**ĐỒ ÁN THỰC HÀNH 1 – LẬP TRÌNH SOCKET**

**MÔN MẠNG MÁY TÍNH**

# **Quy định chung**

- Đồ án được làm theo nhóm: mỗi nhóm tối đa **3** sinh viên, tối thiểu **2** sinh viên (trong trường hợp sỉ số lớp lẻ), sinh viên tự chọn nhóm (sử dụng nhóm thực hành đã đăng ký nếu có). Nhóm sinh viên sẽ chọn đề tài thỏa quy định sau:

***Mã đề tài = (tổng chữ số cuối cùng MSSV của các sinh viên) mod 2 + 1***

- **Các bài làm giống nhau sẽ đều bị điểm 0 toàn bộ phần thực hành tất cả các nhóm liên quan (dù có điểm các bài tập, đồ án thực hành khác)**.

- Môi trường lập trình: Tự do lựa chọn ngôn ngữ lập trình, tự do lựa chọn môi trường hệ điều hành: Windows, Unix/Linux, macOS

- Ngôn ngữ lập trình GV có thể hỗ trợ: C/C++, C#, Java, Python

- Thư viện hỗ trợ lập trình socket cho phép sử dụng: Socket, CSocket, winsock. Tức là chỉ sử dụng các thư viện Socket do ngôn ngữ lập trình cung cấp. **Không phải lập trình website.**

# **Cách thức nộp bài**

* **Nộp bài trực tiếp trên Website môn học, không chấp nhận nộp bài qua email hay hình thức khác.**
* Tên file: **MÃ-ĐỀ\_ MSSV1\_MSSV2\_MSSV3.zip** (Với MSSV1 < MSSV2 < MSSV3)

Ví dụ: Nhóm gồm 2 sinh viên: 2012001, 2012002, và 2012003 làm đề 1, tên file nộp: **1\_2012001\_2012002\_2012003.zip**

**Cấu trúc file nộp gồm:**

1. **Report.pdf:** chứa báo cáo về bài làm
2. **Release**: thư mục chứa file thực thi của chương trình, **nếu có** (\*.exe/ ...). Lưu ý: Nếu viết bằng Python hoặc ngôn ngữ tương tự thì không cần tạo file \*.exe.
3. **Source**: thư mục chứa source code của chương trình, yêu cầu nộp cả project **đã xoá bỏ thư mục Debug** và các file không cần thiết khác.. *Nhóm nào chỉ nộp file \*.cpp và \*.h và không biên dịch được thì bị 0 điểm.*

***Lưu ý: Cần thực hiện đúng các yêu cầu trên, nếu không, bài làm sẽ không được chấm.***

# **Hình thức chấm bài**

Chấm vấn đáp vào thời điểm kết thúc phần thực hành.

# **Tiêu chí đánh giá**

## Về chương trình:

- Mục tiêu của đồ án này tập trung chủ yếu vào 2 vấn đề: lập trình socket, xây dựng giao thức trao đổi giữa client và server. Do đó các tiêu chí đánh giá dựa vào các chức năng chính được liệt kê trong yêu cầu của chương trình (có ghi chú thang điểm cho từng chức năng)

## Về báo cáo:

* Thông tin của nhóm.
* Đánh giá mức độ hoàn thành từ 0 – 100% (Chú thích rõ những mục làm được,chưa làm được và còn bị lỗi)
* Kịch bản giao tiếp của chương trình: Giao thức trao đổi giữa client và server, cấu trúc thông điệp, kiểu dữ liệu của thông điệp, cách tổ chức cơ sở dữ liệu (nếu có).
* Môi trường lập trình và các framework hỗ trợ để thực thi ứng dụng.
* Hướng dẫn sử dụng các tính năng chương trình.
* Bảng phân công công việc và cho biết rõ ràng ai làm việc gì cách rõ ràng. Không chia đều công việc hay cùng làm mọi việc.
* Các nguồn tài liệu tham khảo.

***Lưu ý:*** Trong báo cáo không dán các đoạn source code của chương trình. Mã chương trình chỉ trình bày nếu thật sự cần thiết và nếu cần minh họa cho các mô hình cài đặt hay các cơ chế đồng bộ (minh họa dạng mã giả, prototype hàm).

## Về vấn đáp:

* Chuẩn bị thiết bị, chương trình, báo cáo đầy đủ (không cần in).
* Trả lời các câu hỏi từ GV
* Trường hợp trả lời sai hoặc không trả lời được sẽ trừ trực tiếp điểm vào tổng điểm đồ án.

***Lưu ý:*** Tất cả thành viên của nhóm phải tham gia buổi vấn đáp. Thành viên vắng mặt sẽ xử lý theo quy định sau:

* Có phép (gửi email xin phép trước buổi vấn đáp): trừ điểm vấn đáp trực tiếp
* Không phép: 0 điểm toàn đồ án.

# **Đề bài**

## Đề 1:

Viết ứng dụng cho phép tra cứu thông tin giá cryptocurrency, theo mô hình client-server, sử dụng Socket với các chức năng như sau:

* Tra cứu danh sách các đồng tiền và giá qui đổi của đồng tiền tương ứng.
* Tra cứu 1 đồng tiền và giá qui đổi của đồng tiền đó.

Mô hình ứng dụng:

Client — Server — API

Trong đó:

* Ứng dụng desktop client được viết sử dụng thư viện Socket được cung cấp bởi ngôn ngữ lập trình. Client được sử dụng bởi người dùng, và sử dụng 2 tính năng tra cứu trên. Sinh viên tùy chọn giao diện của client (GUI hay Console).
* Ứng dụng server cũng được viết bằng thư viện Socket được cung cấp bởi ngôn ngữ lập trình để cung cấp 2 tính năng trên cho client có thể tra cứu. Sinh viên cũng tùy chọn giao diện của server, có thể chỉ cần Console là đủ.
* Ứng dụng server có một cơ sở dữ liệu về các cryptocurrency và giá của chúng. Dữ liệu này, sinh viên có thể tự xây dựng lấy (hard-coded, file (XML, JSON, CSV…), cơ sở dữ liệu quan hệ…), hoặc sinh viên có thể sử dụng thư viện bên thứ 3 tích hợp vào server để lấy dữ liệu từ API, hoặc sinh viên có thể kết hợp cả API và CSDL bằng cách lấy dữ liệu từ API và lưu vào CSDL. Lưu ý, nếu có sử dụng đến API thì có thể dùng thư viện bên thứ 3 ở phía ứng dụng server để tải dữ liệu từ API về, cụ thể là thư viện HTTP client để giao tiếp với API.
* API, sinh viên có thể tùy chọn, ví dụ: <https://coinmarketcap.com/api/documentation/v1/#>
* Sinh viên tự thiết kế kịch bản giao tiếp giữa client và server. Ví dụ, khi truy vấn toàn bộ các đồng tiền thì client có thể gửi lên server chuỗi "MARKET ALL", còn khi chỉ truy vấn 1 đồng tiền như BTC thì có thể gửi "MARKET BTC".
* Cơ sở dữ liệu chỉ cần hỗ trợ 10 đồng tiền.
* Hỗ trợ nhiều clients truy cập đồng thời đến server.

## Đề 2:

Viết ứng dụng cho phép tra cứu thông tin thời tiết, theo mô hình client-server, sử dụng Socket với các chức năng như sau:

* Tra cứu danh sách các thành phố trên thế giới và thời tiết của ngày hiện tại của thành phố tương ứng.
* Tra cứu 1 thành phố và thời tiết của thành phố đó.

Mô hình ứng dụng:

Client — Server — API

Trong đó:

* Ứng dụng desktop client được viết sử dụng thư viện Socket được cung cấp bởi ngôn ngữ lập trình. Client được sử dụng bởi người dùng, và sử dụng 2 tính năng tra cứu trên. Sinh viên tùy chọn giao diện của client (GUI hay Console).
* Ứng dụng server cũng được viết bằng thư viện Socket được cung cấp bởi ngôn ngữ lập trình để cung cấp 2 tính năng trên cho client có thể tra cứu. Sinh viên cũng tùy chọn giao diện của server, có thể chỉ cần Console là đủ.
* Ứng dụng server có một cơ sở dữ liệu về các thành phố và thời tiết của chúng. Dữ liệu này, sinh viên có thể tự xây dựng lấy (hard-coded, file (XML, JSON, CSV…), cơ sở dữ liệu quan hệ…), hoặc sinh viên có thể sử dụng thư viện bên thứ 3 tích hợp vào server để lấy dữ liệu từ API, hoặc sinh viên có thể kết hợp cả API và CSDL bằng cách lấy dữ liệu từ API và lưu vào CSDL. Lưu ý, nếu có sử dụng đến API thì có thể dùng thư viện bên thứ 3 ở phía ứng dụng server để tải dữ liệu từ API về, cụ thể là thư viện HTTP client để giao tiếp với API.
* API, sinh viên có thể tùy chọn, ví dụ: <https://www.weatherapi.com/docs/>
* Sinh viên tự thiết kế kịch bản giao tiếp giữa client và server. Ví dụ, khi truy vấn toàn bộ thành phố thì client gửi lên server chuỗi "WEATHER ALL", còn khi truy vấn một thành phố cụ thể như London thì có thể gửi chuỗi "WEATHER LONDON".
* Cơ sở dữ liệu chỉ cần hỗ trợ 10 thành phố.
* Hỗ trợ nhiều clients truy cập đồng thời đến server.

# **Thang điểm**

| Tính năng | Điểm | Ghi chú |
| --- | --- | --- |
| Truy vấn thông tin tất cả (đồng tiền, thành phố) | 2 |  |
| Truy vấn thông tin đến từng đồng tiền/thành phố cụ thể | 2 |  |
| Có sử dụng CSDL | 2 | Nếu hard-coded thì tối đa được 1 điểm. |
| Có sử dụng thư viện bên thứ 3 (API) để lấy thông tin thực tế | 2 |  |
| Hỗ trợ nhiều clients truy cập đồng thời đến server | 2 |  |