**Dokumentasi CRUD MongoDB, Node.js**

**Pemrograman Basis Data**

Kelas B

Agung Budi Prakoso (21120117140004)

Fiqri Abdilah (21120117130061)

Nobi Kharisma (21120117130048)

Petrick Jubel E (21120117120028)

Yusuf Valent A (21120117130046)

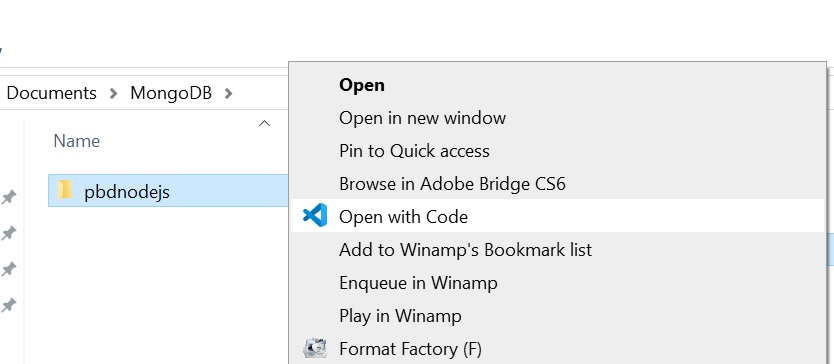
# Langkah Kerja

1. Download & install software yang diperlukan.

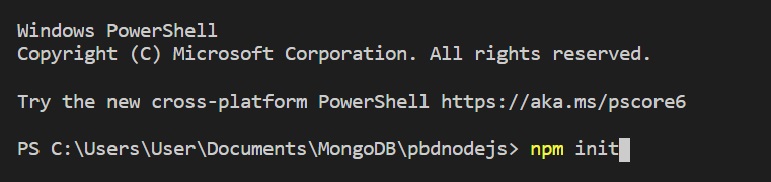
Visual Studio Code 🡪 <https://code.visualstudio.com/download>

MongoDB Community Server 🡪 <https://www.mongodb.com/try/download/community>

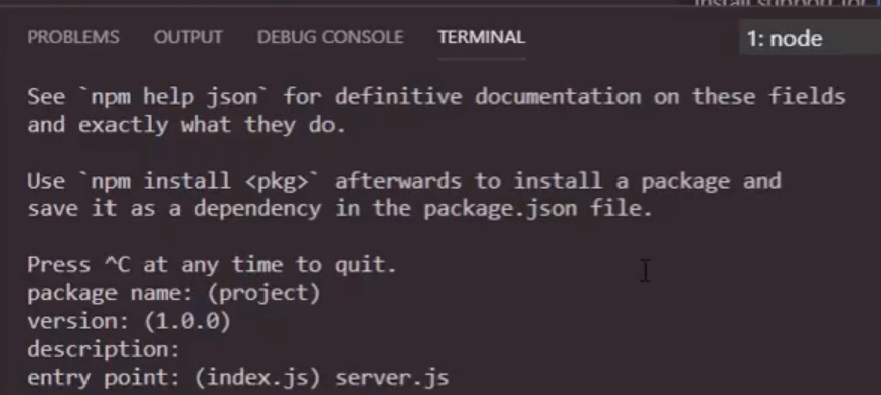
1. Buat folder project, kemudian buka menggunakan Visual Studio Code



1. Masuk ke mode terminal dengan menekan “ctrl+shift+`” . Kemudian tulis perintah “npm init” , lalu menekan enter.



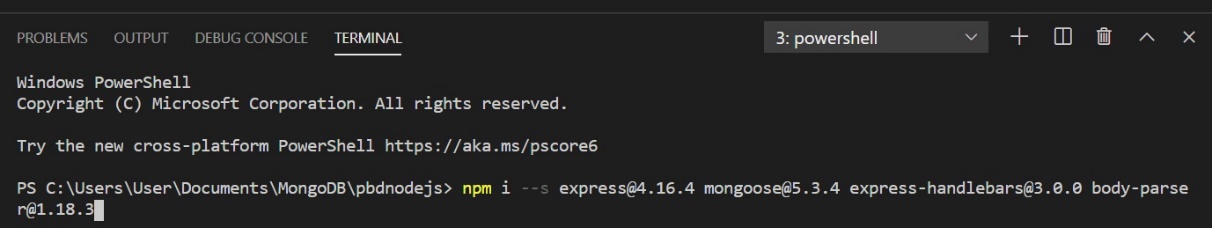
1. Menekan enter sampai pilihan entry point, kemudian menulis perintah “server.js” lalu menekan enter, seperti gambar di bawah ini.



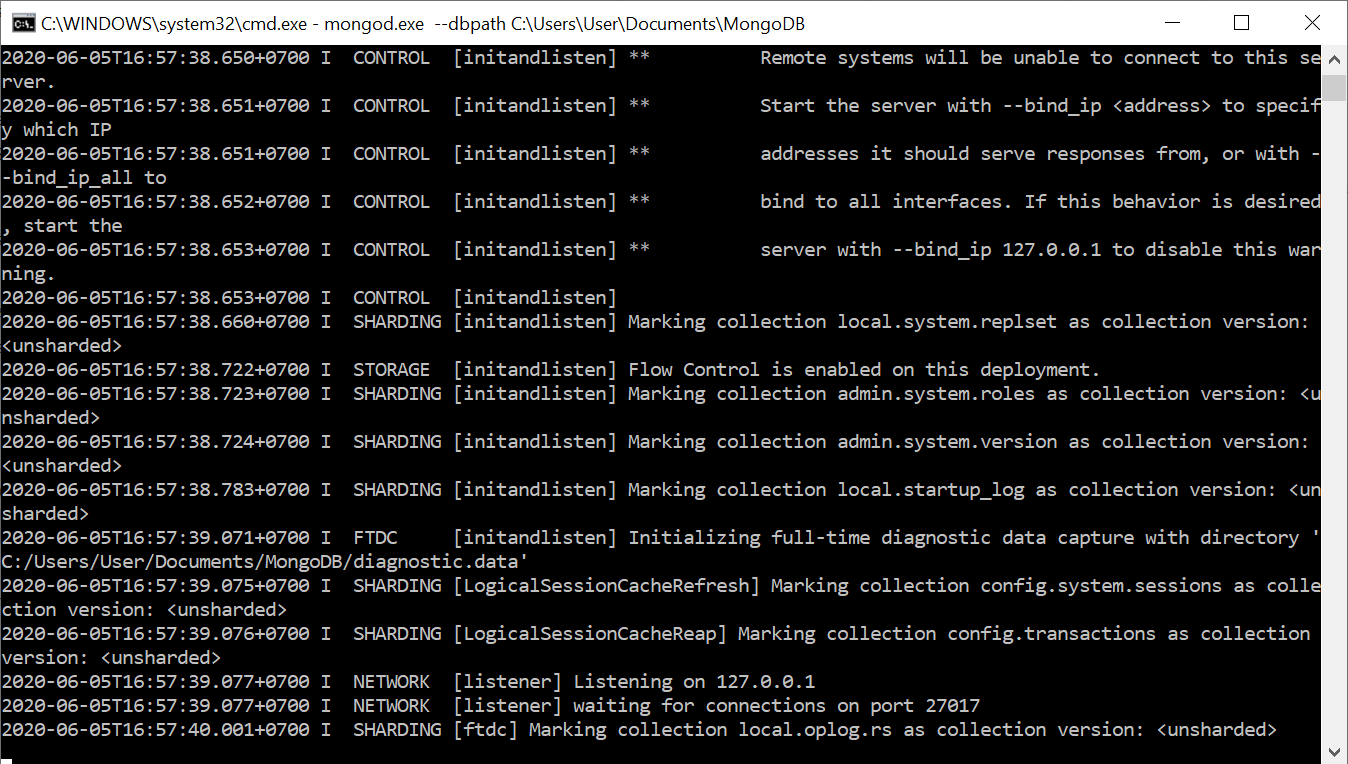
1. Tekan enter sampai selesai kemudian menulis perintah “yes” untuk menyelesaikan.



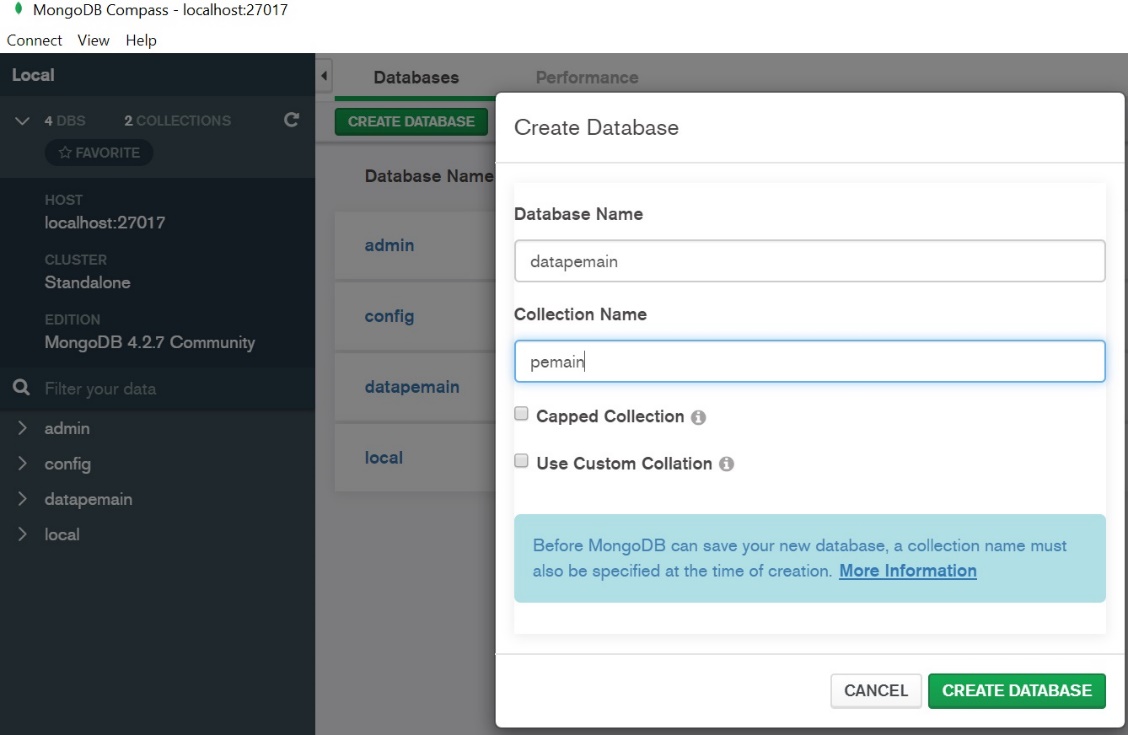
1. Menulis perintah “npm i --s express@4.16.4 mongoose@5.3.4 express-handlebars@3.0.0 body-parser@1.18.3” kemudian menekan enter.



1. Mengkoneksikan MongoDB dengan membuka CMD kemudian menuliskan perintah “cd C:\Program Files (x86)\MongoDB\mongodb-win32-x86\_64-2012plus-4.2.7\bin” tekan enter, perintah ini untuk berpindah lokasi ke MongoDB. Lalu menuliskan perintah ke-2 pada CMD untuk menjalankan Mongo DB “mongod.exe --dbpath C:\Users\User\Documents\MongoDB” tekan enter, perintah ini merupakan lokasi dari project yang dibuat. Jika berhasil maka tampilannya seperti gambar di bawah ini.

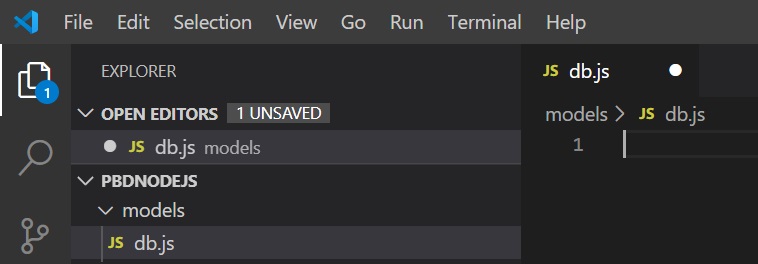


1. Membuka MongoDB Compass kemudian membuat database baru.

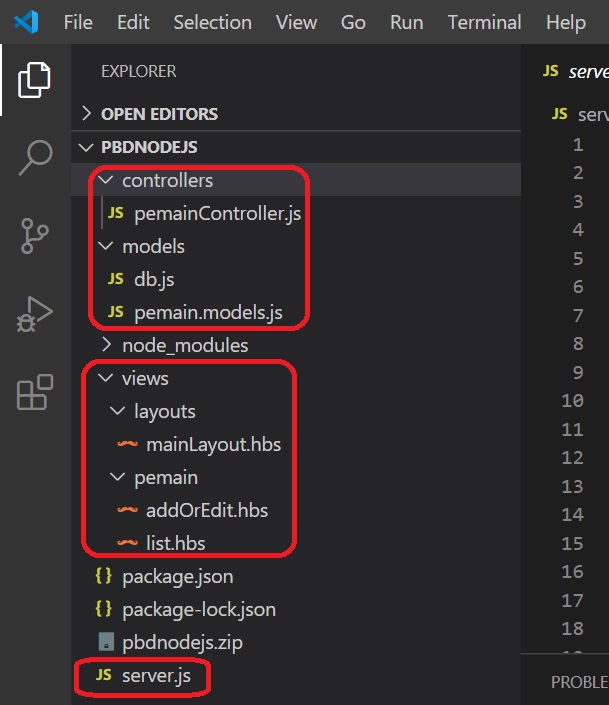


1. Mebuat folder “models”, kemudian membuat file “db.js” pada folder tersebut. Lalu masukkan source code di bawah ini pada file “db.js”. Ini sebagai koneksi ke database MongoDB.

|  |
| --- |
| const mongoose = require ('mongoose');  mongoose.connect('mongodb://localhost:27017/datapemain', { useNewUrlParser: true}, (err) =>{  if (!err) {console.log('MongoDB COnnection Succeeded')}  else {console.log('Error in DB Connection: ' + err)}  });  require('./pemain.models'); |



1. Buat folder & file .js .hbs seperti gambar di bawah ini.



1. Masukkan source di bawah ini sesuai dengan nama filenya.

* **pemainController.js**

|  |
| --- |
| const express = require('express');  var router = express.Router();  const mongoose = require('mongoose');  const Pemain = mongoose.model('Pemain');  router.get('/',(req,res)=>{  res.render("pemain/addOrEdit",{  viewTitle : "Tambah Pemain Bola"  });  });  router.post('/',(req,res)=>{  if (req.body.\_id == '')  insertRecord(req, res);  else  updateRecord(req, res);  });  function insertRecord(req, res){  var pemain = new Pemain();  pemain.fullName = req.body.fullName;  pemain.negara = req.body.negara;  pemain.klub = req.body.klub;  pemain.nomer = req.body.nomer;  pemain.save((err, doc) => {  if(!err)  res.redirect('pemain/list');  else{  if (err.name == 'ValidationError') {  handleValidationError(err, req.body);  res.render("pemain/addOrEdit", {  viewTitle: "Tambah Pemain Bola",  pemain: req.body  });  }  else  console.log('Error during record Insertion : + err');  }  });  }  function updateRecord(req, res){  Pemain.findOneAndUpdate({\_id: req.body.\_id}, req.body, { new: true }, (err, doc) =>{  if (!err) { res.redirect('pemain/list');}  else{  if (err,name == 'ValidationError') {  handleValidationError(err, req.body);  res.render("pemain/addOrEdit", {  viewTitle: 'Update Pemain Bola',  pemain: req.body  });  }  else  console.log('Error during record update : '+ err);  }  })  }  router.get('/list',(req,res)=>{  Pemain.find((err, docs) => {  if (!err) {  res.render("pemain/list", {  list: docs  });  }  else {  console.log('Error in retrieving pemain list :' + err);  }  });  });  function handleValidationError(err,body){  for(field in err.errors)  {  switch(err.errors[field].path) {  case 'fullName':  body['fullNameError'] = err.errors[field].message;  break;  //case 'email':  // body['emailError'] = err.errors[field].message;  // break;  default:  break;  }  }  }  router.get('/:id', (req, res) => {  Pemain.findById(req.params.id, (err, doc) => {  if (!err) {  res.render("pemain/addOrEdit", {  viewTitle: "Update Pemain Bola",  pemain: doc  });  }  });  });  router.get('/delete/:id', (req, res) => {  Pemain.findOneAndRemove(req.params.id, (err, doc) => {  if(!err){  res.redirect('/pemain/list');  }  else {  console.log('Error in pemain delete:' + err);  }  });  });  module.exports = router; |

* **pemain.models.js**

|  |
| --- |
| const mongoose = require('mongoose');  var pemainSchema = new mongoose.Schema({  fullName:{  type: String,  required: 'This field is required.'  },  negara:{  type: String  },  klub:{  type: String  },  nomer:{  type: String  }  });  mongoose.model('Pemain', pemainSchema); |

* **mainLayout.hbs**

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <title>Pemain Sepak Bola</title>  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css" integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO"  crossorigin="anonymous">  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css">  </head>  <body class="bg-secondary">  <div class="row">  <div class="col-md-6 offset-md-3" style="background-color: #fff;margin-top: 25px;padding:20px;">  {{{body}}}  </div>  </div>  </body>  </html> |

* **addOrEdit.hbs**

|  |
| --- |
| <h3>{{viewTitle}}</h3>  <div class="section grey lighten-2">  <div class="row container">  <form action="/pemain" method="POST" autocomplete="off">  <input type="hidden" name="\_id" value="{{pemain.\_id}}">  <div class="form-group">  <label>Full Name</label>  <input type="text" class="form-control" name="fullName" placeholder="Full Name" value="{{pemain.fullName}}">  <div class="text-danger">  {{pemain.fullNameError}}</div>  </div>  <div class="form-group">  <label>Negara</label>  <input type="text" class="form-control" name="negara" placeholder="Negara" value="{{pemain.negara}}">  </div>  <div class="form-row">  <div class="form-group col-md-6">  <label>Klub</label>  <input type="text" class="form-control" name="klub" placeholder="Klub" value="{{pemain.klub}}">  </div>  <div class="form-group col-md-6">  <label>Nomer</label>  <input type="text" class="form-control" name="nomer" placeholder="Nomer" value="{{pemain.nomer}}">  </div>  </div>  <div class="form-group">  <button type="submit" class="btn btn-info"><i class="fa fa-database"></i> Submit</button>  <a class="btn btn-secondary" href="/pemain/list"><i class="fa fa-list-alt"></i> Lihat Daftar Pemain</a>  </div>  </form> |

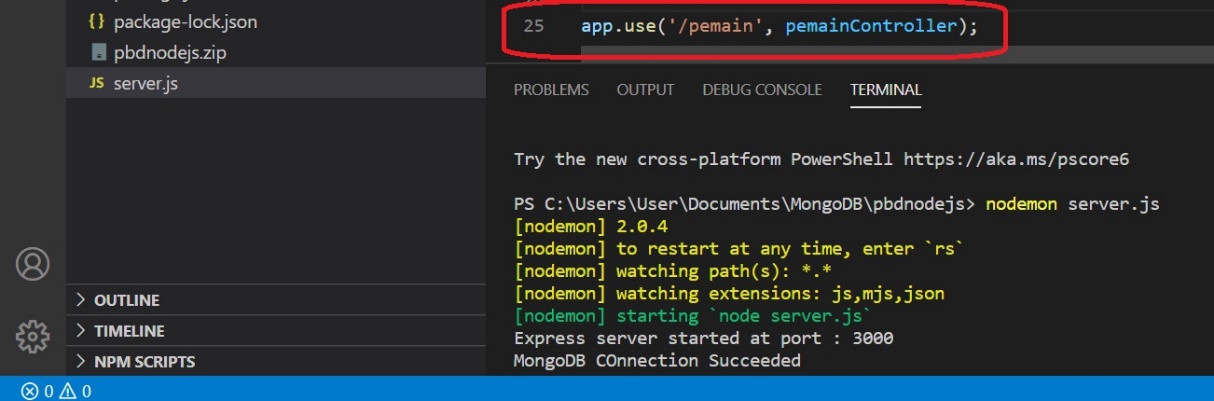
* **list.hbs**

|  |
| --- |
| <h3><a class="btn btn-secondary" href="/pemain"><i class="fa fa-plus"></i> Tambah Pemain</a> Daftar Pemain Bola</h3>  <table class="table table-striped">  <thead>  <tr>  <th>Full Name</th>  <th>Negara</th>  <th>Klub</th>  <th>Nomer</th>  <th></th>  </tr>  </thead>  <tbody>  {{#each list}}  <tr>  <td>{{this.fullName}}</td>  <td>{{this.negara}}</td>  <td>{{this.klub}}</td>  <td>{{this.nomer}}</td>  <td>  <a href="/pemain/{{this.\_id}}"><i style="color: green;" aria-hidden="true">Ubah</i></a>  <a href="/pemain/delete/{{this.\_id}}" onclick="return confirm('Hapus data pemain?');"><i style="color: red;" aria-hidden="true">Hapus</i></a>  </td>  </tr>  {{/each}}  </tbody>  </table> |

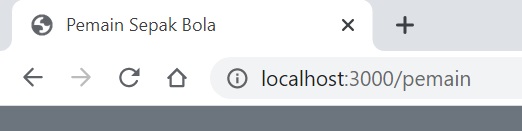
* **server.js**

|  |
| --- |
| require('./models/db');  const express = require('express');  const path = require('path');  const exphbs = require('express-handlebars');  const bodyparser = require('body-parser');  const pemainController = require('./controllers/pemainController');  var app = express();  app.use(bodyparser.urlencoded({  extended: true  }));  app.use(bodyparser.json());  app.set('views', path.join(\_\_dirname, '/views/'));  app.engine('hbs', exphbs({ extname: 'hbs', defaultLayout: 'mainLayout', layoutsDir: \_\_dirname + '/views/layouts/' }));  app.set('view engine', 'hbs');  app.listen(3000, () => {  console.log('Express server started at port : 3000');  });  app.use('/pemain', pemainController); |

1. Menjalankan project dengan masuk ke terminal “ctrl+shift+`” kemudian masukkan perintah “nodemon server.js” atau “node server.js” . Untuk nodemon perlu menginstall terlebih dahulu menggunakan perintah “npn i nodemon”.



1. Membuka project pada browser dengan menuliskan alamat “http://localhost:3000/pemain” , dimana “pemain” dari file server.js pada bagian app.use('/pemain', ....); .



# Hasil

