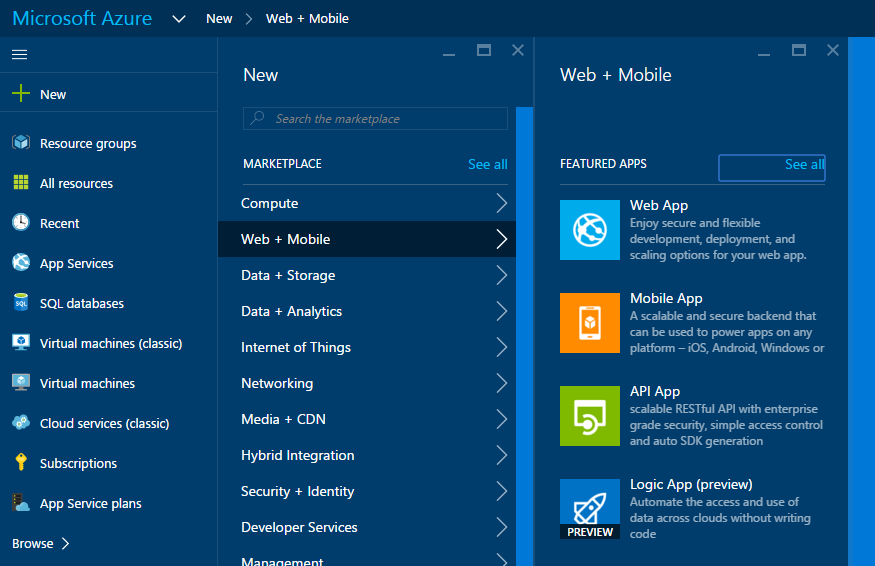
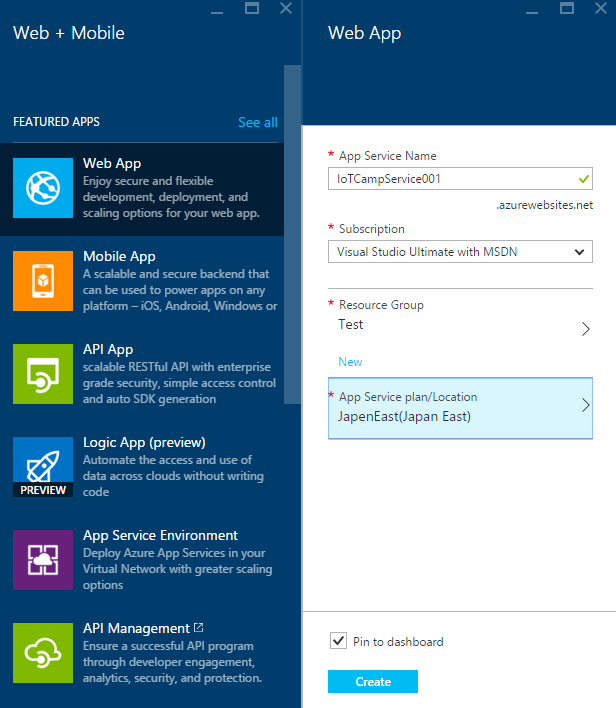
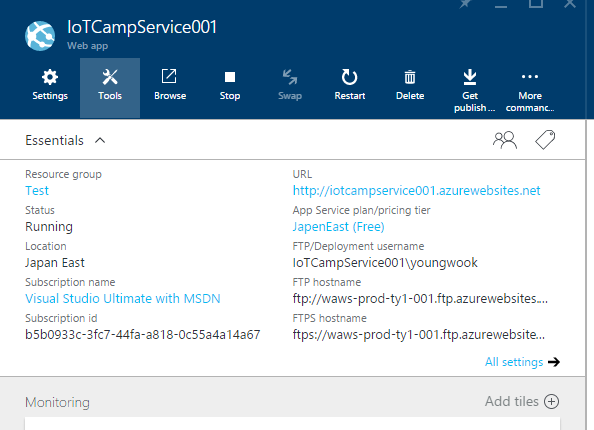


**IoT Service development   
(Azure Website, PHP, MySQL)**

1. Azure Web App 설정하기   
   <https://portal.azure.com>

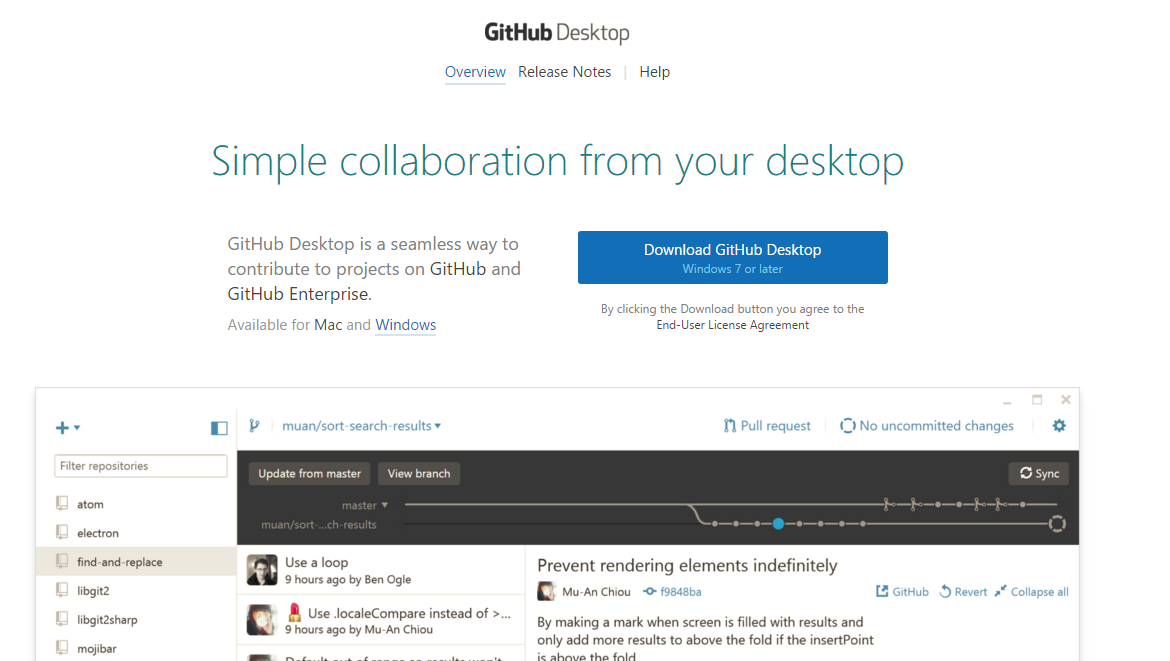


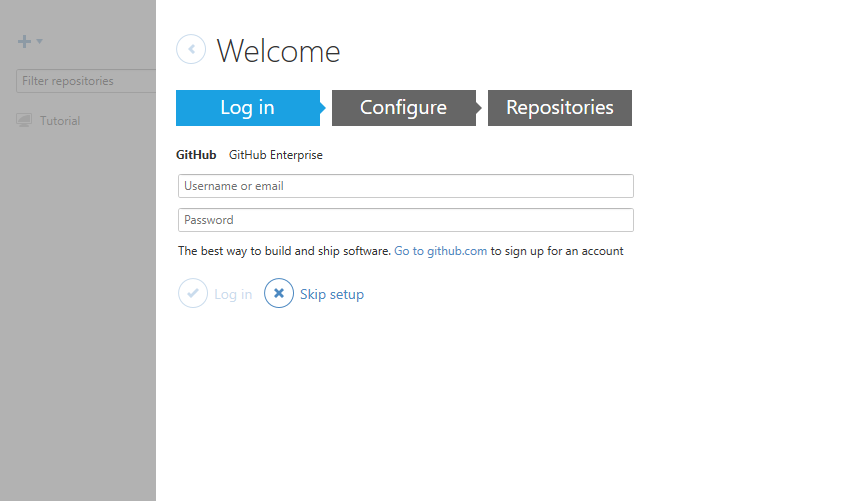
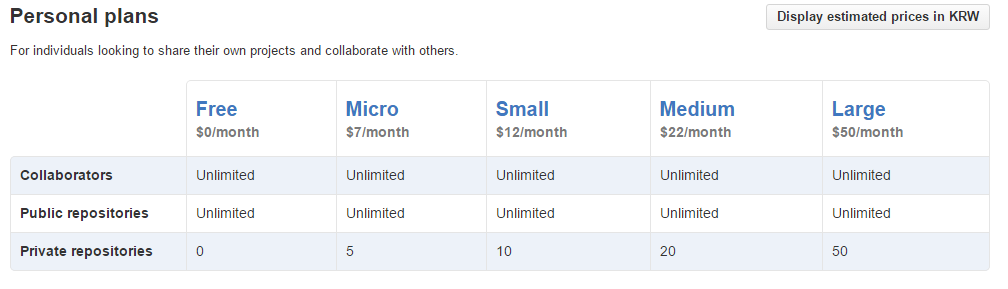


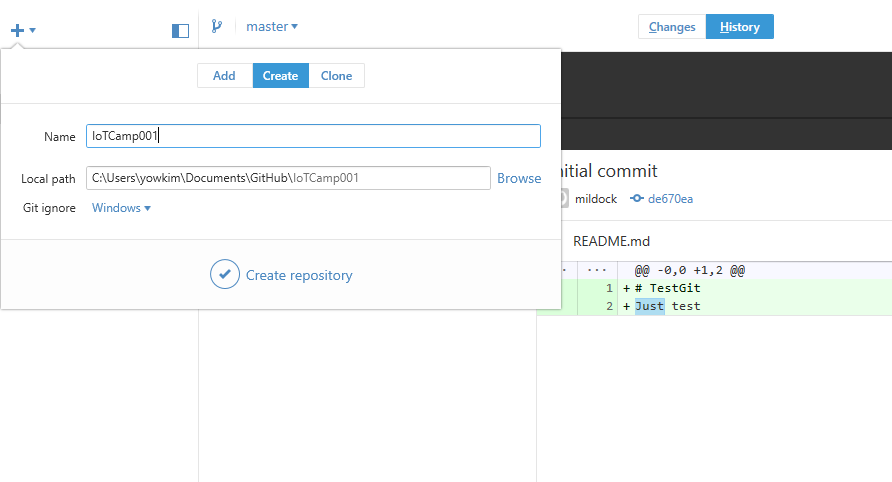
지금 현재 JapenEast가 한국에서는 가장 빠른 속도를 보여준다. 그래서 가급적이면 JapenEast를 사용한다.   


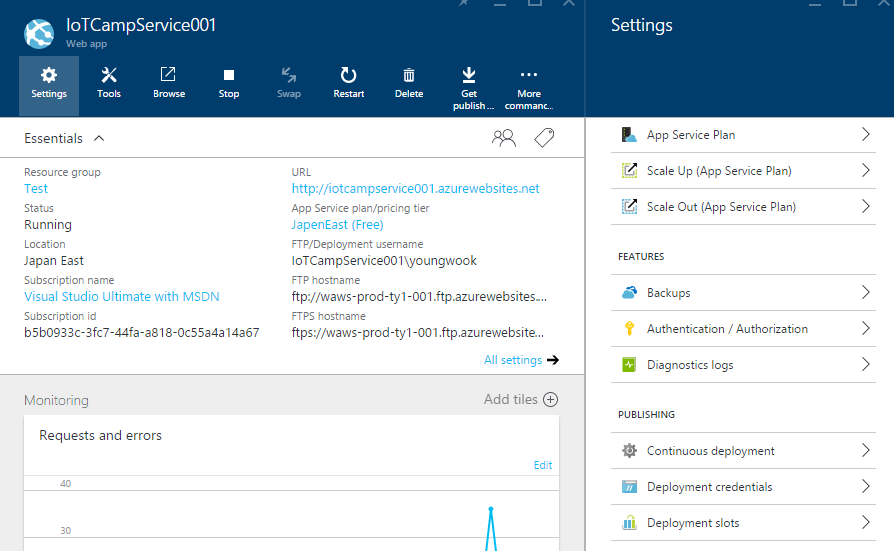
웹 사이트가 생성되고 나면 접근 가능한 URL이 보이고 바로 열어 볼 수 있다.

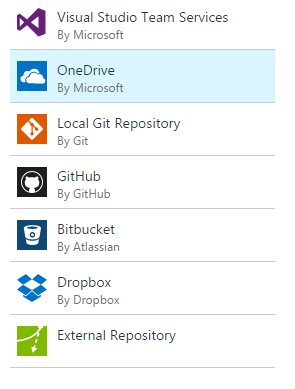
<Option>Visual Studio Online 체험해보기

1. GitHub 설정하기  
   GitHub는 소프트웨어의 형상관리를 위한 도구로 지금 현재 가장 널리 사용되고 있다. Git을 명령어로 사용할 수도 있지만 GitHub Desktop을 사용하면 GUI 기반으로 편리하게 사용할 수 있다.   
    먼저 GitHub Desktop을 다운로드 한다.   
   <https://desktop.github.com/>   
   

GitHub Desktop에 로그인한다. 이때 계정이 없으면 계정을 생성한다.   
  
  
GitHub는 Private 한 레포지터리에 따라서 과금된다.   


새로운 레포지터리를 생성한다.   
  
여기까지 했다면 C:\Users\yowkim\Documents\GitHub\IoTCamp001 경로가 GitHub와 연결된 폴더가 되었다.

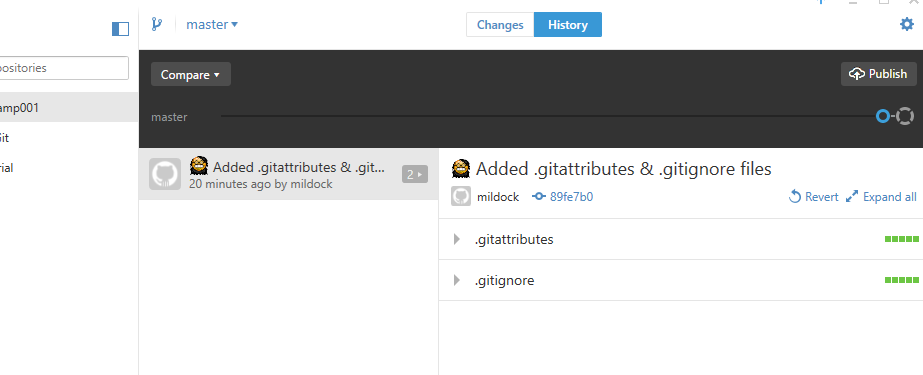
1. Azure Web App과 GitHub 연결하기  
     
   Azure Web App과 GitHub를 연결하면 GitHub에 업데이트함과 동시에 Azure Web App 소스코드를 배포할 수 있다. 이렇게 설정해 두면 소스코드의 형상관리와 배포가 한꺼번에 이루어 지기 때문에 편리한 웹 개발 환경이 구성된다.  
    Azure Web App의 Setting 🡪 Continuous deployment 를 차례데로 선택한다.



Azure Web App은 Visual Studio Team Services, OneDrive, Local Git Repository, GitHub, Bitbucket, Dropbox, External Repository등 다양한 서비스와 연결이 가능하지만 여기서는 GitHub를 선택한다.

여기서 GitHub 계정과 프로젝트를 선택하고 나면 연결이 끝난다.

그런데 방금 만든 프로젝트가 보이지 않는데 그 이유는 아직 GitHub Desktop의 내용이 아직 GitHub에 반영되지 않았기 때문이다.

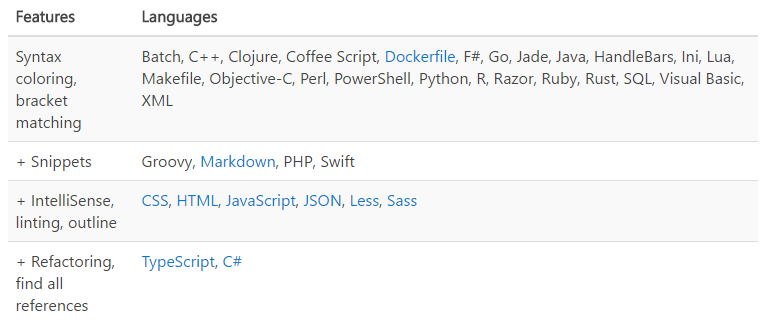


다시 GitHub Desktop으로 가서 Publish 버튼을 눌러서 배포하고 나면 해당 프로젝트가 보이게 된다.

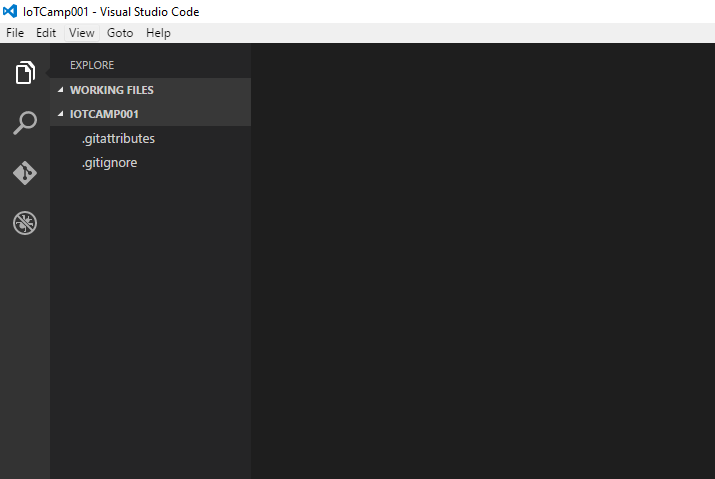
1. Visual Studio Code 설치하기

개발환경으로 Visual Studio는 가장 이상적인 개발 환경을 제공해 주지만 더 가벼운 개발환경을 원하는 경우 Visual Studio Code는 좋은 대안이 될 수 있다.

Visual Studio Code는 Microsoft에서 Open Source로 제공되는 무료 개발 도구이며 Windows뿐만 아니라 Mac, Linux까지 지원된다. 또 지금 인기있는 대부분의 개발언어를 지원한다.

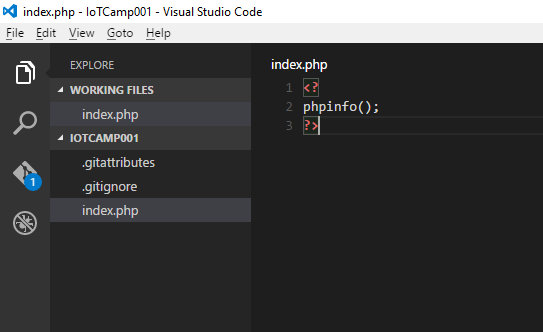
  
 먼저 Visual Studio Code를 <https://www.visualstudio.com> 에서 다운로드 해서 설치한다.

Visual Studio Code가 설치되고 나면 File🡪 Open Folder를 선택해서 GitHub와 연결해 놓은 C:\Users\yowkim\Documents\GitHub\IoTCamp001을 선택한다.

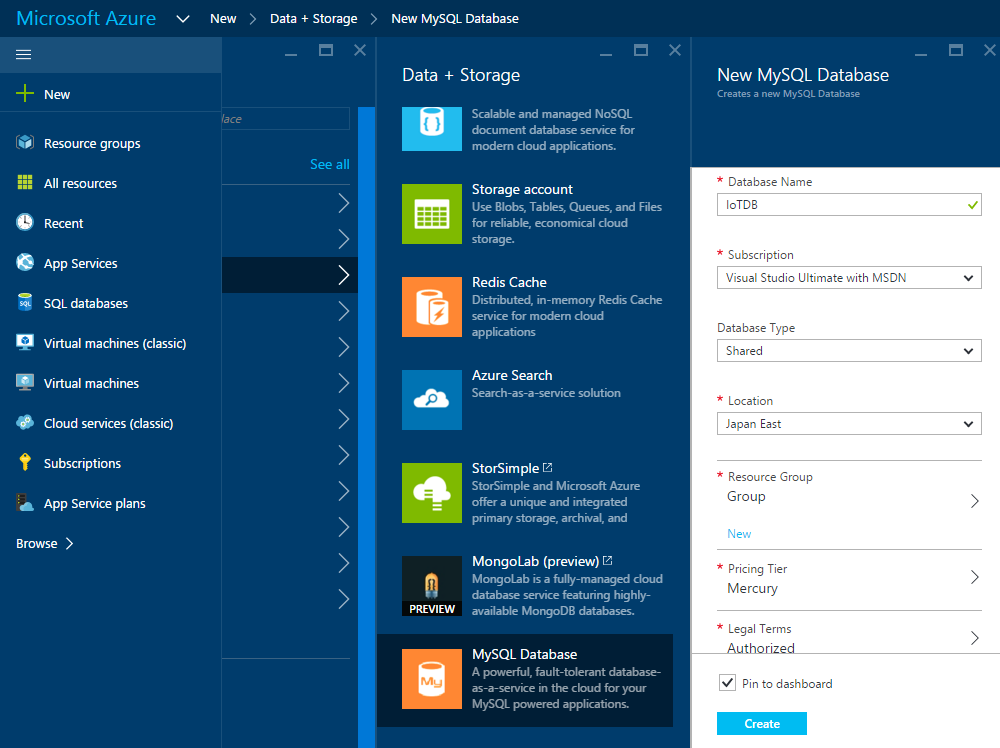


PHP Extension 을 설치하기 위해서 F1을 누른후 ext inst를 입력해서 PHP Code Formatter를 설치한다.

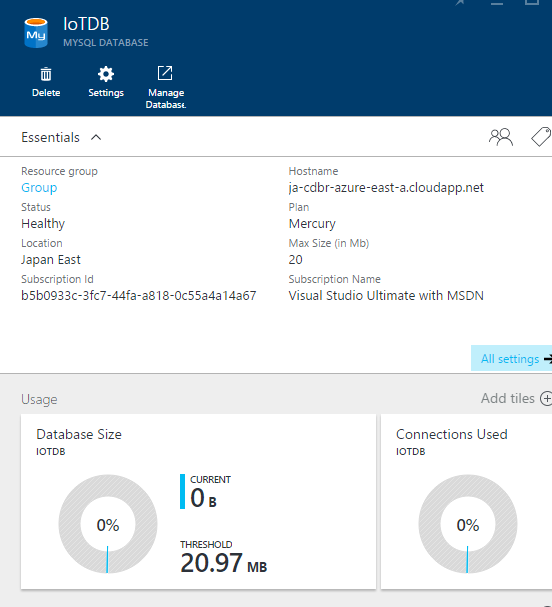
Index.php를 만들고 아래와 같이 코드를 생성하고 나면 Git의 변경사항이 기록된다.



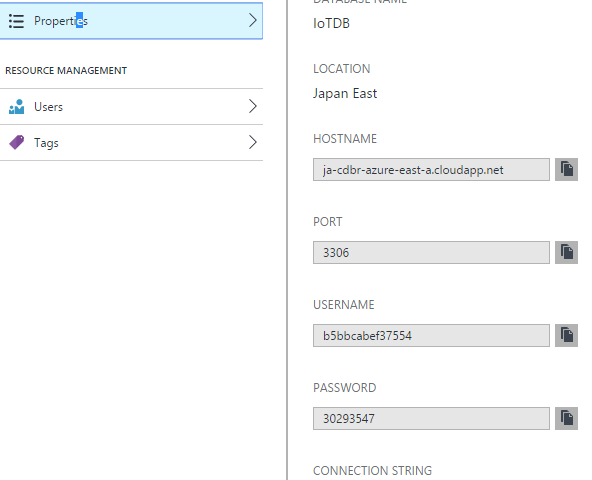
1. MySQL Database 만들기  
   MySQL도 Azure에서 정식으로 서비스 되고 있다. New 🡪 Data + Storage 🡪 MySQL을 차례대로 선택해서 MySQL을 생성할 수 있다. MySQL과 Web App은 같은 데이터 센터에 있어야 한다. 그렇지 않을 경우 데이터 센터간의 트레픽은 과금된다.



Pricing Tier 를 Mercury를 선택하면 무료로 사용할 수 있다. 용량은 작지만 테스트용으로는 적합하다.

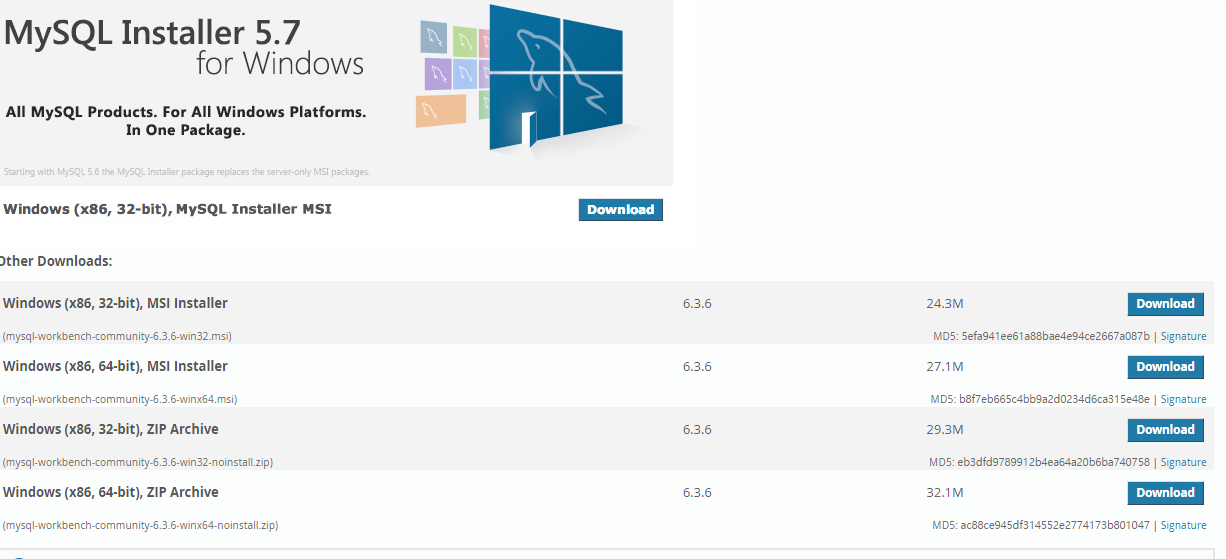


Setting 에 들어가 보면 MySQL의 접속 정보를 확인 할 수 있다.



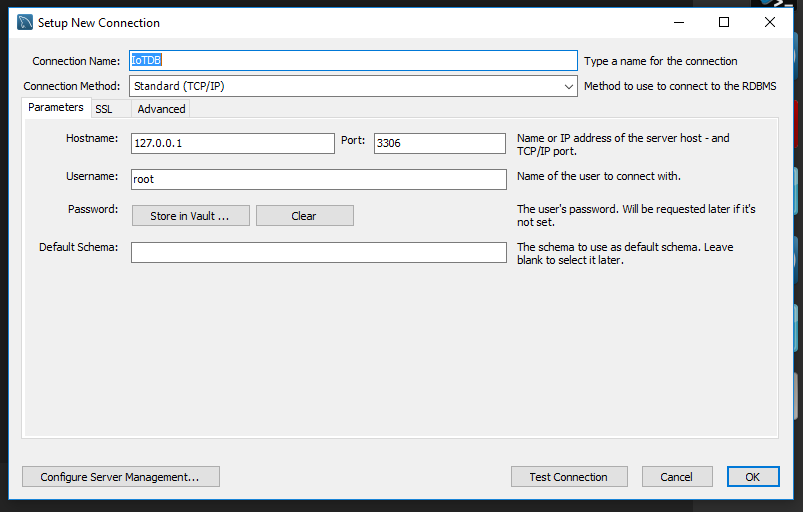
Setting 안에는 Hostname, Port, UserName, Password 등이 포함되어 있다.

MySQL을 관리하기 위해서 관리 도구를 설치한다. MySQL의 관리도구는 여러가지가 있지만 MySQL WorkBench를 사용한다. 먼저 아래 링크에 접속해서 다운로드 받아 설치한다.

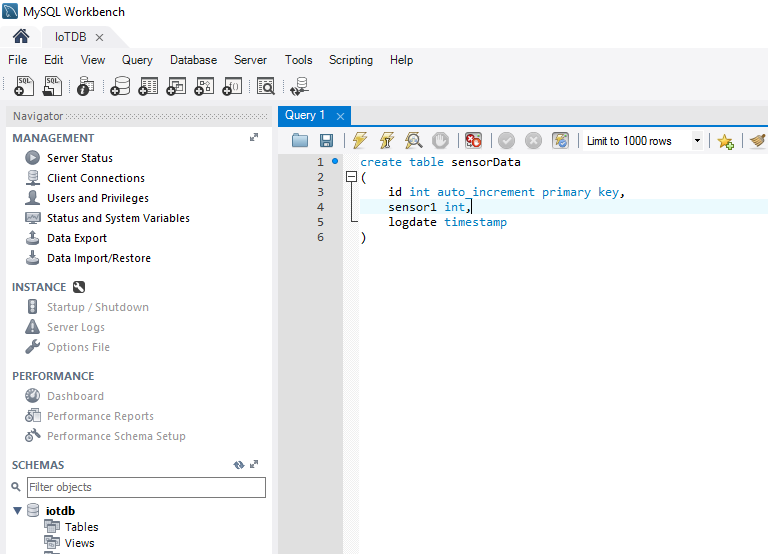
<https://dev.mysql.com/downloads/workbench/>   


설치가 완료되면 로그인 정보를 입력해서 접속한다.

이때 Default Schema를 설정해 두면 편리하다.



연결이 완료되면 Database Table을 생성한다.



create table sensorData

(

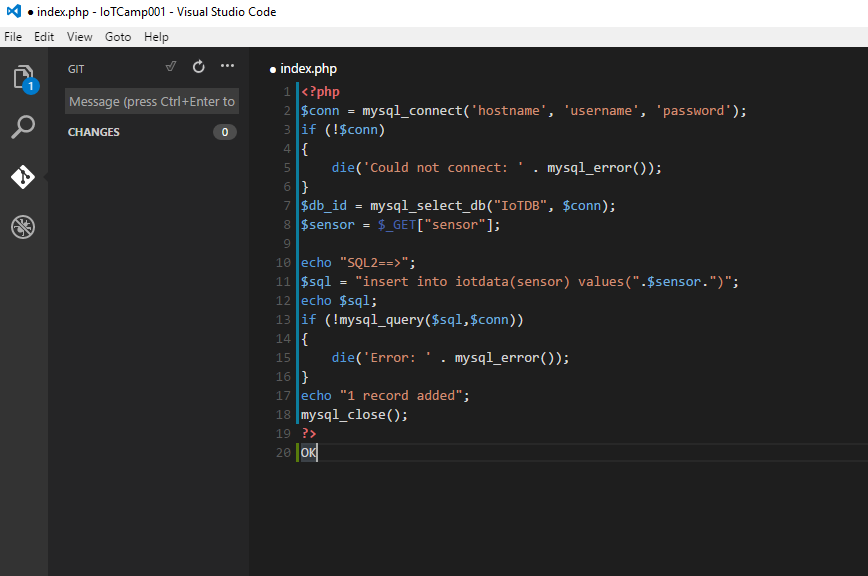
id int auto\_increment primary key,

sensor1 int,

logdate timestamp

)

PHP 코드를 입력해서 서비스를 완성한다.



<?php

$conn = mysql\_connect('hostname', 'username', 'password');

if (!$conn)

{

die('Could not connect: ' . mysql\_error());

}

$db\_id = mysql\_select\_db("IoTDB", $conn);

$sensor = $\_GET["sensor"];

$sql = "insert into sensorData(sensor1) values(".$sensor.")";

if (!mysql\_query($sql,$conn))

{

die('Error: ' . mysql\_error());

}

mysql\_close();

?>

OK

코드를 변경한 후에는 다시 한번 GitHub와 Sync를 해주고 나면   
<http://iotcampservice001.azurewebsites.net/index.php?sensor=50> 와 같이 주소를 입력해서 정상적으로 동작되는지 테스트 해본다.

또 MySQL에서 select \* from sensorData 를 입력해서 제대로 입력되었는지 확인해 본다.

Option

HttpClient client = new HttpClient();

HttpResponseMessage response = await client.GetAsync("http://iotcampservice001.azurewebsites.net/index.php?sensor=" + result.ToString());

Power BI에서 연결이 안될 때

<http://dev.mysql.com/downloads/file/?id=412152>

