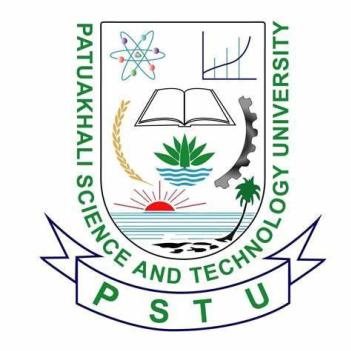
**PATUAKHALI SCIENCE AND TECHNOLOGY UNIVERSITY**

****

**COURSE CODE CCE-224**

**SUBMITTED TO:**

**Prof. Dr. Md Samsuzzaman sobuz**

**Department of Computer and Communication**

**Engineering**

**Faculty of Computer Science and Engineering**

****

**SUBMITTED BY:**

**Abul Basar**

ID: **2102036**,

Registration No: **10163**

**Faculty of Computer Science and Engineering**

****

Date of submission: **14-05-2025**

**Lab-01: XAMPP Installation and University Database Schema Implementation**

**Part 1: Installing XAMPP and Notes on XAMPP Components**

**1. Installing XAMPP**

**XAMPP** is a free, open-source web server solution stack developed by Apache Friends, designed to create a local web server environment for testing and development. Below are the steps to **install XAMPP** on a Windows system (adaptable for macOS or Linux with minor variations):

1. **Download XAMPP**:
   * Visit the official Apache Friends website at <https://www.apachefriends.org>.
   * Select the **XAMPP** version compatible with your operating system (e.g., Windows, macOS, or Linux).
   * Download the installer (e.g., .exe for Windows).
2. **Run the Installer**:
   * Double-click the downloaded .exe file to launch the **XAMPP Setup Wizard**.
   * Click **Next** to proceed.
   * If prompted by Windows User Account Control (UAC), click **Yes** to allow the installation.
3. **Choose Components**:
   * In the **Choose Components** screen, select the components to install. For this assignment, ensure **Apache**, **MySQL** (MariaDB), **PHP**, and **phpMyAdmin** are checked.
   * Optionally, uncheck unnecessary components like FileZilla or Mercury to save space.
   * Click **Next**.
4. **Select Installation Directory**:
   * Choose the default installation path (e.g., C:\xampp on Windows) or specify another location.
   * Click **Next** and then **Install** to begin extracting files.
5. **Complete Installation**:
   * Once installation is complete, click **Finish**.
   * Optionally, check the box to open the **XAMPP Control Panel** immediately.
6. **Start XAMPP Services**:
   * Open the **XAMPP Control Panel** (e.g., xampp-control.exe in the installation folder).
   * Click **Start** next to **Apache** and **MySQL** modules. Ensure they turn green, indicating they are running.
   * If ports are blocked (e.g., port 80 for Apache), configure alternative ports (e.g., 8080) in the Apache configuration file (httpd.conf).
7. **Test Installation**:
   * Open a web browser and navigate to http://localhost. The **XAMPP dashboard** should appear.
   * Access **phpMyAdmin** by visiting http://localhost/phpmyadmin to verify database functionality.

**Note**: For security, set a password for the **MySQL root user** via the **XAMPP Security Console** (accessible from the dashboard) and stop **Apache** and **MySQL** when not in use to save resources.

**2. Short Notes on XAMPP Components**

**XAMPP** stands for **Cross-Platform (X), Apache (A), MariaDB/MySQL (M), PHP (P), and Perl (P)**. Below are concise notes on its core components:

* **Apache**:
  + **Definition**: An open-source **web server** software developed by the Apache Software Foundation.
  + **Function**: Serves web content (e.g., HTML, PHP pages) over HTTP/HTTPS to browsers.
  + **Role in XAMPP**: Hosts web applications locally, allowing developers to test websites before deployment.
  + **Key Feature**: Configurable via httpd.conf for custom ports, virtual hosts, and modules.
* **MariaDB/MySQL**:
  + **Definition**: A relational **database management system** (RDBMS), with **MariaDB** replacing **MySQL** in newer **XAMPP** versions.
  + **Function**: Stores and manages structured data for web applications using **SQL**.
  + **Role in XAMPP**: Provides a database backend for dynamic websites (e.g., WordPress, CMS).
  + **Key Feature**: Supports efficient data storage, retrieval, and manipulation.
* **PHP**:
  + **Definition**: A server-side **scripting language** (Hypertext Preprocessor) for web development.
  + **Function**: Generates dynamic web content by interacting with databases and servers.
  + **Role in XAMPP**: Powers server-side logic for web applications, integrated with **Apache**.
  + **Key Feature**: Configurable via php.ini for error handling, extensions, and performance.
* **Perl**:
  + **Definition**: A high-level **programming language** designed for text processing and web development.
  + **Function**: Supports scripting for web applications, network programming, and automation.
  + **Role in XAMPP**: Less commonly used but included for compatibility with legacy applications.
  + **Key Feature**: Versatile for parsing and processing large datasets.
* **phpMyAdmin**:
  + **Definition**: A web-based **database administration tool** written in **PHP**.
  + **Function**: Provides a graphical interface to manage **MariaDB/MySQL** databases.
  + **Role in XAMPP**: Simplifies database creation, table management, and **SQL** query execution.
  + **Key Feature**: Supports import/export of databases and user account management.

**Additional Tools**:

* **FileZilla FTP Server**: Facilitates file transfers for web applications.
* **Mercury Mail Server**: Supports local email testing.
* **Tomcat**: Enables Java-based web applications (optional).

**Cross-Platform Support**: **XAMPP** runs on Windows, macOS, and Linux, making it versatile for developers.

**Part 2: Creating a Database and University Database Schema**

**1. Creating a Database**

Assuming your ID is a placeholder (e.g., student123), we will create a **database** named student123. Follow these steps using **phpMyAdmin**:

1. **Access phpMyAdmin**:
   * Open a browser and navigate to http://localhost/phpmyadmin.
   * Ensure **Apache** and **MySQL** are running in the **XAMPP Control Panel**.
2. **Create Database**:
   * In **phpMyAdmin**, click the **Databases** tab in the top menu.
   * Under **Create database**, enter the database name student123.
   * Select **Collation** as utf8mb4\_unicode\_ci (recommended for Unicode support).
   * Click **Create**. The database student123 will appear in the left sidebar.
3. **Verify Database**:
   * Click on student123 in the left sidebar to select it.
   * The interface will display options to create tables within this database.

**2. University Database Schema Implementation**

The **university database schema** typically includes tables to manage students, courses, departments, instructors, and enrollments. Since the specific schema was not provided, a standard **university database schema** is assumed with the following tables:

* **Department**: Stores department information.
* **Instructor**: Stores instructor details.
* **Student**: Stores student information.
* **Course**: Stores course details.
* **Enrollment**: Tracks student course enrollments.

Below are the **SQL** statements to create these tables in the student123 database, including appropriate **attributes**, **data types**, **primary keys**, and **foreign keys**.

**Step-by-Step Table Creation**

1. **Select Database**:
   * In **phpMyAdmin**, click on the student123 database in the left sidebar.
2. **Create Tables Using SQL**:
   * Click the **SQL** tab in **phpMyAdmin**.
   * Copy and paste the following **SQL** code to create the tables.
   * Click **Go** to execute the query.

sql

Copy

*-- Create Department Table*

CREATE TABLE Department (

dept\_id INT PRIMARY KEY,

dept\_name VARCHAR(50) NOT NULL,

building VARCHAR(50)

);

*-- Create Instructor Table*

CREATE TABLE Instructor (

instructor\_id INT PRIMARY KEY,

first\_name VARCHAR(50) NOT NULL,

last\_name VARCHAR(50) NOT NULL,

dept\_id INT,

FOREIGN KEY (dept\_id) REFERENCES Department(dept\_id)

);

*-- Create Student Table*

CREATE TABLE Student (

student\_id INT PRIMARY KEY,

first\_name VARCHAR(50) NOT NULL,

last\_name VARCHAR(50) NOT NULL,

dept\_id INT,

enrollment\_year INT,

FOREIGN KEY (dept\_id) REFERENCES Department(dept\_id)

);

*-- Create Course Table*

CREATE TABLE Course (

course\_id INT PRIMARY KEY,

course\_name VARCHAR(100) NOT NULL,

dept\_id INT,

credits INT,

FOREIGN KEY (dept\_id) REFERENCES Department(dept\_id)

);

*-- Create Enrollment Table*

CREATE TABLE Enrollment (

student\_id INT,

course\_id INT,

semester VARCHAR(20),

grade VARCHAR(2),

PRIMARY KEY (student\_id, course\_id, semester),

FOREIGN KEY (student\_id) REFERENCES Student(student\_id),

FOREIGN KEY (course\_id) REFERENCES Course(course\_id)

);

1. **Verify Table Creation**:
   * Click the **Structure** tab in **phpMyAdmin**.
   * Confirm that the tables (Department, Instructor, Student, Course, Enrollment) appear with their respective columns and constraints.
   * Check the **Relation view** (if available) to verify **foreign key** relationships.

**Table Descriptions**

* **Department**:
  + **Attributes**: dept\_id (unique identifier), dept\_name (department name), building (location).
  + **Primary Key**: dept\_id.
  + **Purpose**: Stores department details, referenced by other tables.
* **Instructor**:
  + **Attributes**: instructor\_id (unique identifier), first\_name, last\_name, dept\_id (links to department).
  + **Primary Key**: instructor\_id.
  + **Foreign Key**: dept\_id references Department(dept\_id).
  + **Purpose**: Manages instructor information.
* **Student**:
  + **Attributes**: student\_id (unique identifier), first\_name, last\_name, dept\_id, enrollment\_year.
  + **Primary Key**: student\_id.
  + **Foreign Key**: dept\_id references Department(dept\_id).
  + **Purpose**: Stores student details.
* **Course**:
  + **Attributes**: course\_id (unique identifier), course\_name, dept\_id, credits.
  + **Primary Key**: course\_id.
  + **Foreign Key**: dept\_id references Department(dept\_id).
  + **Purpose**: Manages course information.
* **Enrollment**:
  + **Attributes**: student\_id, course\_id, semester, grade.
  + **Primary Key**: Composite key (student\_id, course\_id, semester).
  + **Foreign Keys**: student\_id references Student(student\_id), course\_id references Course(course\_id).
  + **Purpose**: Tracks student enrollments and grades.

**Sample Data Insertion (Optional)**

To test the database, you can insert sample data using the following **SQL** statements in the **SQL** tab:

sql

Copy

*-- Insert into Department*

INSERT INTO Department (dept\_id, dept\_name, building) VALUES

(1, 'Computer Science', 'Tech Hall'),

(2, 'Mathematics', 'Math Building');

*-- Insert into Instructor*

INSERT INTO Instructor (instructor\_id, first\_name, last\_name, dept\_id) VALUES

(101, 'John', 'Smith', 1),

(102, 'Jane', 'Doe', 2);

*-- Insert into Student*

INSERT INTO Student (student\_id, first\_name, last\_name, dept\_id, enrollment\_year) VALUES

(1001, 'Alice', 'Brown', 1, 2023),

(1002, 'Bob', 'Wilson', 2, 2024);

*-- Insert into Course*

INSERT INTO Course (course\_id, course\_name, dept\_id, credits) VALUES

(201, 'Database Systems', 1, 3),

(202, 'Calculus I', 2, 4);

*-- Insert into Enrollment*

INSERT INTO Enrollment (student\_id, course\_id, semester, grade) VALUES

(1001, 201, 'Fall 2023', 'A'),

(1002, 202, 'Spring 2024', 'B+');

1. **Test the Database**:
   * Use the **SQL** tab to run queries, e.g., SELECT \* FROM Student; to view student records.
   * Explore **phpMyAdmin**’s interface to browse tables, export data, or modify structures.