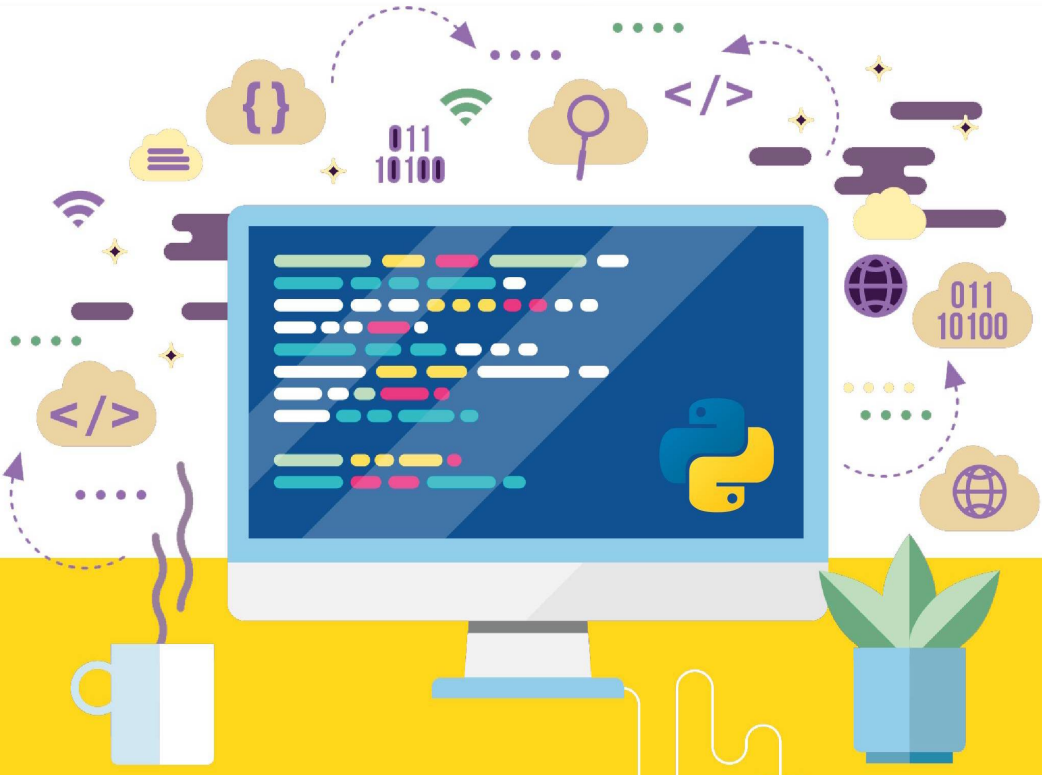
Python and MySQL
for Beginner



FATIMAH RAHMAT | MOHAMAD IQBAL HAKIM CHE OMAR | NURUL SHAKIRAH MOHD ZAWAWI

Python and MySQL
for Beginner

FATIMAH RAHMAT
MOHAMAD IQBAL HAKIM CHE OMAR
NURUL SHAKIRAH MOHD ZAWAWI

First Edition 2023

Politeknik Mersing
Jalan Nitar,
86800 Mersing
Johor Darul Ta'zim
Telephone : 07-7980001
Fax : 07-7980002
Website : https://pmj.mypolycc.edu.my/

Printed in Malaysia
First Printing, 2023
eISBN : 978-967-2904-57-1

Authors :
Fatimah Rahmat
Mohamad Iqbal Hakim Che Omar
Nurul Shakirah Mohd Zawawi

PREFACE

The need to share knowledge and experience on how the Python programming language could have a positive impact on the learning process led to the creation of this book focuses on the installation process, use of MySQL for database connections, and fundamental Python language concepts.

This book offers helpful resources, advice, and examples, especially for students and anybody else interested in learning the Python programming language.

The Create, Read, Update, and Delete (CRUD) component of an application employing Python scripts using MySQL is also demonstrated in this book's sample examples.

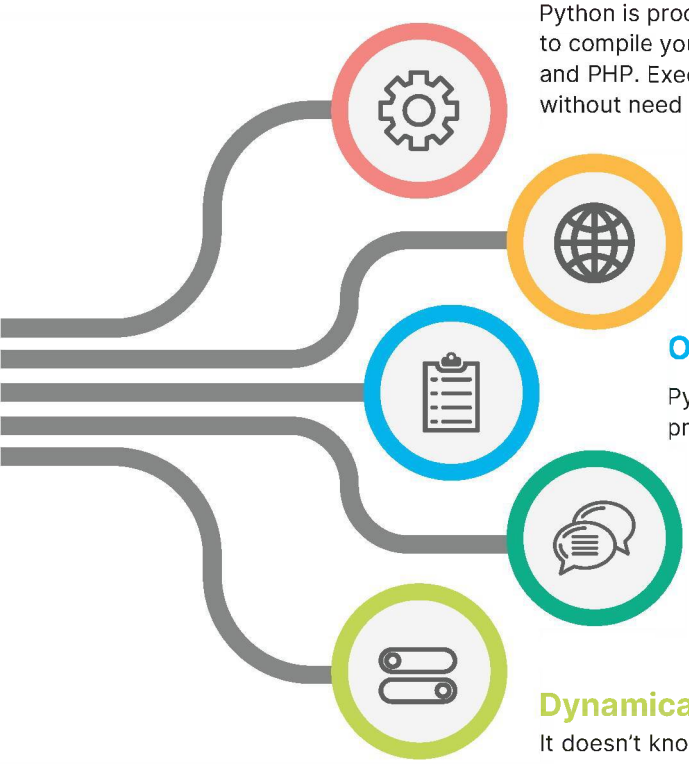
This book is intended to assist readers in achieving their goals.

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

CHAPTER 1: INTRODUCTION

What is Python Programming?



Interpreted

Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP. Execution is implemented directly and freely from source without need to be compiled into machine code or binary format.

Interactive

You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Object-Oriented

Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

Beginner's Language

Python is a great language for the beginner- level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

Dynamically typed language

It doesn't know about the type of the variable until the code is run. So declaration is of no use. What it does is, It stores that value at some memory location and then binds that variable name to that memory container.

Python is a **high-level programming language** that was first released in 1991 by **Guido van Rossum**. The language was designed to be easy to read and write, with a **focus on code readability**. Python's design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than languages like C++ or Java.

Basic Principles of Python

Basic Core language

Python is designed so that there really isn't that much to learn in the basic language. For example, there is only one basic structure for conditional programming (if/else/elif), two looping commands (while and for), and a consistent method of handling errors (try/except) which apply to all python programs.

Modules

Self-contained programs which define a variety of functions and data types that you can call in order to do tasks beyond the scope of the basic core language by using the import command.

Object oriented programming

A basic concept of object oriented programming is encapsulation, the ability to define an object that contains your data and all the information a program needs to operate on that data. In this way, when you call a function (known as a method in object-oriented lingo), you don't need to specify a lot of details about your data, because your data object ``knows'' all about itself. In addition, objects can inherit from other objects, so if you or someone else has designed an object that's very close to one you're interested in, you only have to construct those methods which differ from the existing object, allowing you to save a lot of work.

Namespace and variable scoping

The same variable name can be used in different parts of a program without fear of destroying the value of a variable you're not concerned with.

Exception handling

When you're performing an operation that might result in an error, you can surround it with a try loop, and provide an exception clause to tell python what to do when a particular error arises.

Some of the most popular Python **libraries and frameworks** include:

- NumPy, a library for numerical computing
- pandas, a library for data manipulation and analysis
- scikit-learn, a machine learning library
- TensorFlow, a library for deep learning
- Django, a web framework
- Flask, a micro web framework

CHAPTER 2
REQUIREMENTS

CHAPTER 2: REQUIREMENTS

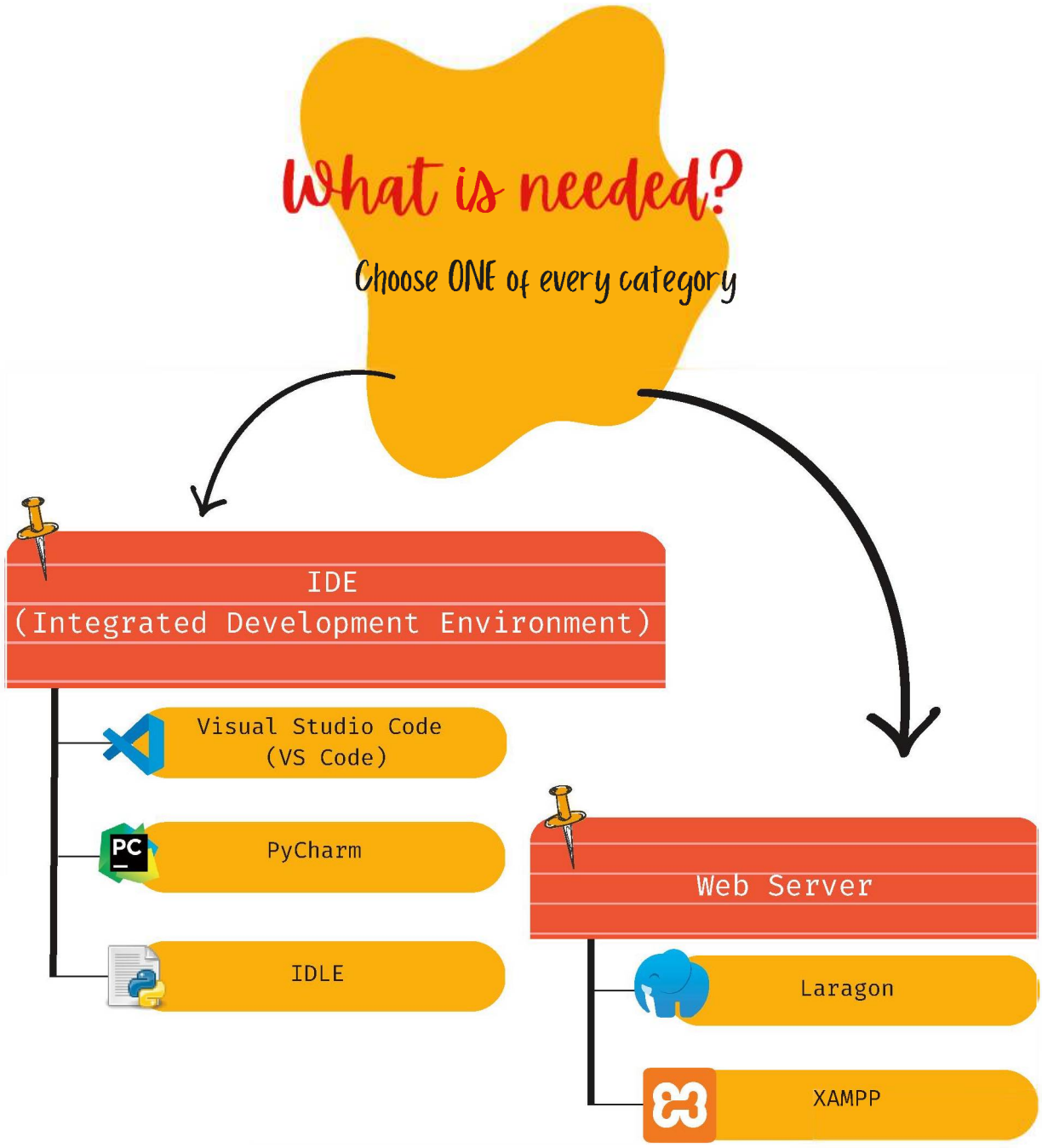
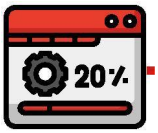


Table 1: Visual Studio Code versus PyCharm versus IDLE

CRITERIA	VS Code	PyCharm	IDLE
Environment	An IDE (Integrated Development Environment).	Code editor that provides a similar experience to an IDE through extensions.	An IDLE Integrated Development and Learning Environment
Performances	Lightweight because it does not need much space.	Consumes a lot of resources because it requires a lot of memory and large storage space.	Coded in 100% pure Python, using the tkinter GUI toolkit
Platform	Free and compatible with all platforms: Windows, Linux and Mac	Cross platform IDE. More powerful and has a commercial version.	Cross-platform: works mostly the same on Windows, Unix, and macOS
Features	Potential errors are automatically highlighted in red, making it easy to find and fix errors. It goes a step further by adding the Issues tab, listing all potential bugs in one place and making it easy to review. To use Python with VS Code, you need to install the Python formatter and linter.	Main features is the "Search Anywhere" that allows you to search outside your project. Can find files, classes, symbols, and user interface elements even if they don't exist in your current project. Code completion is better with PyCharm. It displays function signatures as part of the autocomplete selection list. Has some additional features such as Sort by Name, Quick Documentation and Quick Definition. Quick documentation shows the function's signature and return type, as well as the function's comments while quick definition shows the feature code and is convenient.	Python shell window (interactive interpreter) with colorizing of code input, output, and error messages. Multi-window text editor with multiple undo, colorizing, smart indent, call tips, auto completion, and other features. Search within any window, replace within editor windows, and search through multiple files (grep). Debugger with persistent breakpoints, stepping, and viewing of global and local namespaces Configuration, browsers, and other dialogs

CRITERIA	VS Code	PyCharm	IDLE
Extensions	Need some extensions to make your code editor like an IDE that fits well into Python. Detects the type of project you're working on and then suggests and embeds the necessary extensions for that project.	Built with Python in mind, and available extensions aimed at improving PyCharm. More than 3000 JetBrains plugins are available and compatible with all of them.	Can be extended with additional libraries and tools through the use of pip, the package manager for Python.
Database Integration	Database integration available on VS Code with the extension called SQLTools. A beginner might find it difficult to use or navigate, and it can be susceptible to bugs.	Uses a plugin called Database Navigator that allows connecting to databases like MySQL, Oracle, PostgreSQL, and others, all within the app. Other than that, you can create a database connection, issue queries to a database, receive database objects, and more. However, this is only available in the professional version, and must be purchased.	Need to install a database driver for the specific database you are using (such as psycopg2 for PostgreSQL or mysql-connector-python for MySQL). Once you have the driver installed, you can use the appropriate library to connect to the database and perform operations such as querying and updating data. You can also use an ORM (Object-Relational Mapping) library such as SQLAlchemy to simplify the process of interacting with the database.

Minimum requirements for Visual Studio Code (VS Code) installation



To **install and run** Visual Studio Code (VS Code) for Python development, **your system should meet the following minimum requirements**:

1. Operating System: Windows 7 or later, macOS 10.10 or later, or Linux (64-bit)
2. Processor: 1.6GHz or faster, 2-core
3. Memory: 2 GB or more
4. Hard Disk Space: At least 1 GB of free space
5. Display: 1024x768 resolution or higher
6. Internet connection: Required for installation and updates
7. Administrator rights: Installation of VS Code requires administrator rights on Windows and macOS.
8. Python: To use VS Code for Python development, you will need to have Python installed on your system. You can download the latest version of Python from the official website.
9. Python Extension for Visual Studio Code: In order to use VS Code for Python development, you will need to install the Python extension for VS Code. This extension provides rich support for Python development, including IntelliSense, debugging, and linting.
10. A Python environment: Python can be used with virtual environments like Anaconda, venv, virtualenv and pipenv.

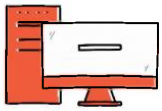
Minimum requirements for PyCharm installation

PyCharm is a popular and widely used Integrated Development Environment (IDE) for Python development developed by JetBrains. To **install and run** PyCharm, **your system should meet the following minimum requirements**:

1. Operating System: Windows, macOS, or Linux
2. Processor: 2 GHz or faster
3. Memory: 2 GB of RAM or more
4. Hard Disk Space: At least 1 GB of free space
5. Display: 1024x768 resolution or higher
6. Internet connection: Required for installation and updates
7. Administrator rights: Installation of PyCharm requires administrator rights on Windows and macOS.
8. Python: PyCharm is a Python IDE, so you will need to have Python installed on your system. You can download the latest version of Python from the official website.
9. Java: PyCharm requires Java to be installed on your system. The latest version of Java can be downloaded from the official website.
10. A Python environment: PyCharm can be used with virtual environments like Anaconda, venv, virtualenv and pipenv.

You may need additional tools or libraries depending on the specific Python development tasks you plan to do. Be sure to check the documentation for any libraries or frameworks you plan to use to ensure that your system meets their requirements.

Minimum requirements for IDLE installation



IDLE (Integrated Development and Learning Environment) is the default, built-in Python development environment that comes with Python. The **requirements to install and run** IDLE are as follows:

1. Operating System: Windows, macOS, or Linux
2. Python: IDLE is built into Python, so you will need to have Python installed on your system. You can download the latest version of Python from the official website.
3. Memory: IDLE is a lightweight development environment, so it does not require a large amount of memory to run.
4. Display: IDLE requires a minimum resolution of 1024x768 or higher.
5. Internet connection: Not required for IDLE installation as it comes with Python installation itself.
6. Administrator rights: IDLE does not require administrator rights to install or run.
7. A Python environment: IDLE can be used with virtual environments like Anaconda, venv, virtualenv and pipenv.

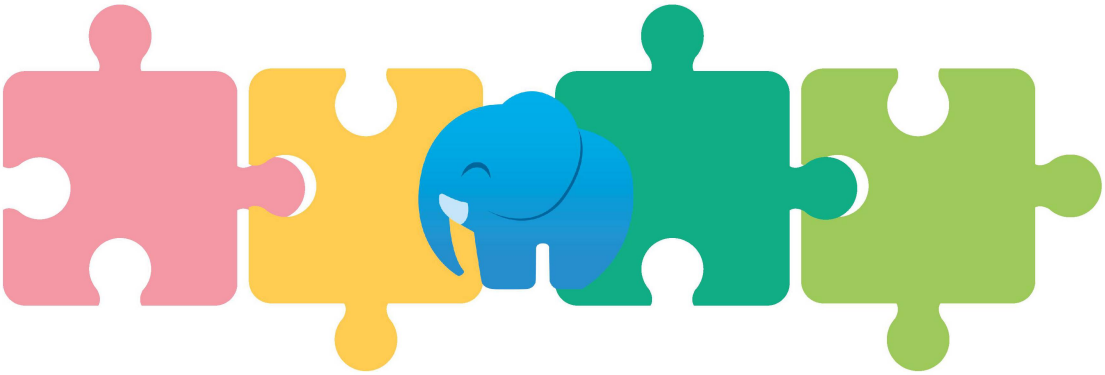
As IDLE is already included with the Python installation, there is no need to install it separately. You just need to open the terminal or command prompt and type python -m idlelib to open IDLE.

Please note that IDLE is a basic text editor and does not have some of the features of advanced IDEs like PyCharm, visual studio code, etc. It's best suited for beginners or small scale projects.

Laragon

The first version of Laragon was released in 2016 and was primarily focused on providing a simple and lightweight package for setting up a local web development environment

Provides a fast and easy way to spin up an isolated Windows development (like a Virtual Machine, it doesn't touch your OS). Users can install it as a software, start it up, do their programming, and just exit when finished. The platform comes pre-installed with many popular applications like Node.



Portable, isolated, fast & powerful universal development environment for PHP, Node. js, Python, Java, Go, Ruby. It is fast, lightweight, easy-to-use and easy-to-extend. Great for building and managing modern web applications.

Built from the ground up for ease of use and perfect compatibility with Laravel.

Before installing Laragon, here are a few things you should be aware of:

1. **System Requirements**: Laragon requires Windows 7 or later, and at least 1 GB of RAM.
2. **Disk Space**: Make sure you have enough disk space to install Laragon and any projects you plan to work on.
3. **Firewall**: If you have a firewall enabled, you may need to configure it to allow Laragon to access the internet and connect to other services.
4. **Anti-virus**: Some anti-virus software may interfere with the installation or operation of Laragon. Make sure to temporarily disable your anti-virus software before installing Laragon, and add an exception for it if needed.
5. **Existing Installation**: If you have an existing web server, PHP, or MySQL installation, you may need to remove it or reconfigure it before installing Laragon.
6. **Backup**: It's always a good idea to make a backup of any important files or data before making any changes to your system, in case something goes wrong during the installation.
7. **Familiarize yourself with the tools**: To get the most out of Laragon, it's a good idea to be familiar with the tools it includes, such as Apache, PHP, and MySQL, so that you can configure and troubleshoot them as needed.

By following these steps, it should make the installation process of Laragon smoother and avoid any potential issues.

CHAPTER 3
INSTALLING PYTHON AND IDE ENVIRONMENT SETUP

CHAPTER 3: INSTALLING PYTHON AND IDE ENVIRONMENT SETUP

01

Python installation. Go to Figure 1 (https://www.python.org/) and download the latest version of Python. Follow the instructions by scan the QR Code.

After installing, you can run Python Interpreter by type IDLE in search Windows and press enter (Figure 2). An IDLE Shell will appear. You can type your coding here (Figure 3).

Source: YouTube

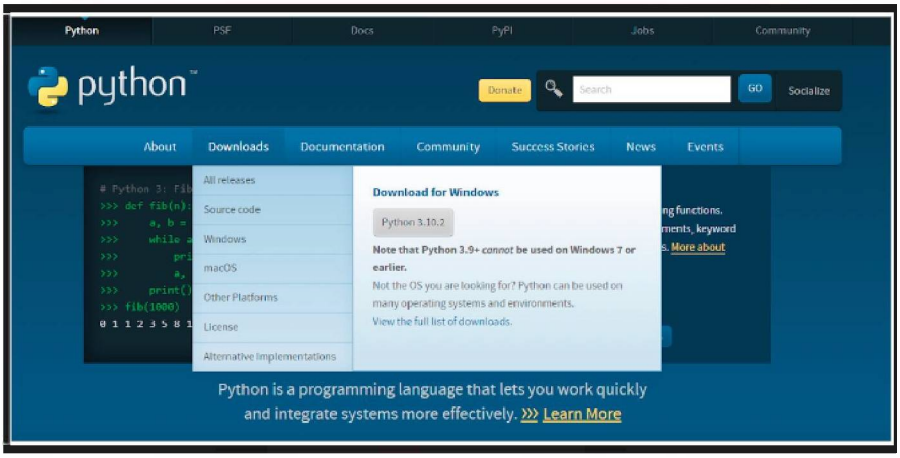


Figure 1: Python.org

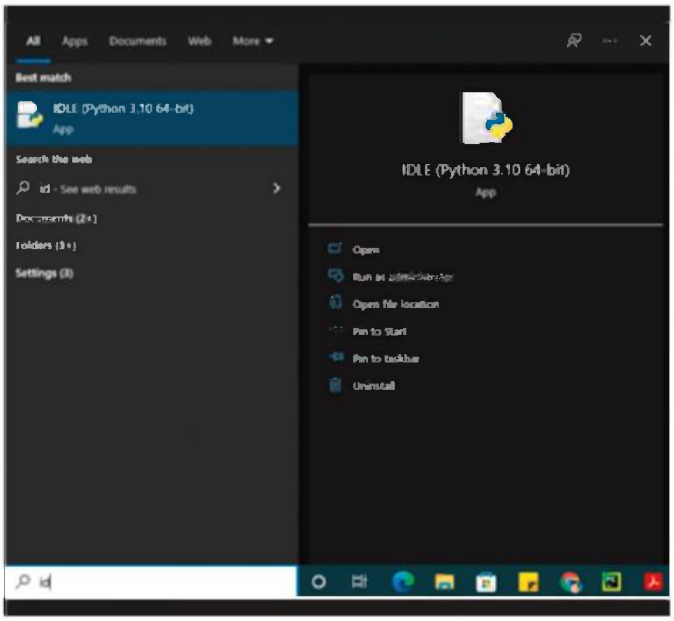


Figure 2: Search IDLE

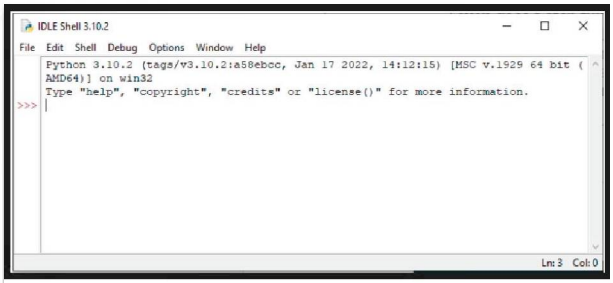


Figure 3: IDLE Shell

02

IDE Installation for Visual Studio Code. Go to Figure 4 (https://code.visualstudio.com/Download) and download the suitable version with your machine. Follow the instructions on How to install Visual Studio Code on Windows 10/11 [2022 Update] Complete Guide by scan the QR Code.

Interface VSCode Figure 5. You can start coding here.

Source: YouTube

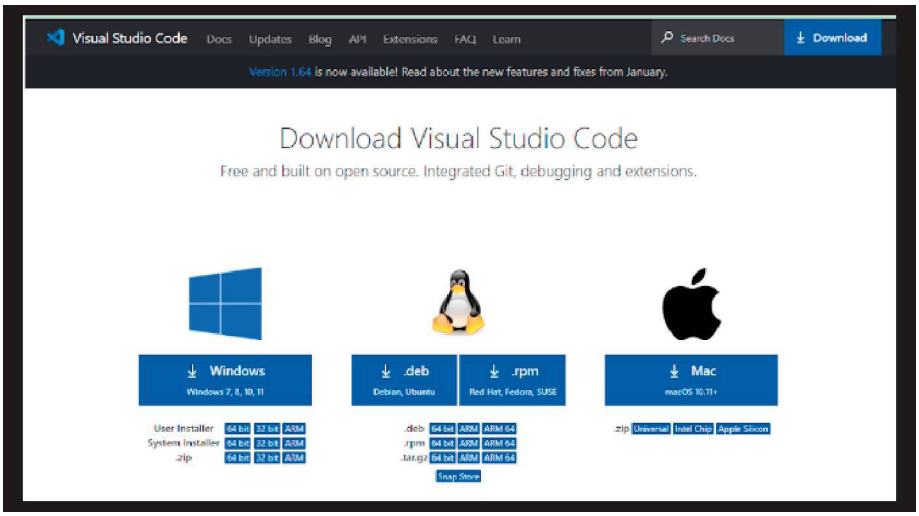


Figure 4: Download Visual Studio Code - Mac, Linux, Windows

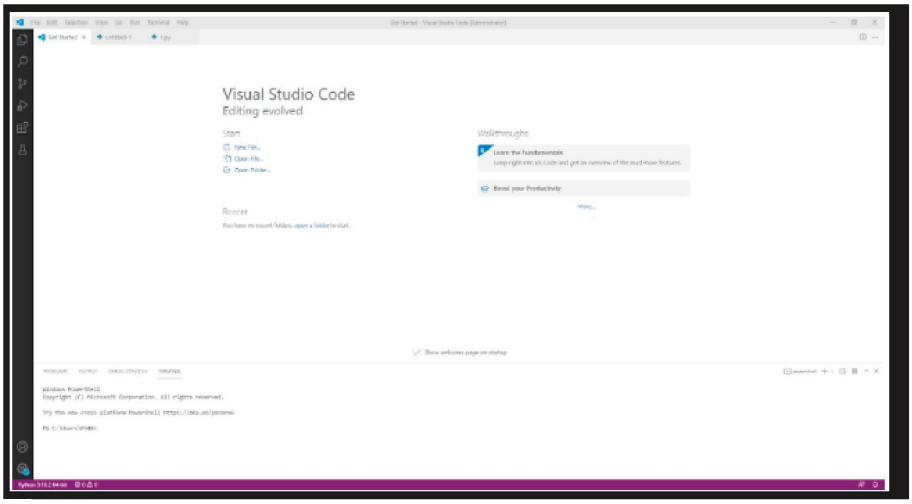


Figure 5: Interface VSCode

03

IIDE Installation for PyCharm. Go to Figure 6 (https://www.jetbrains.com/pycharm/download/#section=windows) and download the Community version. Follow the instructions on Install Python 3.10 and PyCharm on Windows 10 by scan the QR Code.

Source: YouTube

Interface PyCharm Figure 7. You can start coding here. For PyCharm users, if the module involves different projects the module installation needs to be done each time.

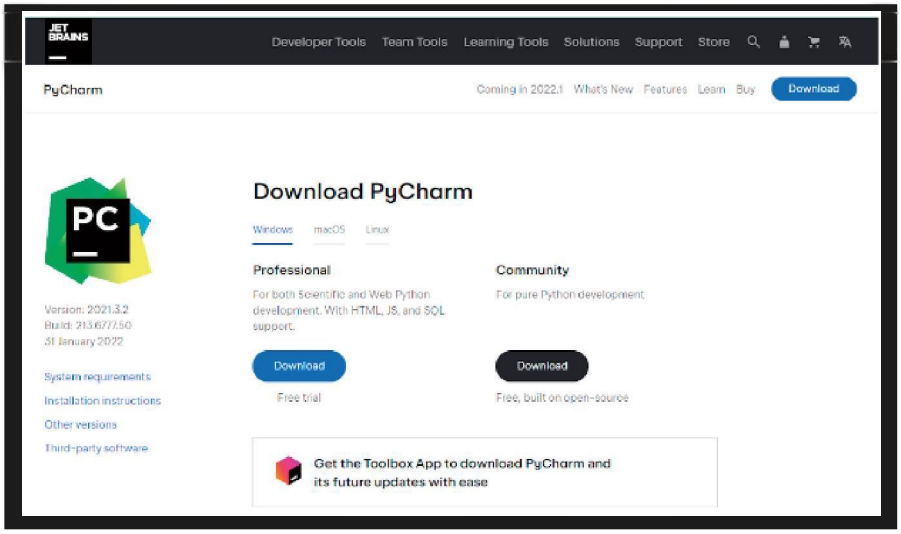


Figure 6: Download PyCharm: Python IDE for Professional Developers by JetBrains

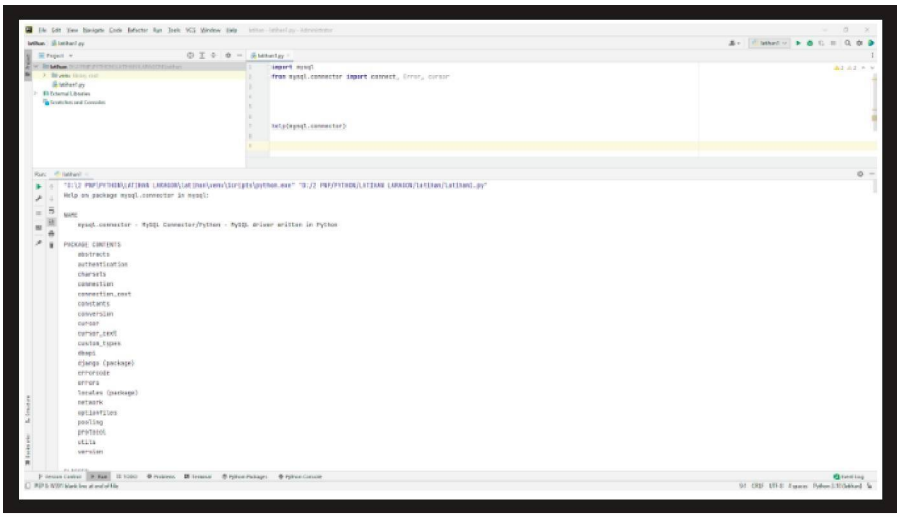


Figure 7: Interface PyCharm

04

Command prompt application needs.

Three steps to check the Python version on your Windows operating system.

1. Open the command prompt application: Press the Windows key to open the start screen. In the search box type "command". Click on the command prompt application as Figure 8.
2. Execute command: type python --version and press enter.
3. The Python version appears in the next line right below your command.

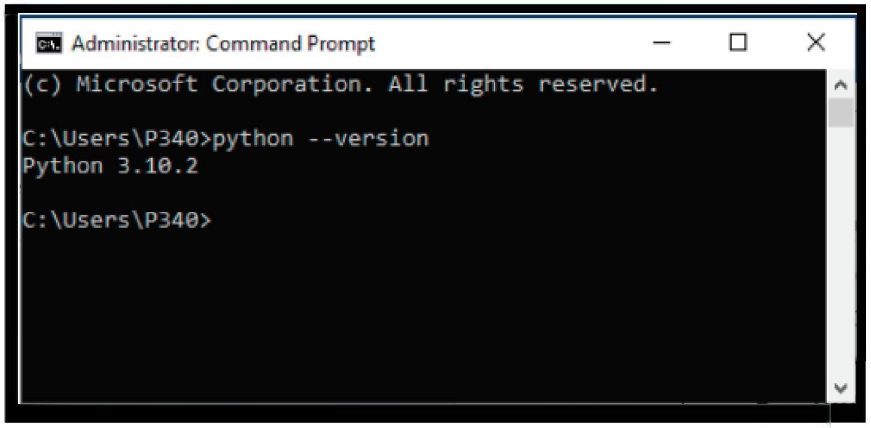


Figure 8: Interface Command Prompt

05

Web server installation for Laragon. Go to Figure 9 (https://laragon.org/download/), download Laragon - Full (173 MB). Follow the instructions for complete installation.

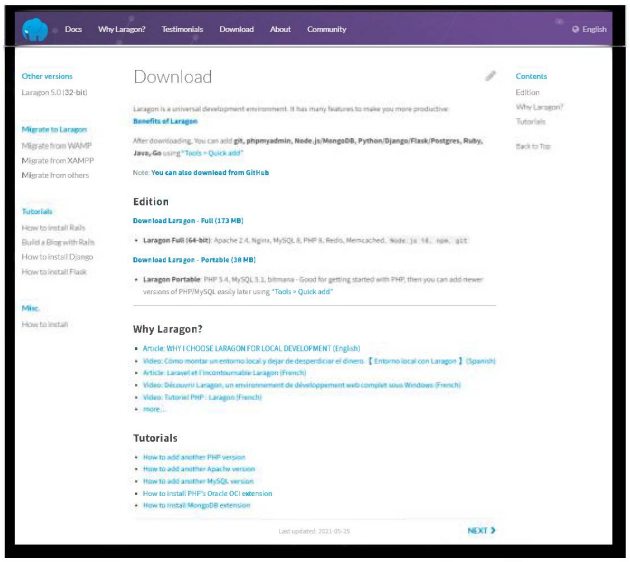


Figure 9: Download | Laragon - portable, isolated, fast & powerful universal development environment for PHP, Node.js, Python, Java, Go, Ruby.

CHAPTER 4
FIRST PROJECT ACTIVITY

CHAPTER 4: FIRST PROJECT ACTIVITY

01

Set up local web server Laragon

Right click Start All > Click MySQL > Click MySQL (Figure 10)

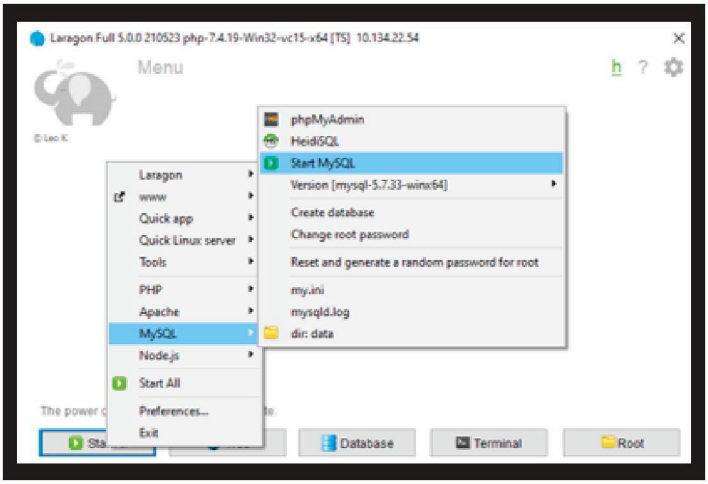


Figure 10: Start up MySql

Click database as Figure 11.

Figure 11: Click Database

01

Session manager window will appear. Click New. *Please make sure the port is correct, user (usually root) with password (usually empty)*, then click open (Figure 12)

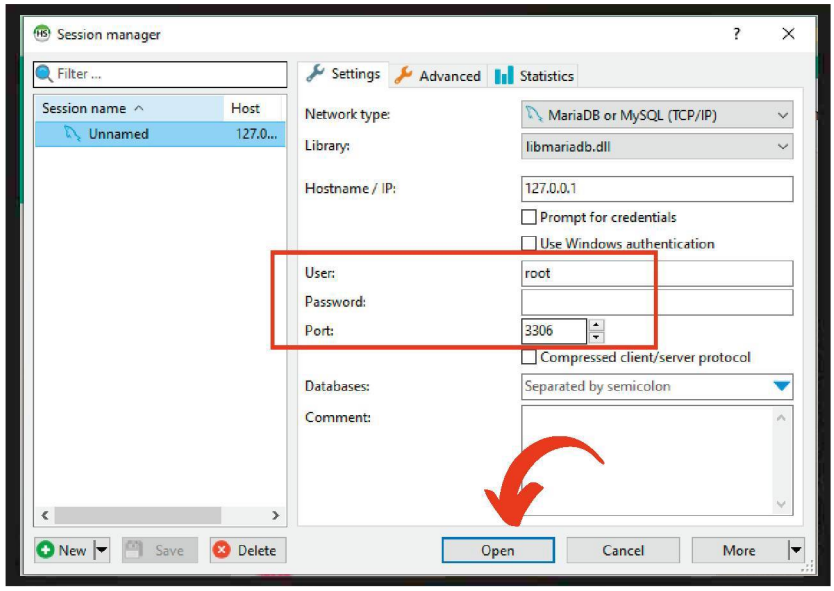


Figure 12: Session Manager

The screen will appear as Figure 13. db1 (database name) and table1(table name). If you want rename -> right-click the database or table name > rename. Press F5 for refresh table or database. If user want to see the data of the table, user need to click selected table > Click data.

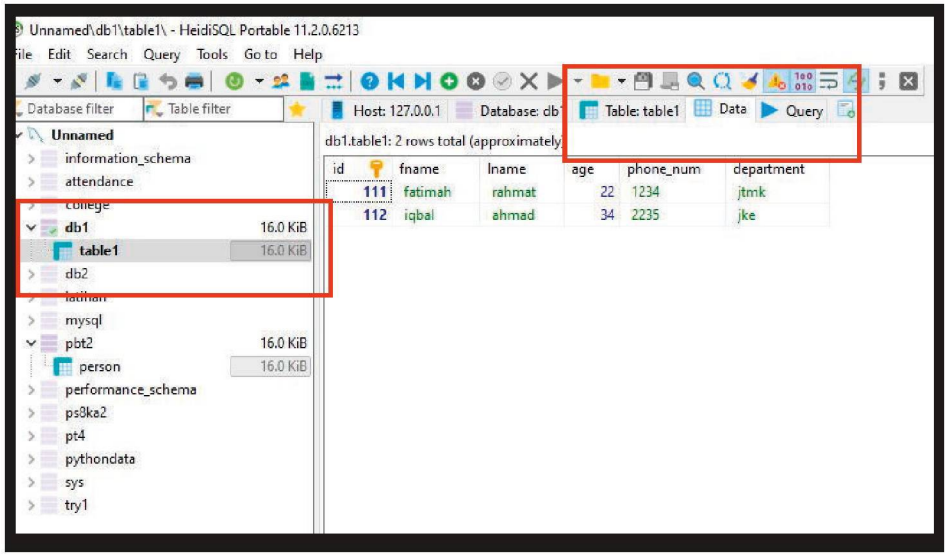


Figure 13: Database Interface

02

Choose your favourite IDE either PyCharm or VSCode.

Those IDE use different database configuration.

PyCharm User Configuration into database

Create a new project named PythonAndMysql.

Click Icon Setting > select Setting or used shortcut key Ctrl+ Alt + S (Figure 14).

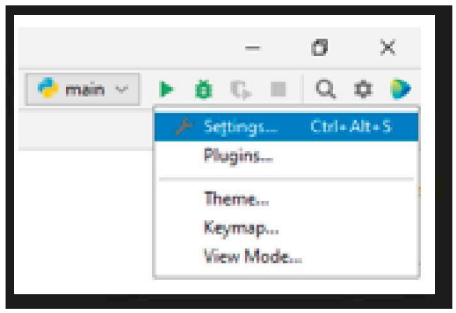


Figure 14: Settings

Pop window will appear.

Expand Project: PythonAndMysql and choose Python Interpreter (Figure 15).

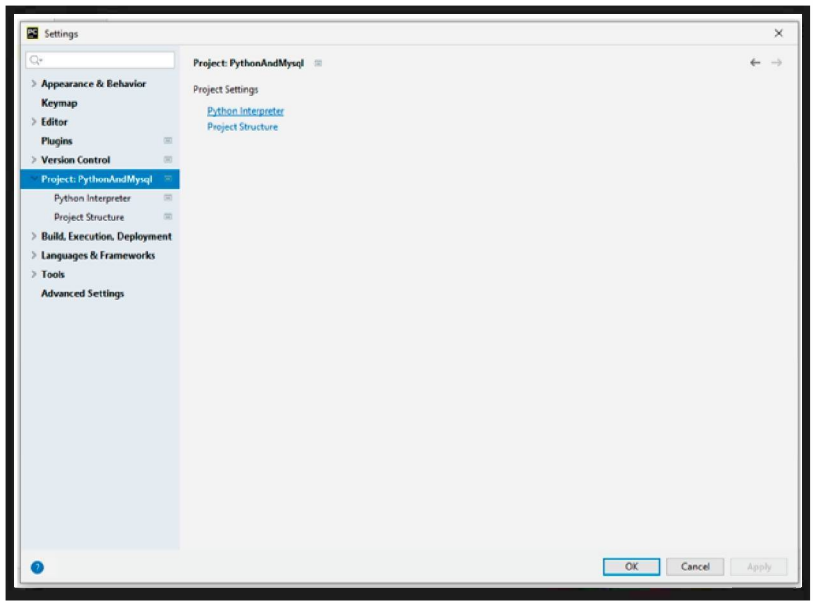
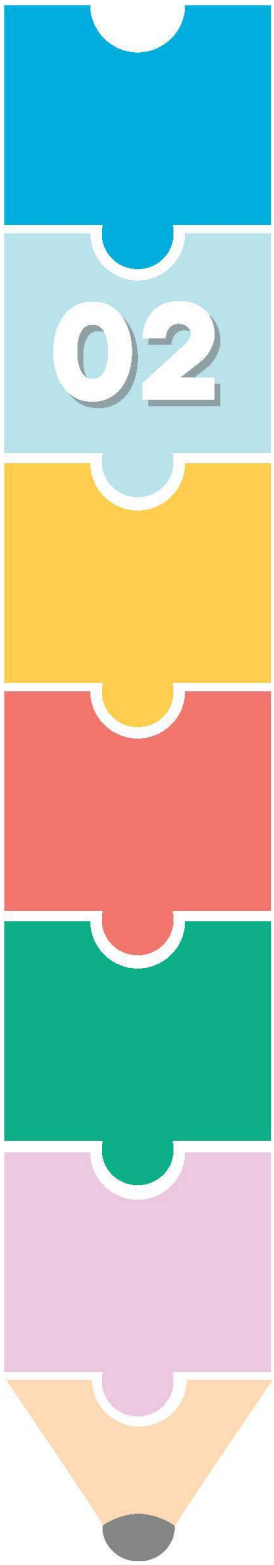


Figure 15: Settings PyCharm window



Python Interpreter interface will be shown. Here we will install a specific module name: mysql.connector. This action can be done by click in plus icon (+) or use shortcut key Alt + Insert (Figure 16)

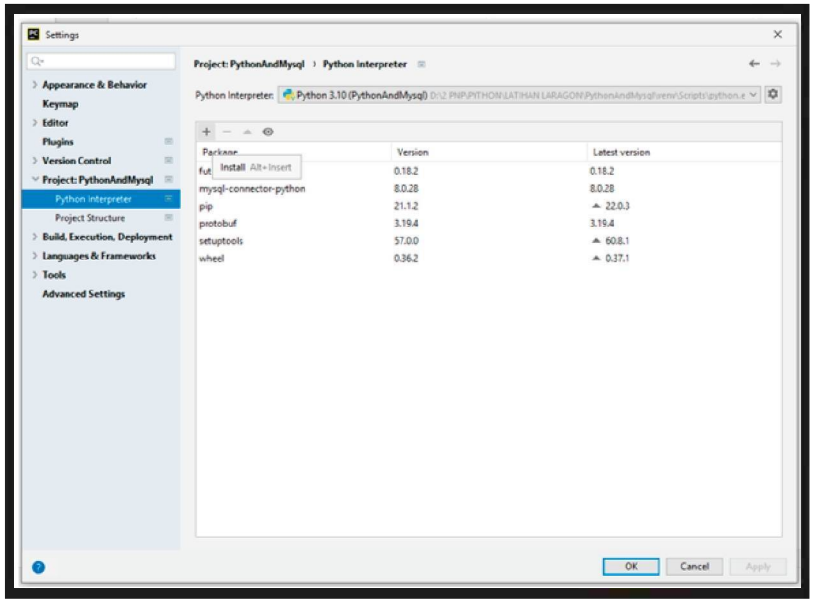


Figure 16: Add module

Search mysql-connector-python on the search field click Install Package. Please be sure that you have an internet connection, since you are installing a package from the cloud. (Figure 17)

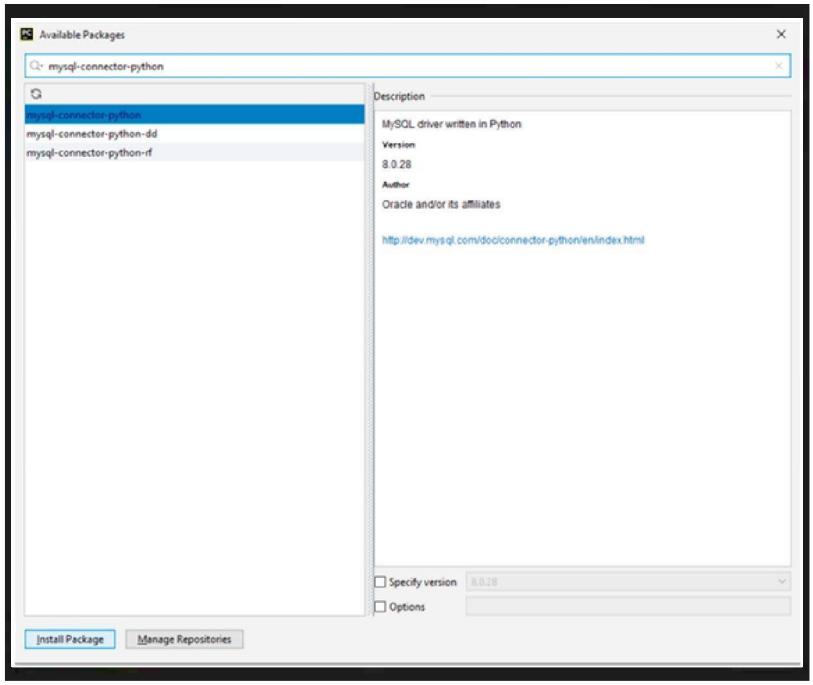
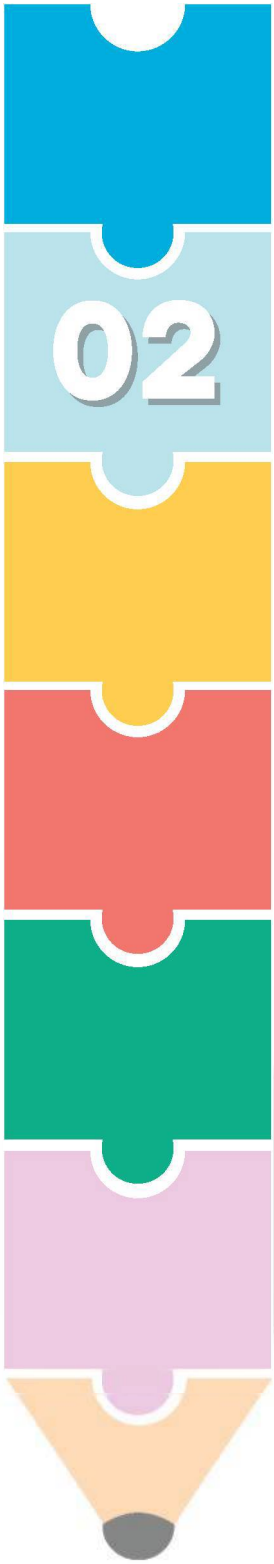


Figure 17: Available Packages window



If installation is successful a green bar will appear with a message stating that the installation is successful. Please try again if not successful. Close the python interpreter pop up screen and head to the code editor interface. (Figure 18).

Figure 18: Package installed successfully

File name main.py is the default fail that will be run by the PyCharm (Figure 19).

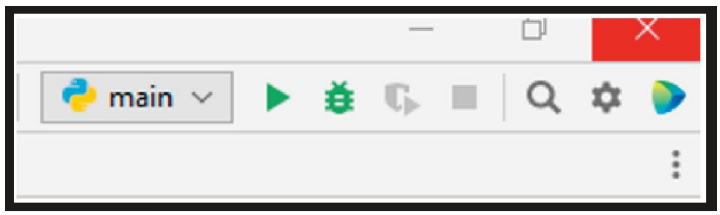


Figure 19: Default file

VSCode User Configuration into database

1. Open the command prompt application: Press the Windows key to open the start screen. In the search box type "command". Click on the command prompt application.
2. Execute command: type pip install mysql-connector-python and press enter.
3. The advantage of using VSCode, the user only needs to install it once and the module can be used by any Python project.

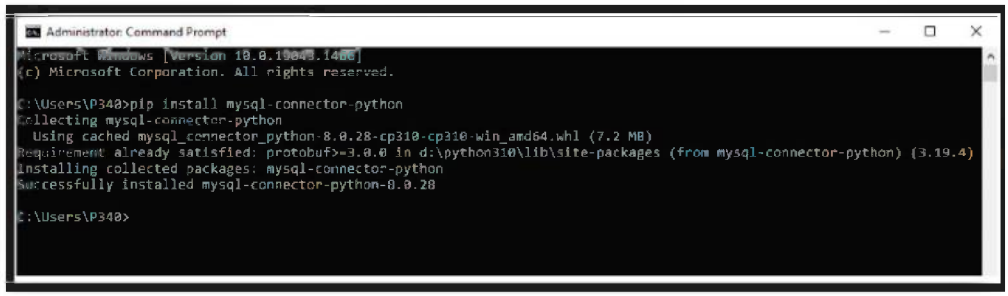


Figure 20: Interface Command Prompt for mysql-connector-python module installation

Requirement mysql.connector installation into database.

MySQL is a Relational Database Management System (RDBMS) whereas the structured Query Language (SQL) is the language used for handling the RDBMS using commands i.e Creating, Inserting, Updating and Deleting the data from the databases. SQL commands are case insensitive i.e CREATE and create signify the same command.

MySQL Connector/Python enables Python programs to access MySQL databases, using an API that is compliant with the Python Database API Specification v2.0 (PEP 249). It is written in pure Python and does not have any dependencies except for the Python Standard Library.

Basic Python keywords do you need to know.

connect: to make a connection to the database, all credentials such as username, server name, server password and database name need to be specified.

error: This is exception handling, if there are any issues, it will be included in this Error

cursor: to help our program execute SQL command/ operation (can be renamed using "as" method)

03

In this activity:

- You will build a program with Create Read Update Delete (**CRUD**) method.
- Your program will display **10 menu** (Figure 21).
- Each menu has it owns specific task. User can choose any menu even there is no database has been created.
- You will create your own specific module and submodule.
- You will build your own function from scratch to fill in the selected menu.
- Use conditional structure to display the menu.
- Write the code and observe the output for each menu.

```
**********POLYTECHNIC MERSING DATABASE**********
1. CREATE DATABASE
2. DROP DATABASE
3. CREATE TABLE
4. DROP TABLE
5. INSERT
6. UPDATE
7. DELETE
8. DISPLAY
9. SHOW DATABASE
10. EXIT

Enter your choice :
```

Figure 21: Program Menu

Let's try the activity

Create module

Create your own module, name it as **moduleDB** in your directory with **four(4) submodules** (Figure 22):

- deleteData.py
- displayData.py
- insertData.py
- updateData.py

Your **main program** is **main.py**.

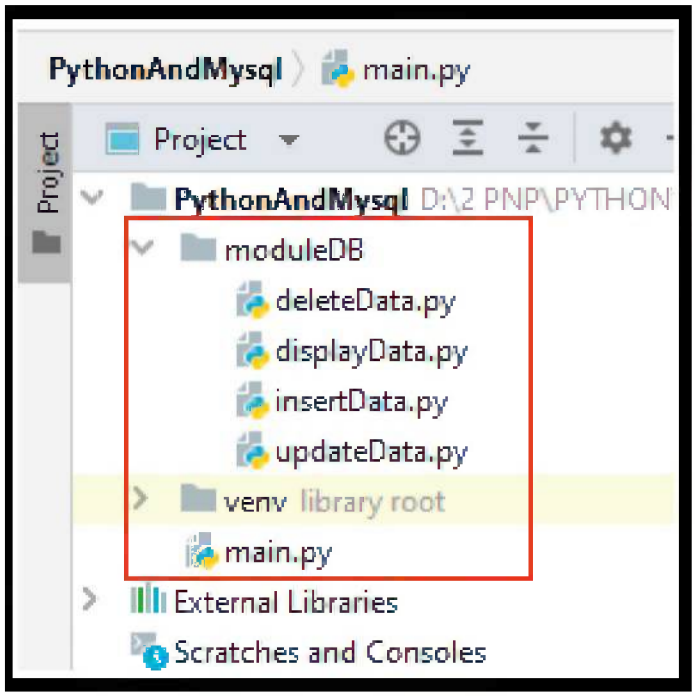


Figure 22: List file in PythonAndMysql directory

Now, move to ModuleDb directory to complete our submodules

1 Create insertData.py

```
import mysql.connector

# insert data into table
def insert(host, username, password, database, table_name):
   try:
        db = mysql.connector.connect(
            host=host,
            user=username,
            password=password,
            database=database
        )

        # A display to insert data
        id = input("Enter Lecturer Id : ")
        fname = input("Enter Lecturer First Name : ")
        lname = input("Enter Lecturer Last Name : ")
        age = input("Enter Age : ")
        phone_num = input("Enter Phone Number : ")
        department = input("Enter Department : ")

        cursor = db.cursor()
          sql = "insert into " + table_name + "(id, fname, lname, age,
phone_num, department) values(%s, %s, %s, %s, %s, %s)"
        print(sql)

        val = (id, fname, lname, age, phone_num, department)
        cursor.execute(sql, val)
        db.commit()
        # For a connection obtained from a connection pool,
        # close() does not actually close it but returns it to the pool
        # and makes it available for subsequent connection requests.
        db.close()
        print('One record inserted into ' + table_name)

   except:
         # .rollback method sends a ROLLBACK statement to the MySQL
server,
         # undoing all data changes from the current transaction.
         # By default, Connector/Python does not autocommit,
         # so it is possible to cancel transactions when using
transactional storage engines such as InnoDB.
        db.rollback()
```

2 Create deleteData.py

```
import mysql.connector

# delete data in table
def delete(host, username, password, database, table_name):
    print("")

     # The connect() constructor creates a connection to the MySQL
server and returns a MySQLConnection object
    db = mysql.connector.connect(
        host=host,
        user=username,
        password=password,
        database=database
    )

     # .cursor method returns a MySQLCursor() object, or a subclass of
it depending on the passed arguments.
    # The returned object is a cursor.
    cursor = db.cursor()
    sql_Delete_query = "Delete from "+table_name+" where id = %s"
    Id = input('Enter lecturer id : ')
     # .execute method executes the given database operation (query or
command)
    cursor.execute(sql_Delete_query, (Id,))

    # This method sends a COMMIT statement to the MySQL server,
    # committing the current transaction.
    # Since by default Connector/Python does not autocommit,
    # it is important to call this method after every transaction
    # that modifies data for tables that use transactional storage
engines.
    # https://dev.mysql.com/doc/connector-python/en/connector-python-
api-mysqlconnection-commit.html
    db.commit()
    print("One row(s) deleted from "+table_name)
```

3 Create displayData.py

```
import mysql.connector

# display data
def displayTableInfromation(host, username, password, database,table_name):

    # The connect() constructor creates a connection to the MySQL server and
returns a MySQLConnection object
    db = mysql.connector.connect(
        host=host,
        user=username,
        password=password,
        database=database
    )

     # .cursor method returns a MySQLCursor() object, or a subclass of it
depending on the passed arguments.
    # The returned object is a cursor.
    cursor = db.cursor()

    # Reports whether the connection to MySQL Server is available.
    if db.is_connected():

          # .execute method executes the given database operation (query or
command)
        cursor.execute("SELECT * FROM "+table_name)

        # row is just a variable.
        # this variable will print every column for each row of data.
        for row in cursor:
            print("\nLecturer ID : ", row[0])
            print("First Name : ", row[1])
            print("Last Name : ", row[2])
            print("Age : ", row[3])
            print("Phone Number : ", row[4])
            print("Department : ", row[5])
```

4 Create updateData.py

```
import mysql.connector

# update data into table
def update(host, username, password, database, table_name):
    db = mysql.connector.connect(
        host=host,
        user=username,
        password=password,
        database=database
    )
    id = input("Enter Lecturer Id : ")
    fname = input("Enter Lecturer First Name : ")
    lname = input("Enter Lecturer Last Name : ")
    age = input("Enter Age : ")
    phone_num = input("Enter Phone Number : ")
    department = input("Enter Department : ")

    cursor = db.cursor()
    try:
        sqlFormula = "UPDATE "+table_name+" SET fname = %s WHERE id = %s"
        cursor.execute(sqlFormula, (fname, id))

        sqlFormula = "UPDATE "+table_name+" SET lname = %s WHERE id = %s"
        cursor.execute(sqlFormula, (lname, id))

        sqlFormula = "UPDATE "+table_name+" SET age = %s WHERE id = %s"
        cursor.execute(sqlFormula, (age, id))

        sqlFormula = "UPDATE "+table_name+" SET phone_num = %s WHERE id = %s"
        cursor.execute(sqlFormula, (phone_num, id))

       sqlFormula = "UPDATE "+table_name+" SET department = %s WHERE id = %s"
        cursor.execute(sqlFormula, (department, id))

        db.commit()

        print("entries updated in " + table_name)

    except:
        db.rollback()
```

5 Run and observe the output.

01

Import module in main.py

```
# import mysql.connector with specified submodule to be use: connect,
Error and cursor
from mysql.connector import connect, Error, cursor

# import our own module moduleDB and specified submodule insertData,
updateData, deleteData and displayData
# The "as" keyword is used to create an alias
from moduleDB import insertData as insert, updateData as update,
deleteData as delete, displayData as display
```

02

Create database function in main.py

```
# function 1: create new database
def create_db(username, password):
    try: # try block lets you test a block of code for errors
        with connect(
                # "host" is server name / targeted server to be connected
                host=glbHost,
                user=username,
                password=password
        ) as connection:
            dbname = input("What is DB name you want to create? ")
            create_db_query = "CREATE DATABASE "+dbname
            print(create_db_query)
            with connection.cursor() as cursor:
                cursor.execute(create_db_query)
            print("Database "+dbname+" is created!")
    # except block lets you handle the error
    except Error as e:
        print("Opss, something is wrong", e)
    # NameError raised when a variable is not found in local or global
scope.
    except NameError:
        print("NameError is raised when the identifier being accessed is
not defined in the local or global scope.")
```

03

Drop database function in main.py

```
# function 2: delete database
def drop_db(username, password):
    try:
        # database connection
        # set all database credential (host,user, password)
        with connect(
                host=glbHost,
                user=username,
                password=password
        )as connection:
            dbname = input("What is DB name you want to drop?: ")
            drop_db_query = "DROP DATABASE %s" % dbname
            with connection.cursor() as cursor:
                cursor.execute(drop_db_query)
            print('Database ', dbname, ' has been dropped.')

    except Error as e:
        print(format(e))
```

04

Create table function in main.py

```
# function 3: create new table
def create_table(username, password):
    dbName = input('Enter database name first: ')
    try:
        # database connection
        # set all database credential (host,user, password, database)
        with connect(
                host=glbHost,
                user=username,
                password=password,
                database=dbName
        ) as connection:
            table_name = input('insert table name you want to crete: ')
            create_table_query = "CREATE TABLE "+table_name+"
            (id INT AUTO_INCREMENT PRIMARY KEY, fname VARCHAR(50), " \
             "lname VARCHAR(50), age INT, phone_num VARCHAR(100), " \
             "department VARCHAR(100)) "
            print(create_table_query)
            with connection.cursor() as cursor:
                cursor.execute(create_table_query)

                for x in cursor:
                    print(x)
    except Error as e:
        print(format(e))
```

05

Drop table function in main.py

```
# function 4: delete table
def drop_table(username, password):
    dbName = input('From which database you want to delete the table?: ')
    try:
        # database connection
        # set all database credential (host,user, password, database)
        with connect(
                host=glbHost,
                user=username,
                password=password,
                database=dbName
        ) as connection:
            dbtable = input('Enter table name : ')
            drop_table_query = "DROP Tables " + dbtable
            print(drop_table_query)
            with connection.cursor() as cursor:
                cursor.execute(drop_table_query)
                print('Table', dbtable, 'has been dropped.')
    except Error as e:
        print(format(e))
```

06

Display all databases function

```
# function 5: display all database existed
def show_database(username, password):
    try:
        # database connection
        # set all database credential (host,user, password)
        with connect(
                host=glbHost,
                user=username,
                password=password
        ) as connection:
            cursor = connection.cursor()
            databases = ("show databases")
            cursor.execute(databases)
            i = 0
            for (databases) in cursor:
                i = i+1
                print("Database ",i ,"is :", databases[0])
    except Error as e:
        print("Opss, something is wrong", e)
    except NameError:
        print("NameError is raised when the identifier being accessed is not
defined in the local or global scope.")
```

07

Finish the whole program

```
# display text
print("**********POLYTECHNIC MERSING DATABASE**********")
print("1. CREATE DATABASE \n2. DROP DATABASE \n3. CREATE TABLE "
      "\n4. DROP TABLE \n5. INSERT \n6. UPDATE \n7. DELETE "
      "\n8. DISPLAY \n9. SHOW DATABASE \n10. EXIT\n")

while True:
    choice = input("Enter your choice :")
    if choice in ('1', '2', '3', '4', '5', '6', '7', '8', '9'):

        # glbHost is a global variable for "localhost"
        # "localhost" is the targeted server to be connected
        glbHost = "localhost"
        u = input("Enter Mysql Username: ")
        p = input("Enter Mysql Password: ")
        i = 0

        if choice == '1':
            # go create_db function
            create_db(u, p)

        elif choice == '2':
            # go to drop_db function
            drop_db(u, p)

        elif choice == '3':
            # go to create_table function
            create_table(u, p)

        elif choice == '4':
            # go to drop_table function
            drop_table(u, p)

        elif choice == '5':
            database = input("Enter database name: ")
            table_name = input("Enter table name: ")
            # go to moduleDB insertData.py
            insert.insert(glbHost, u, p, database, table_name)

        elif choice == '6':
            database = input("Enter database name: ")
            table_name = input("Enter table name: ")
            # go to moduleDB updateData.py
            update.update(glbHost, u, p, database, table_name)

        elif choice == '7':
            database = input("Enter database name: ")
            table_name = input("Enter table name: ")
            # go to moduleDB deleteData.py
            delete.delete(glbHost, u, p, database, table_name)

        elif choice == '8':
            database = input("Enter database name: ")
            table_name = input("Enter table name: ")
            # go to moduleDB deleteData.py
            display.displayTableInfromation(glbHost, u, p, database,
table_name)

        elif choice == '9':
            # go to show_database function
            show_database(u, p)

        elif choice == '10':
            exit()
    else:
        print("Invalid Input ")
```

References

Akhter, M. (2022, January 11). How to install Visual Studio Code on Windows 10/11 [2022 Update] Complete Guide [Video]. YouTube. https://www.youtube.com/watch?v=JPZsB_6yHVo

Ansgar, B. (n.d.). HeidiSQL. Retrieved from https://www.heidisql.com/

Computer Science. (2021, November 13). Install Python 3.10 and PyCharm on Windows 10 [Video]. YouTube. https://www.youtube.com/watch?v=WJynvGY-2wk

Downey, A. (n.d.). Introduction to GUI programming. Python Textbook. Retrieved from https://python-textbok.readthedocs.io/en/1.0/Introduction_to_GUI_Programming.html

GeeksforGeeks. (n.d.). MySQL-Connector-Python module in Python. Retrieved from https://www.geeksforgeeks.org/mysql-connector-python-module-in-python/

JetBrains. (n.d.). PyCharm download. Retrieved from https://www.jetbrains.com/pycharm/download/#section=windows

Laragon. (n.d.). Laragon - portable, isolated, fast & powerful universal development environment for PHP, Node.js, Python, Java, Go, Ruby. Retrieved from https://laragon.org/download/

Microsoft. (n.d.). Visual Studio Code download. Retrieved from https://code.visualstudio.com/Download

Ngo, J. (2021, February 3). PyCharm vs. VS Code: Which is the better code editor? [Blog post]. LogRocket. https://blog.logrocket.com/pycharm-vs-vscode/#:~:text=PyCharm%20and%20VS%20Code%20are,to%20an%20IDE%20through%20extensions.

Oracle Corporation. (n.d.). MySQL Connector/Python Developer Guide. Retrieved from https://dev.mysql.com/doc/connector-python/en/

Python Software Foundation. (n.d.). IDLE. Python 3 documentation. Retrieved from https://docs.python.org/3/library/idle.html

Python Software Foundation. (n.d.). Python. Retrieved from https://www.python.org/

Santos, A. (2021, November 5). How to Install Python 3.10.2 on Windows 10 [Video]. YouTube. https://www.youtube.com/watch?v=uKHVNKd3f20

Sid Martin Biotechnology Institute. (n.d.). What is Laragon used for? Retrieved from https://www.sidmartinbio.org/what-is-laragon-used-for/

Srinivasan, S. (2019, December 3). PyCharm vs VS Code. Tangent Technologies. https://tangenttechnologies.ca/blog/pycharm-vs-vscode/

POLITEKNIK MERSING
Jalan Nitar,
86800 Mersing
Johor Darul Ta'zim
Telephone : 07-7980001
Fax : 07-7980002
Website : https://pmj.mypolycc.edu.my/

e ISBN 978-967-2904-57-1