


<p style="text-align: center;">ปฏิบัติการที่ 1 :</p> <p style="text-align: center;">วิชา 523419 Advanced Web Application Development</p> <p>รหัสนักศึกษา..... ชื่อ.....</p> <p>วัตถุประสงค์ การติดตั้งและการใช้งาน Node.js Express Angular CLI MongoDB</p> <p>ไฟล์ที่จำเป็น Node.js, Express, MongoDB, MongoDB Compress, Angular</p>	<p>คะแนน</p>
--	---------------------

แบบฝึกปฏิบัติการที่ 1.1 การติดตั้งและการใช้งาน Node.js และ Express

1. ดาวน์โหลดและติดตั้งโปรแกรม Node.js จากเว็บไซต์นี้ <https://nodejs.org/en/download/>
2. หลังจากดาวน์โหลดไฟล์ติดตั้งมาแล้ว ให้ดับเบิลคลิก  **node-v12.16.1-x64** เพื่อติดตั้ง Node.js และทำตามขั้นตอนที่ปรากฏบนหน้าจอ
3. สร้างโฟลเดอร์สำหรับเก็บแอปพลิเคชัน ชื่อ “lab1” และเข้าไปยังโฟลเดอร์ที่สร้าง
4. สร้างโฟลเดอร์สำหรับเก็บโหนดแอปพลิเคชัน ชื่อ “lab1-express-app”
5. เปิดโปรแกรม Command Prompt ขึ้นมาและใช้คำสั่ง cd เพื่อเข้าไปยังโฟลเดอร์ “lab1-express-app”
6. สร้างไฟล์ package.json โดยกรอกคำสั่ง “npm init --yes” (โดย --yes หมายถึงต้องการใช้ค่าเริ่มต้น)
7. ทำการติดตั้ง Express โดยอ้างอิงจากเว็บไซต์ <https://www.npmjs.com/package/express> โดยไปที่หน้าต่างโปรแกรม Command Prompt กรอกคำสั่ง “npm install express”
8. สร้างแอปพลิเคชันออบเจกต์จาก Express โดยสร้างไฟล์ใหม่ ตั้งชื่อไฟล์ว่า “app.js” และกรอกโค้ดด้านล่างนี้ลงไป

```
// โหลดโมดูล express จากนั้นเก็บผลลัพธ์ลงในตัวแปร expressFunction
const expressFunction = require('express');

// เรียกฟังก์ชัน expressFunction() ซึ่งผลลัพธ์จะได้กลับมาเป็นออบเจกต์ แล้วนำไปเก็บยังตัวแปร express
const expressApp = expressFunction();

expressApp.use(function (req, res, next) {

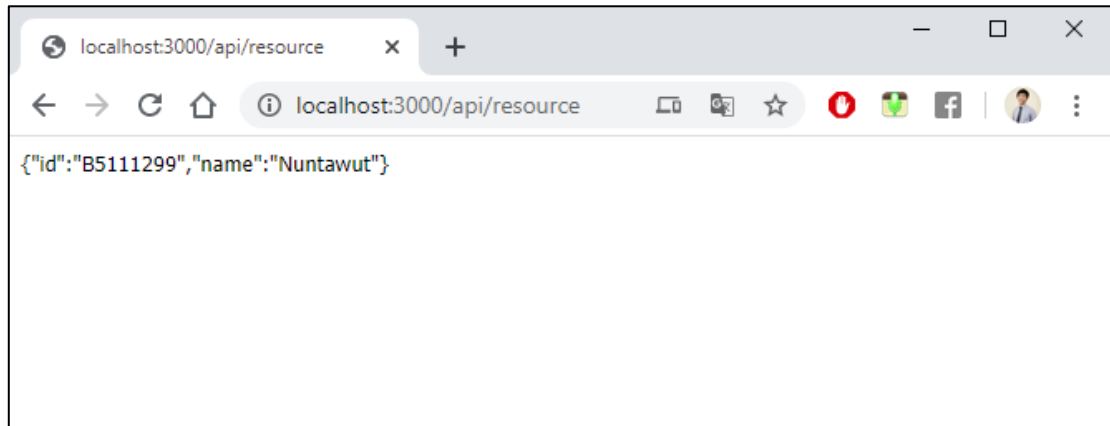
    // Website you wish to allow to connect
    res.setHeader('Access-Control-Allow-Origin', 'http://localhost:4200');

    // Pass to next layer of middleware
    next();
});

// เรียกใช้เมธอด get เพื่อตรวจสอบพารามิเตอร์ที่ส่งมาพร้อมกับ HTTP Request โดยกำหนด Endpoint
expressApp.get('/api/resource', function(req, res){
    const myJson = {id: 'B5111299', name: 'Nuntawut'};
    res.send(myJson);
});

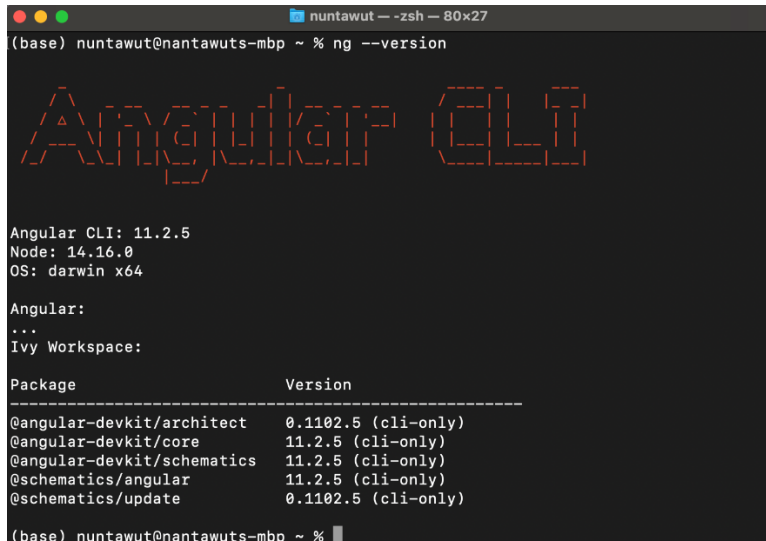
// สร้าง Event Listener รอการเชื่อมต่อผ่านจากพอร์ต 3000
expressApp.listen(3000, function(){
    console.log('Listening on port 3000');
});
```

- ทดสอบการทำงานของ Node.js และ Express ที่หน้าต่างโปรแกรม Command Prompt กรอกคำสั่ง “node app.js”
- เปิดเบราว์เซอร์ จากนั้นกรอก url คือ “http://localhost:3000/api/resource” จะพบข้อมูลในรูปแบบ JSON ปรากฏที่หน้าเว็บเพจ ดังรูปต่อไปนี้



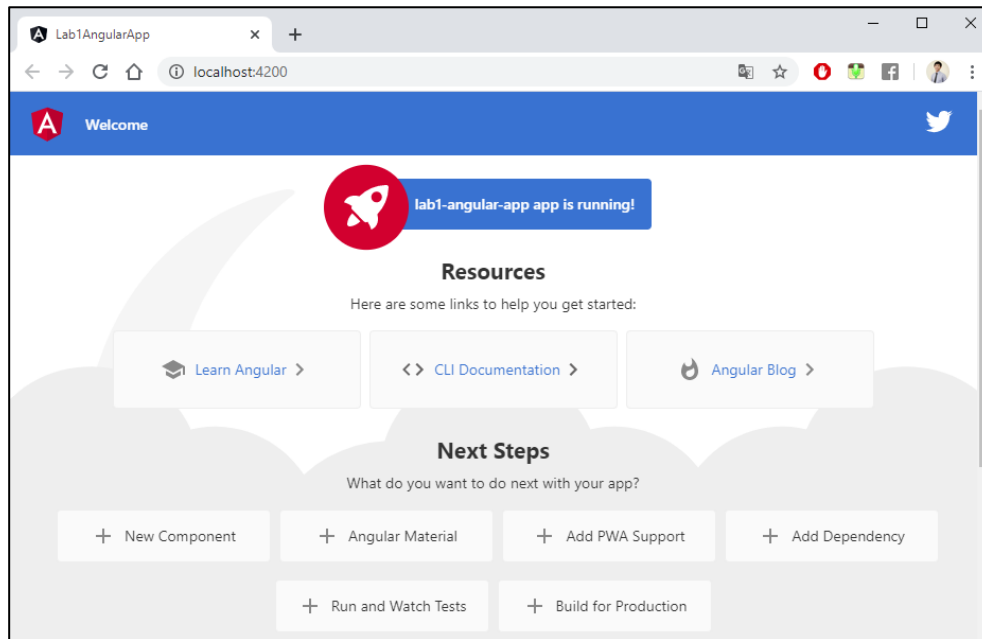
แบบฝึกปฏิบัติการที่ 1.2 การติดตั้งและการใช้งาน Angular CLI

- ติดตั้ง Angular CLI โดยเปิดโปรแกรม Command Prompt และพิมพ์คำสั่ง “npm install -g @angular/cli”
- ตรวจสอบเวอร์ชัน โดยพิมพ์คำสั่ง “ng --version” หากพบรายละเอียดของ Angular CLI แสดงว่าติดตั้งสำเร็จ ดังรูปต่อไปนี้



- เปิดโปรแกรม Command Prompt ขึ้นมาและใช้คำสั่ง cd เพื่อเข้าไปยังโฟลเดอร์ “lab1”
- สร้างแอปพลิเคชันใหม่ โดยกรอกคำสั่ง “ng new lab1-angular-app”
 - จะมีข้อความถามว่าจะใช้งาน Angular routing หรือไม่ ถ้าต้องการใช้ให้ตอบ Y
 - จะมีข้อความบอกให้เลือกปรับแต่งหน้าตาของแอปพลิเคชันให้เลือกเป็น CSS
- ใช้คำสั่ง cd เพื่อเข้าไปยังโฟลเดอร์ “lab1-angular-app”
- หน้าต่าง Command Prompt ทดสอบการทำงานของแอปพลิเคชันด้วยการพิมพ์คำสั่ง “ng serve --open”

7. เปิดเบราว์เซอร์ จากนั้นกรอก url คือ “http://localhost:4200” จะปรากฏที่หน้าเว็บเพจ ดังรูปต่อไปนี้



8. เข้าไปทำการแก้ไขไฟล์ lab1-angular-app/src/app/app.module.ts ตามโค้ดด้านล่างนี้

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { HttpClientModule } from '@angular/common/http';

import { AppComponent } from './app.component';

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    HttpClientModule,
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

10. เข้าไปทำการแก้ไขไฟล์ lab1-angular-app/src/app/ app.component.ts ตามโค้ดด้านล่างนี้

```
import { Component, OnInit } from '@angular/core';
import { HttpClient } from '@angular/common/http';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent implements OnInit {

  constructor(private http: HttpClient) { }

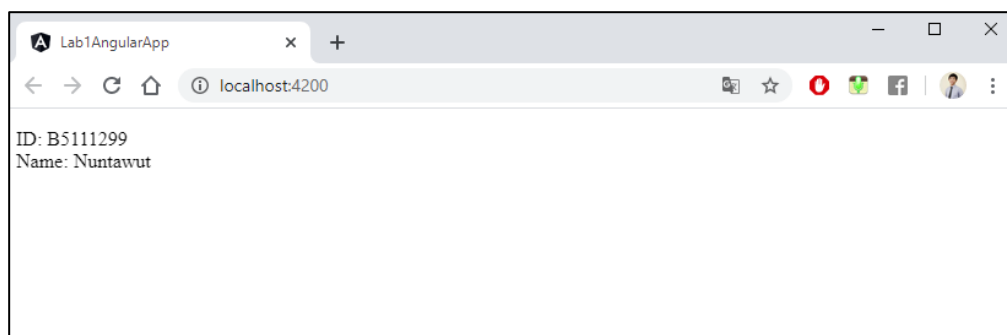
  id?: string;
  name?: string;

  ngOnInit() {
    this.http.get<any>('http://localhost:3000/api/resource').subscribe(data => {
      this.id = data.id;
      this.name = data.name;
    })
  }
}
```


11. เข้าไปทำการแก้ไขไฟล์ lab1-angular-app/src/app/ app.component.html ตามโค้ดด้านล่างนี้

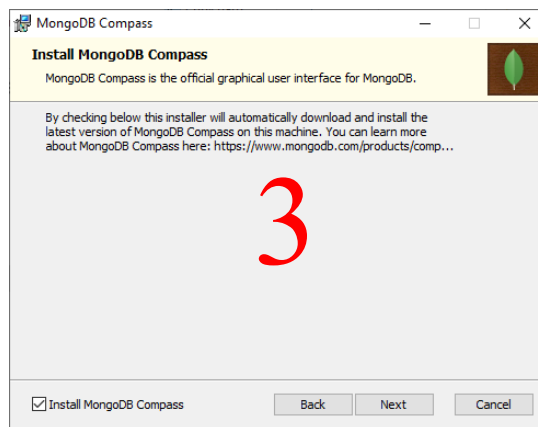
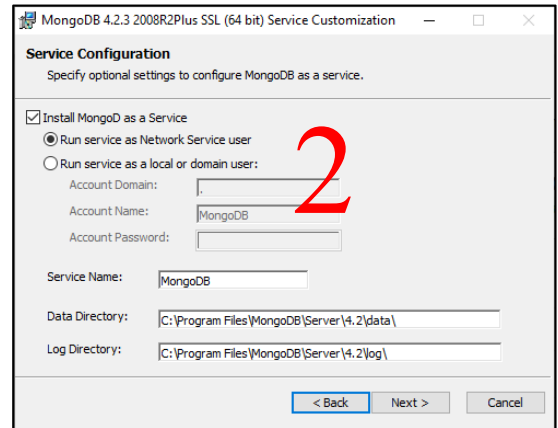
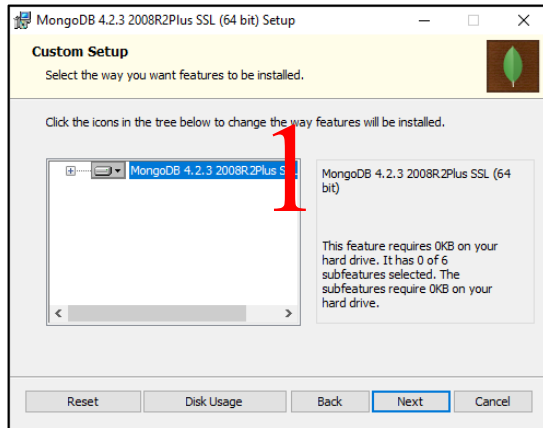
```
<p>
  ID: {{id}} <br/>
  Name: {{name}}
</p>
```

12. เปิดเบราว์เซอร์ จากนั้นกรอก url คือ “http://localhost:4200” จะปรากฏที่หน้าเว็บเพจ ดังรูปต่อไปนี้

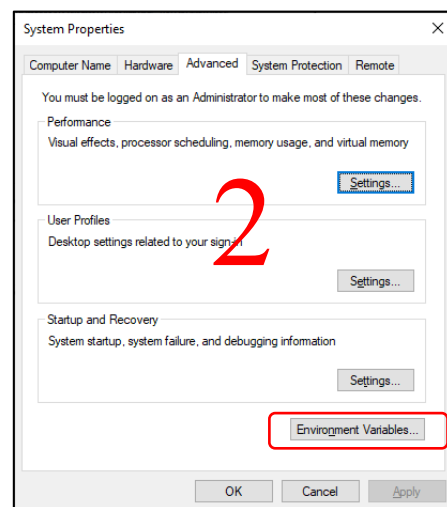
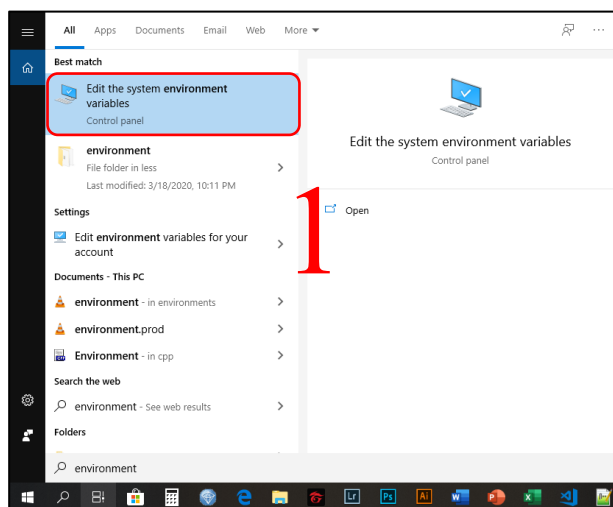


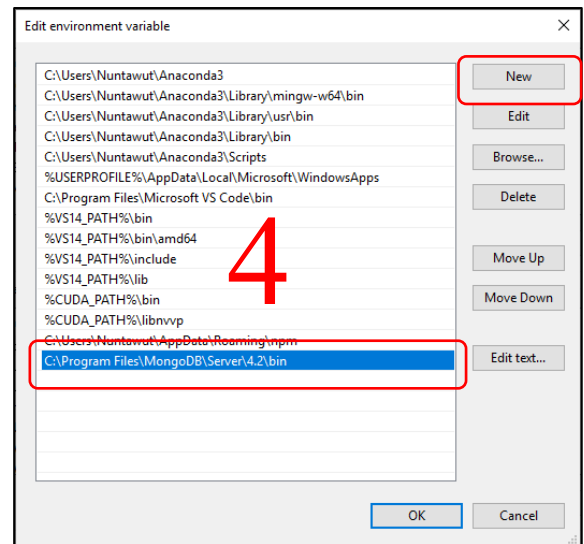
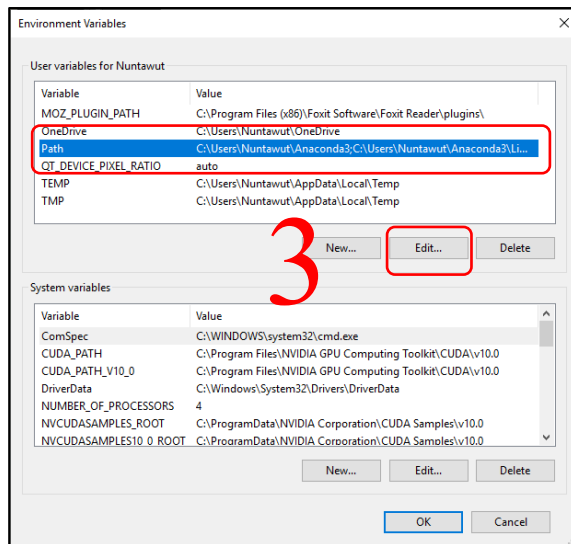
แบบฝึกปฏิบัติการที่ 1.3 การติดตั้ง MongoDB

1. ดาวน์โหลดและติดตั้งโปรแกรม MongoDB Community Server จากเว็บไซต์นี้
<https://www.mongodb.com/download-center/community>
2. หลังจากดาวน์โหลดไฟล์ติดตั้งมาแล้ว ให้ดับเบิลคลิก  **mongodb-win32-x86_64-2012plus-4.2.3-signed** เพื่อติดตั้ง MongoDB Community Server และทำตามขั้นตอนที่ปรากฏบนหน้าจอ โดยตั้งค่าดังรูปต่อไปนี้

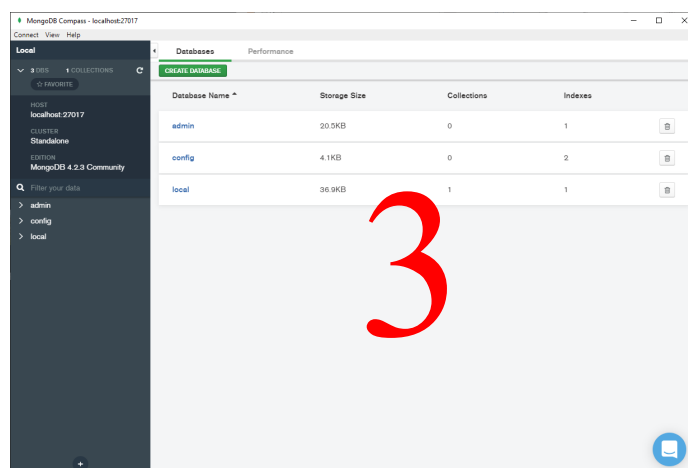
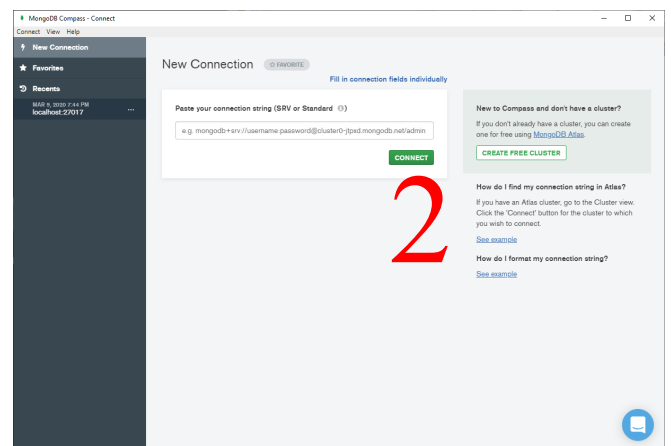
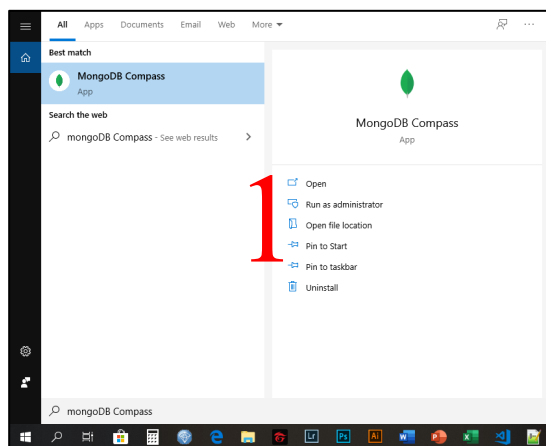


3. กำหนดพารสำหรับเรียกใช้ MongoDB

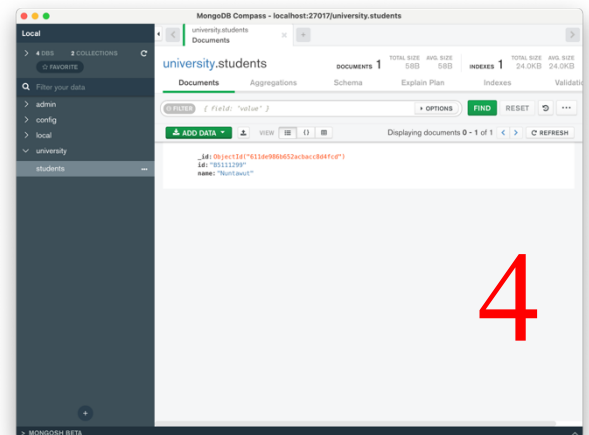
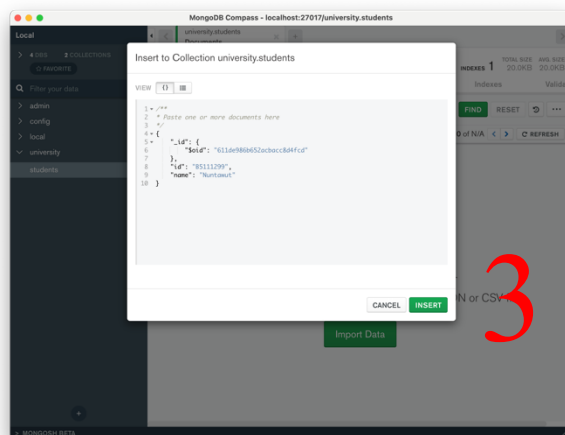
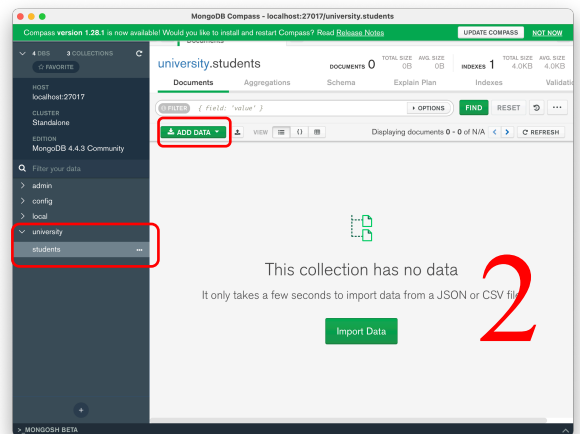
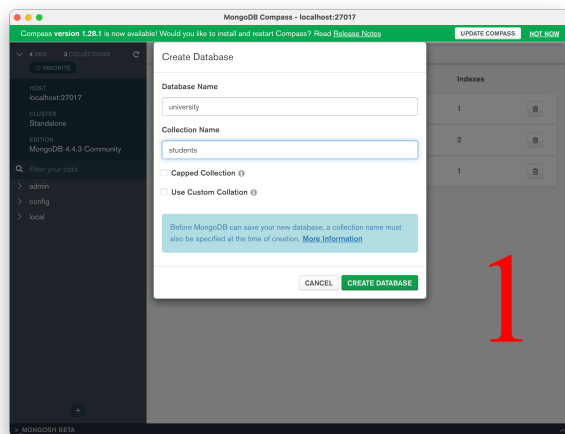




4. กำหนดโฟลเดอร์สำหรับใช้เก็บฐานข้อมูลของ MongoDB ในไดฟ์หลักของคอมพิวเตอร์ ซึ่งส่วนมากคือไดฟ์ C และสร้างโฟลเดอร์ตามรูปแบบดังนี้ C:/data/myDB
5. ปิดการใช้งาน MongoDB ในแบบ Windows Service ไปที่หน้าต่าง Command Prompt และพิมพ์คำสั่ง “net stop MongoDB”
6. ทดลองการทำงานของ MongoDB โดยเข้าไปที่โปรแกรม MongoDB Compass ที่ถูกติดตั้งมาพร้อมกับ MongoDB ตั้งแต่แรก และทำการคลิก Connect ทดสอบการเชื่อมต่อ ดังรูปต่อไปนี้



7. ให้ นศ. ทำการสร้างฐานข้อมูลโดยไปที่ **CREATE DATABASE** ชื่อ “university” และกำหนด collection ชื่อ “students” ผลลัพธ์ดังรูปต่อไปนี้



แบบฝึกปฏิบัติการที่ 1.4 นำข้อมูลจาก MongoDB มาแสดงที่ Angular

1. เปิดโปรแกรม Command Prompt ขึ้นมาและใช้คำสั่ง `cd` เพื่อเข้าไปยังโฟลเดอร์ “lab1-express-app”
2. ทำการติดตั้ง mongoose โดยอ้างอิงจากเว็บไซต์ <https://www.npmjs.com/package/mongoose> โดยไปที่หน้าต่างโปรแกรม Command Prompt กรอกคำสั่ง “`npm install mongoose`”
3. เข้าไปทำการแก้ไขไฟล์ lab1-express-app /app.js ตามโค้ดด้านล่างนี้

//โหลดโมดูล express จากนั้นเก็บผลลัพธ์ลงในตัวแปร expressFunction

```
const expressFunction = require('express');
const mongoose = require('mongoose');
const url = 'mongodb://localhost:27017/university';
const config = {
  autoIndex: true,
  useNewUrlParser: true,
  useUnifiedTopology: true
};
var Schema = require("mongoose").Schema;
const userSchema = Schema({
  stdid: String,
  name: String,
}, {
  collection: 'students'
});

let Students
try {
  Students = mongoose.model('students')
} catch (error) {
  Students = mongoose.model('students', userSchema);
}
```

//เรียกฟังก์ชัน expressFunction() ซึ่งผลลัพธ์จะได้กลับมาเป็นออบเจกต์ แล้วนำไปเก็บยังตัวแปร express

```
const expressApp = expressFunction();
expressApp.use(function (req, res, next) {
  // Website you wish to allow to connect
  res.setHeader('Access-Control-Allow-Origin', 'http://localhost:4200');
  // Pass to next layer of middleware
  next();
});
expressApp.use(expressFunction.json());
expressApp.use((req, res, next) =>{
  mongoose.connect(url, config)
    .then(() => {
      console.log('Connected to MongoDB...');
      next();
    })
    .catch(err =>{
      console.log('Cannot connect to MongoDB');
      res.status(501).send('Cannot connect to MongoDB')
    });
})
```

//เรียกใช้เมธอด get เพื่อตรวจสอบพารามิเตอร์ที่ส่งมาพร้อมกับ HTTP Request โดยกำหนด Endpoint

```
expressApp.get('/api/resource', function(req, res){
  Students.findOne({id:"B5111299"},(err, data) => {
    res.send(data)
  });
});
```

//สร้าง Event Listener รอการเชื่อมต่อผ่านจากพอร์ต 3000

```
expressApp.listen(3000, function(){
  console.log('Listening on port 3000');
});
```


4. เปิดเบราว์เซอร์ จากนั้นกรอก url คือ “http://localhost:4200” จะปรากฏที่หน้าเว็บเพจ ดังรูปต่อไปนี

