
  country,
  zip
} = db.Sequelize.define('users', {
  id: {
    type: Sequelize.INTEGER,
    primaryKey: true,
    autoIncrement: true,
  },
  firstName: {
    type: Sequelize.STRING,
  },
  lastName: {
    type: Sequelize.STRING,
  },
  email: {
    type: Sequelize.STRING,
  },
  password: {
    type: Sequelize.STRING,
  },
  month: {
    type: Sequelize.STRING,
  },
  day: {
    type: Sequelize.STRING,
  },
  year: {
    type: Sequelize.STRING,
  },
  phone: {
    type: Sequelize.STRING,
  },
  country: {
    type: Sequelize.STRING,
  },
  zip: {
    type: Sequelize.STRING,
  },
});
```

Next run the server code as following:

The screenshot shows the Visual Studio Code interface with the `server.js` file open in the editor. The file contains the following code:

```
1 var express = require('express');
2 var cors = require('cors');
3 var bodyParser = require('body-parser');
4 var app = express();
5 var port = process.env.PORT || 5000;
6
7 app.use(bodyParser.json());
8 app.use(cors());
9 app.use(bodyParser.urlencoded({extended: false}));
10
11 var Users = require('./routes/Users');
12
13 app.use('/users', Users);
14
15 app.listen(port, () => {
16   console.log('Server is running on port: ' + port);
17 });
```

The Explorer sidebar on the left shows the project structure, including `server.js`. The Terminal panel at the bottom shows the command `npm start` being executed, resulting in the output: `Server is running on port: 5000`.

2. My-app (frontend)

Run the frontend code as following:

The screenshot shows the Visual Studio Code interface with the `index.js` file open in the editor. The file contains the following code:

```
1 import React from 'react';
2 import ReactDOM from 'react-dom';
3 import './index.css';
4 import App from './App';
5 import { Provider } from 'react-redux';
6 import { ConnectedRouter } from 'connected-react-router';
7 import store, { history } from './redux/store';
8
9 ReactDOM.render(
10   <Provider store={store}>
11     <ConnectedRouter history={history}>
12       <App />
13     </ConnectedRouter>
14   </Provider>,
15   document.getElementById('root')
16 );
```

The Explorer sidebar on the left shows the project structure, including `index.js`. The Terminal panel at the bottom shows the command `npm start` being executed, resulting in the output: `Compiled with warnings.` and `./src/redux/actions/userAction.js`.