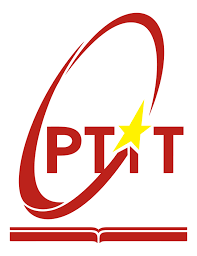
HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

**KHOA CÔNG NGHỆ THÔNG TIN**

**-----🙠🕮🙢 -----**



**BÀI TẬP LỚN**

**LẬP TRÌNH MẠNG**

Giảng viên: Nguyễn Hoàng Anh

Nhóm 05– Đề số 9

Danh sách sinh viên:

1. Nguyễn Hữu Ước – B15DCCN638
2. Nguyễn Văn Đạo – B15DCCN104
3. Nguyễn Mạnh Hùng – B15DCCN6252
4. Nguyễn Hoàng Việt Anh – B15DCCN032

**Hà Nội, Tháng 11/2017**

**GAME TRẮC NGHIỆM NHANH**

**THI ĐẤU ĐỐI KHÁNG ONLINE**

Contents

[**Danh sách thành viên và đóng góp: 3**](#_Toc530356030)

[**1. Java – Kiến trúc Swing và mô hình MVC 3**](#_Toc530356031)

[**1.1. Kiến trúc MVC 5**](#_Toc530356032)

[**1.2. Sự tương tác trong MVC 7**](#_Toc530356033)

[**1.3. MVC trong Swing………………………………………………………………………………7**](#_Toc530356034)

[**2. Mô tả hệ thống game 8**](#_Toc530356035)

[**3. Các chức năng chính của game. 9**](#_Toc530356036)

[**3.1. Chức năng người dung: 10**](#_Toc530356037)

[**3.2. Chức năng đăng ký, đăng nhập 10**](#_Toc530356038)

[**3.3. Quả trình đăng nhập 10**](#_Toc530356039)

[**3.4. Chức năng quản lý thông tin cá nhân: 10**](#_Toc530356040)

[**3.5. Chức năng thống kê Top User: 11**](#_Toc530356041)

[**3.6. Chức năng quản lý hộp thư, bạn bè 11**](#_Toc530356042)

[**3.7. Chức năng chơi Game 11**](#_Toc530356044)

[**3.8. Chức năng report hệ thống 12**](#_Toc530356045)

[**a. Thống kê danh sách người dùng đăng ký trong ngày 12**](#_Toc530356046)

[**b. Thống kê danh sách người dùng đang online 12**](#_Toc530356047)

[**c. Thống kê Top User 12**](#_Toc530356048)

[**4. Xây dựng biểu đồ lớp 13**](#_Toc530356049)

[**4.1. Xây dựng biểu đồ lớp cho Project 13**](#_Toc530356050)

[**4.2. Thiết kế Database 13**](#_Toc530356051)

[**4.3. Thiết kế Database client 14**](#_Toc530356052)

[**4.4. Database nganhangcauhoi 15**](#_Toc530356053)

[**5. Kết quả cho tới thời điểm hiện tại 16**](#_Toc530356054)

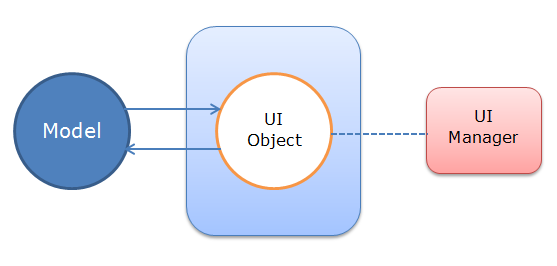
[**6. Kết luận 19**](#_Toc530356055)

[**- Những kết quả đạt được. 19**](#_Toc530356056)

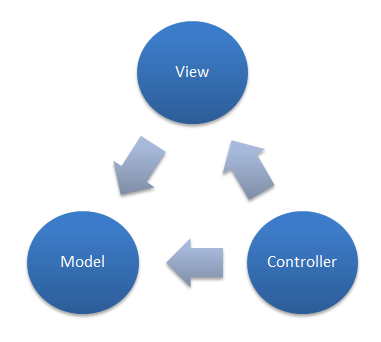
[**TÀI LIỆU THAM KHẢO 19**](#_Toc530356057)

# Danh sách thành viên và đóng góp:

1. Nguyễn Hữu Ước
2. Nguyễn Văn Đạo
3. Nguyễn Mạnh Hùng
4. Nguyễn Hoàng Việt Anh
5. **Java – Kiến trúc Swing và mô hình MVC**



Các component của Swing, bạn có thể nhận ra rằng chúng được thiết kế để tách riêng hai phần giao diện và dữ liệu. Cụ thể hơn các thành phần này được thiết kế dựa trên mô hình MVC (Model-View-Controller). Việc áp dụng mô hình này giúp cho việc quản lý dữ liệu và thay đổi giao diện của các component hiệu quả hơn.



–      Model: lưu trữ hoặc làm cầu nối với dữ liệu.

–      View: hiển thị giao diện và nhận các thông điệp người dùng. View có thể truy vấn dữ liệu từ Model để hiển thị lên màn hình.

–      Controller: làm trung gian chuyển các thông điệp người dùng từ View đến Model và cập nhật lại View khi cần thiết.

Trong Swing, theo quy mô của component, mô hình được thay đổi để phù hợp cho việc áp dụng cho các component.  Cụ thể phần View và Controller sẽ được gộp chung lại trong một

* 1. **Kiến trúc MVC**  
       
       
      MVC được biết đến như là một thiết kế giao diện người dùng hướng đối tượng khá tốt đã được đưa ra vào khoảng cuối những năm của thập kỷ 1970. Những thành phần được chia thành ba phần: một Model (mô hình), một View(hiển thị), một Controller (Điều khiển). Mỗi thành phần Swing dựa trên một phiên bản mới hơn của Thiết kế này. Trước khi thảo luận về MVC làm việc trong Swing, chúng ta cần hiểu nó được thiết kế ban đầu như thế nào.  
       
     **Model (Mô hình)**  
       
       
      Model chứa đựng trạng thái dữ liệu của mỗi thành phần. Có những mô hình khác nhau cho những thành phần khác nhau. Ví dụ, model của một scrollbar có thể chứa thông tin về vị trí hiện tại có thể điều chỉnh được "thumb", giá trị nhỏ nhất và giá trị lớn nhất của nó, và độ rộng của thumb (quan hệ với dãy giá trị). Một menu lại khác, nó có thể đơn giản chỉ chứa một danh sách các menu item mà người dùng có thể chọn.  
     Model chịu trách nhiệm nắm giữ tất cả các thể hiện của trạng thái của thành phần. Ví dụ, những giá trị như trạng thái nhấn hoặc không nhấn của một button, và dữ liệu ký tự của một thành phần text cũng như thông tin về việc nó có cấu trúc như thế nào. Một Model có thể chịu trách nhiệm với việc giao tiếp một cách gián tiếp với View và Controller. Gián tiếp có nghĩa là Model không biết View và Controller của nó - nó không duy trì hoặc tìm kiếm các dẫn xuất đến chúng. Thay vào đó, Model sẽ gởi đi những khai báo hoặc những broadcast (những gì chúng ta biết là những sự kiện).  
       
     **View (Hiển thị)**  
      View liên quan đến việc bạn nhìn thấy thành phần trên màn hình như thế nào. Đây chính là thành phần "look". Một ví dụ dễ thấy hiển thị có thể khác nhau như thế nào, hãy nhìn vào một ứng dụng Windows trong hai platform GUI khác nhau. Hầu hết frame của cửa sổ có một thanh tiêu đề nằm trên đỉnh của cửa sổ. Tuy nhiên, thanh tiêu đề có thể có một nút close ở phía bên trái như trong platform MacOS, hoặc nó có thể có một nút close ở phía bên phải như trong platform Windows. Đó là những ví dụ những kiểu hiển thị khác nhau của cùng một kiểu đối tượng cửa sổ.  
     View xác định việc hiện thị trực quan của Model của thành phần. View chịu trách nhiệm giữ việc thay thế nó trên màn hình luôn được cập nhật, nó có thể nhận những thông điệp gián tiếp từ Model hoặc những thông điệp từ Controller.  
       
     **Controller (Điều khiển)**  
      Controller là phần kết nối với giao tiếp người dùng để ra lệnh cho thành phần tương tác như thế nào với những sự kiện. Những sự kiện đến từ nhiều trường hợp - ví dụ, một click chuột, sự kiện bàn phím bắt đầu một lệnh menu cụ thể, hoặc thậm chí một lệnh vẽ lại một phần của màn hình.  
      Controller chịu trách nhiệm xác định thành phần nào tác động lại bất kỳ sự kiện nào từ thiết bị input như bàn phím và chuột. Controller chính là "feel" của thành phần, và nó cũng xác định hành động nào được thực thi khi thành phần được sử dụng. Controller có thể nhận thông điệp từ View, và những thông điệp gián tiếp từ Model.  
       
      Hình dưới chỉ ra Model, View, Controller làm việc với nhau như thế nào để tạo ra một scrollbar. Model nắm giữ những thông tin về giá trị min và max. View xác định chính xác vẽ scrollbar như thế nào và vẽ ở vị trí nào. Cuối cùng, Controller chịu trách nhiệm xử lý các sự kiện chuột. Kết quả là một scrollbar mang đầy đủ chức năng của MVC
  2. **Sự tương tác trong MVC**  
     Với MVC, mỗi yếu tố Model, View, Controller yêu cầu những dịch vụ của những yếu tố khác để giữ bản thân nó tiếp tục được cập nhật.  
     Ví dụ, giả sử rằng chúng ta có một checkbox được check trong giao diện. Nếu Controller xác định người dùng thực hiện một click chuột, nó có thể gởi một thông điệp cho View. Nếu View xác định rằng click xảy ra trên checkbox, nó gởi một thông điệp cho Controller.Controller cập nhật lại Model dựa trên sự kiện nhận được. Model sau khi cập nhật bản thân nó và thông báo một thông điệp, sẽ được nhận bởi View, để thông báo với View rằng nó phải cập nhật lại bản thân nó dựa trên trạng thái mới của Model. Tại đây có một vòng quay liên tục được lặp lại. Theo cách này, một Model không hạn chế một View hoặc Controller cụ thể, điều này cho phép chúng ta có nhiều View và Controller khác nhau thực thi một Model.

​

## MVC trong Swing Swing thật sự sử dụng một kiểu riêng đơn giản của thiết kế MVC gọi là Model - Delegate. Thiết kế này kết hợp View và Controller vào trong một yếu tố đơn là UI delegate, sẽ vẽ thành phần trên màn hình và xử lý những sự kiện GUI. Kết hợp khả năng đồ họa và xử lý sự kiện là một chút dễ dàng trong Java, khi hầu hết việc xử lý sự kiện đã được thực hiện cẩn thận trong AWT. Như bạn có thể mong đợi, việc giao tiếp giữa Model và UI delegate sau khi kết hợp được chỉ ra ở hình dưới.

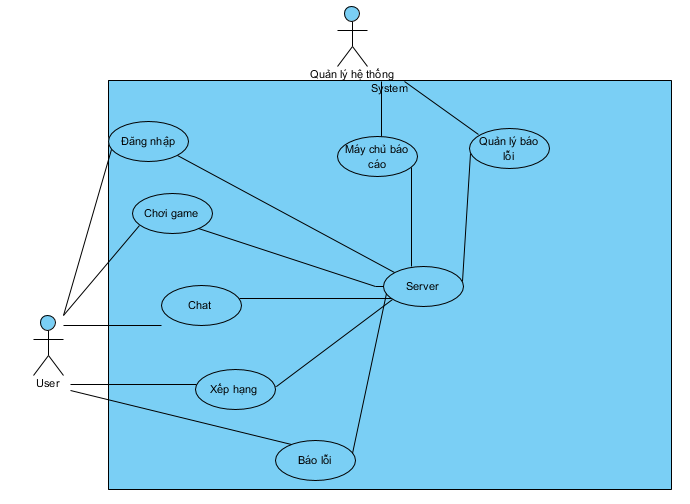
​

Hãy nhìn lại: mỗi thành phần Swing chứa một Model và một UI delegate. Model chịu trách nhiệm cho việc nắm giữ thông tin về trạng thái của thành phần. UI delegate chịu trách nhiệm nắm giữ thông tin về việc vẽ thành phần trên màn hình như thế nào. Thêm vào đó, UI delegate (tập hợp với AWT) tương tác lại với những sự kiện riêng biệt để truyền lại thông qua thành phần.  
  
Chú ý việc chia lại của Model và UI delegate trong thiết kế MVC là một sự thuận lợi cực kỳ. Một diện mạo duy nhất của kiến trúc MVC là khả năng hiển thị nhiều tầng (multi tie) cho một cùng Model. Ví dụ, nếu bạn muốn hiển thị cùng dữ liệu trong một biểu đồ và trong một bảng, bạn có thể dựa trên việc hiển thị của hai thành phần trên cùng một mô hình dữ liệu duy nhất. Theo cách này, nếu dữ liệu cần được thay đổi, bạn có thể chỉ sửa nó ở một nơi - View sẽ cập nhật lại bản thân chúng tương ứng. Theo cùng cách như thế, việc chia ra những thành phần đại diện từ Model sẽ cho người dùng thêm tiện lợi của việc chọn lựa một thành phần nào để xem mà không hề ảnh hưởng đến dữ liệu của nó. Bằng việc sử dụng hướng đi này, trong kết hợp với thiết kế lightweight, Swing có thể cung cấp mỗi thành phần với những cảm quan pluggable của nó.

1. **Mô tả hệ thống game**

* Hệ thống có một server và nhiều client. Server lưu toàn bộ thông tin và dữ liệu.
* Để chơi, người chơi phải login vào tài khoản của mình từ một máy client. Sau khi login thành công, giao diện hiện lên một danh sách các người chơi đang online, mỗi người chơi có các thông tin: tên, tổng số điểm hiện có của người chơi, trạng thái (hoặc đang bận nếu đang chơi với người khác, hoặc đang rỗi nếu không chơi với ai)
* Muốn mời (thách đấu) ai thì người chơi click vào tên của đối thủ đó trong danh sách online.
* Khi bị thách đấu, người chơi có thể chấp nhận (OK), hoặc từ chối (Reject).
* Khi chấp nhận, 2 người chơi sẽ vào chơi với nhau, và server sẽ làm trọng tài. Giao diện chơi gồm một danh sách N câu hỏi theo dạng trắc nghiệm, ô thời gian và nút thoát.

Mô hình Use Case của hệ thống



Hình 1: Sơ đồ use case của hệ thống

1. Hệ thống gồm có: Người dùng đăng nhập, người chơi, hàng và người quản trị hệ thống. Người dùng được cấp sẵn Account và password để đăng nhập sau đó chơi

2. Người chơi: Là người dùng đã có acc. Người chơi có thể đăng nhập vào hệ thống, chat với bạn bè, quản lý hộp tin nhắn, gửi tin nhắn, chơi

3. Quản trị hệ thống: Đây là bộ phận quan trọng nhất của hệ thống. Quản trị hệ thống có 2 nhiệm vụ chính: quản lý server và quản lý kênh báo lỗi.

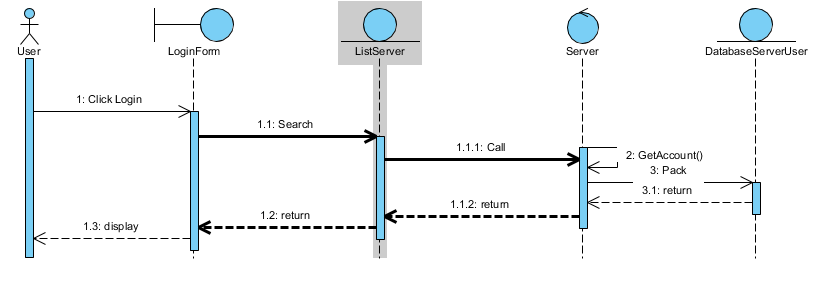
4. Quản lý server: theo dõi tình hình hệ thống thông qua hệ thống report (sốlượng người online, CPU server, các lỗi phát sinh của hệ thống, tình trạng database,…). Người quản trị server còn có nhiệm vụ thay đổi các thông số của hệ thống khi cần.

5. Quản lý báo lỗi: Người quản trị hệ thống sẽ phải cập nhật thường xuyênnhững lỗi người dung phản ảnh qua hệ thống báo lỗi, từ đó kiểm tra hệ thống vàthông báo cho những người có trách nhiệm để sửa lỗi hệ thống

1. **Các chức năng chính của game.** 
   1. **Chức năng người dung:**

Chức năng người dùng được phân chia làm các chức năng nhờ:

* Đăng nhập
* Chức năng chơi game
* Chức năng quản lý thông tin cá nhân
* Chức năng thống kê Top User
* Chức năng quản lý hộp thư, bạn bè.
* Chức năng chat, gửi tin nhắn
* Chức năng thể hiện cảm xúc, tương tác với người khác
* Chức năng báo lỗi hệ thống
  1. **Chức năng đăng ký, đăng nhập**
* Nhiệm vụ ban đầu của người dùng đăng nhập. Do tài khoản đã được cấp phát từ trước
* Các thông tin khác người dùng có thể update sau khi đã đăng nhập thành công vào hệ thống. Bảo mật thông tin chính là yếu tố rất quan trọng đem lại sự tin tưởng của người dùng đối với hệ thống
  1. **Quả trình đăng nhập**

****

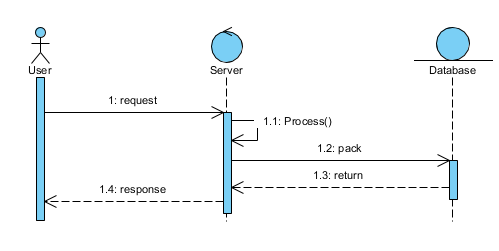
Hình 2: Sơ đồ tuần tự cho quá trình đăng nhập

Sẽ kết nối tới hệ thống database để kiểm tra username, password của người sử dụng và trả về cho người dùng là họ đã đăng nhập thành công hay thất bại. Người dùng có thể chọn chức năng ghi nhớ tài khoản để ở những lần đăng nhập sau người dùng sẽ bỏ qua được bước nhập username, password.

## Chức năng quản lý thông tin cá nhân:

1. Tên tài khoản
2. Danh hiệu
3. Số trận thắng
4. Số trận thua
5. Số lần hòa
6. Số điểm
   1. **Chức năng thống kê Top User:**

Với một hệ thống game Online, chức năng quan trọng không thể thiếu đó là top game thủ. Chức năng này có tác dụng kích thích người chơi đua top để có thứhạng cao nhất. Top user bao gồm: Top Trận thắng, Top điểm cao, Top trả lời được nhiều câu hỏi nhất.



Hình 3: Sơ đồ tuần tự cho chức năng TopUser

Quá trình lấy thông tin được thực hiện qua 4 bước: User gửi request lên server, server kết nối đến database để lấy thông tin, server xử lý thông tin nhận được rồi gửi trả về cho người dùng.

## Chức năng quản lý hộp thư, bạn bè

Ý tưởng về chức năng hộp thư, bạn bè được thiết kế .Với chức năng này người dùng có thể lấy được danh sách bạn bè, có thể biết ai đang online, offline. Có thể chat với bạn bè, hay có thể nhắn tin offline cho người đó. Người chơi chỉ có thể kết bạn với tối đa 5000 người, hộp tin nhắn chỉ có thể hiện thị không giới hạn tin nhắn.

## Chức năng chơi Game

Chức năng quan trọng nhất của hệ thống Game là người dùng có thể chơi các trò chơi. Đây là nơi người chơi có sự trao đổi, tương tác với nhau nhiều nhất. Chức năng này đặc biệt quan trọng, đòi hỏi hệ thống phải xử lý tốt, tránh trường hợp bị lag, hack, …

Để vào chơi game, người chơi sẽ phải thực hiện các thao tác: đăng nhập, chọn đề muốn chơi, chọn đối thủ thách đấu, cuối cùng là chơi game.Chức năng báo lỗi hệ thống Đối với bất kỳ hệ thống nào, việc phát sinh lỗi là không thể tránh khỏi. Với hàng ngàn người dùng sử dụng phần mềm, đây chính là đối tượng gần với hệ thống nhất. Khi gặp lỗi, người dùng có thể vào mục báo lỗi để gửi lỗi lên server, server sẽ tiếp nhận lỗi và ghi vào kênh báo lỗi để người quản trị được biết. Ngoài báo lỗi, người dùng có thể đóng góp ý kiến của mình về hệ thống thông qua chức năng báo lỗi này. Khi đó ứng dụng sẽ chuyển người chơi đến trang web chứa chơi game để người dùng chơi.

## 3.8. Chức năng report hệ thống

### a. Thống kê danh sách người dùng đăng ký trong ngày

Chức năng này dùng để liệt kê danh sách những tài khoản đã đăng ký nick trong ngày hiện tại. Ngoài ra, người quản trị còn có thể lựa chọn xem danh sách người dùng đã đăng ký nick của những ngày trước đó. Các thông tin thống kê trong danh sách người dùng bao gồm: username, giới tính, số điện thoại, thắng, thua, bạn bè, ngày đăng ký tài khoản, trạng thái Online, server hiện tại đang kết nối đến, thời điểm đăng nhập lần cuối cùng. Ở mục này có thêm chức năng search tài khoản người dùng theo username. Chức năng này giúp người quản trị hệ thống dễ dàng quản lý được người sử dụng,đặc biệt là khi có khiếu nại hay vấn đề gì đó không bình thường từ phía người sử dụng.

### b. Thống kê danh sách người dùng đang online

Chức năng này giúp người quản trị biết được tại thời điểm hiện tại có baonhiêu người đang online và danh sách những tài khoản đang online. Thống kê danh sách giao dịch trong ngày Mục này sẽ liệt kê danh sách người dùng đã nạp tiền ở ngày hiện tại. Ngườiquản trị cũng có thể xem giao dịch của những ngày trước đó.

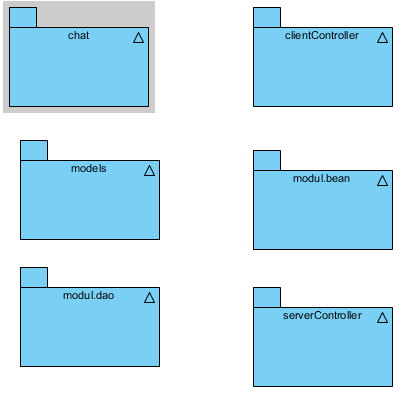
### c. Thống kê Top User

Thống kê Top User Mục này sẽ thống kê top người dùng theo thứ tự giảm dần từ trên xuống dựa theo tiền và điểm kinh nghiệm. Cụ thể có các loại Top cần thống kê bao gồm: Top cao thủ, top thất thủ. Ở ứng dụng ở phía client, người dùng chỉ xem được Top này với số lượngthống kê là 10 người. Còn ở bảng thống kê này, người quản trị có thể có thể theo dõi danh sách thống kê bao gồm tất cả người dùng của hệ thống được sắp xếp theo thứ tự giảm dần

# 4. Xây dựng biểu đồ lớp

## 4.1. Xây dựng biểu đồ lớp cho Project

Server được xây dựng gồm 4 thành phần chính được chia thành 4 packet:chat, clientController, models, modul.bean, modul.dao, serverController

****

Hình 4: Biểu đồ lớp cho project

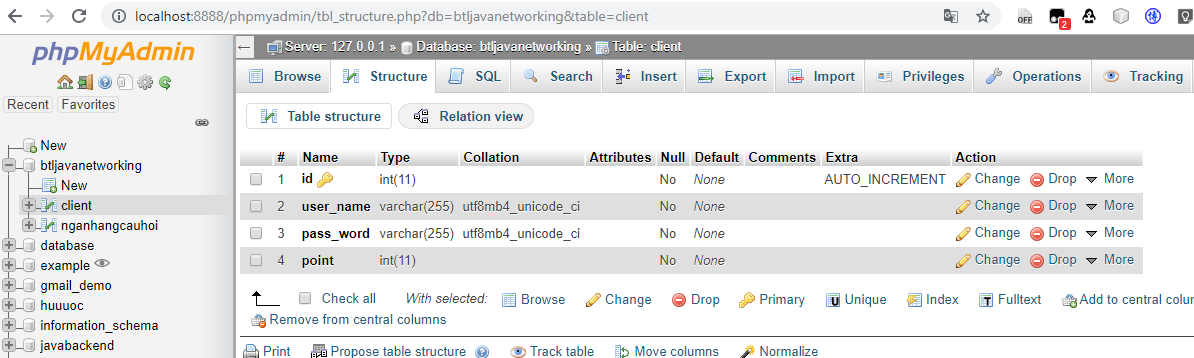
## 4.2. Thiết kế Database

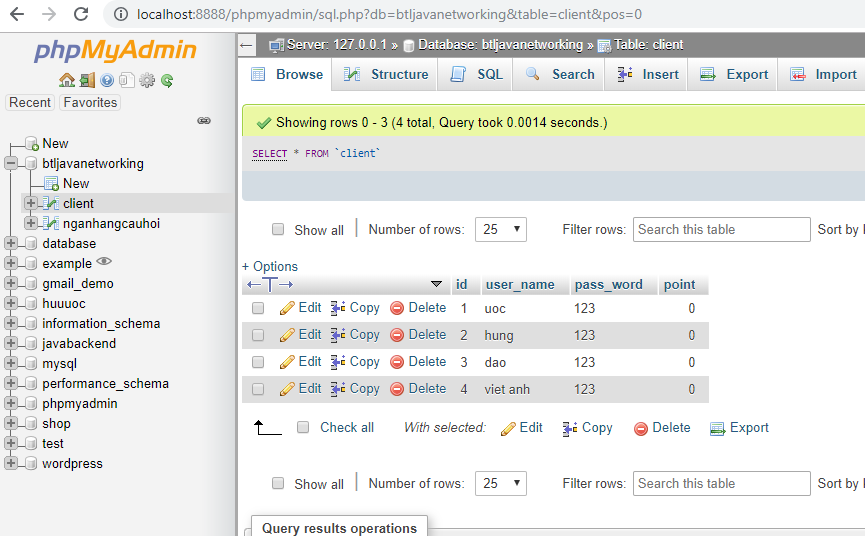
Trong hệ thống chia làm 2 loại database:

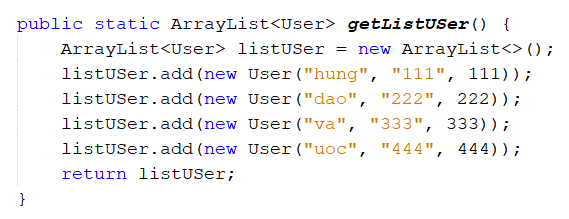
Database client: lưu trữ toàn bộ thông tin liên quan đến tài khoản người dùng.

Database nganhangcauhoit: lưu trữ thông tin về nội dung của game gồm các gói câu hỏi theo chủ đề.

**4.3. Thiết kế Database client**

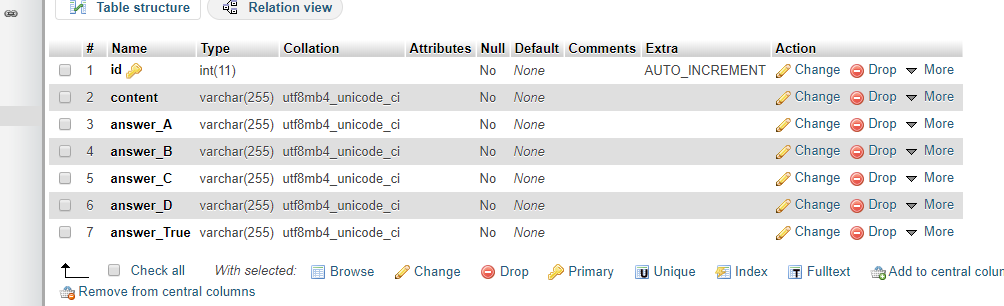


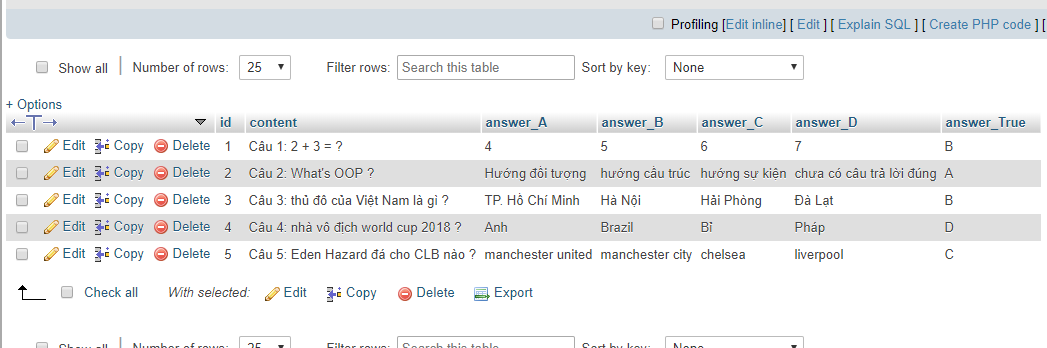




Bảng mô tả các table trong database client

## 4.4. Database nganhangcauhoi

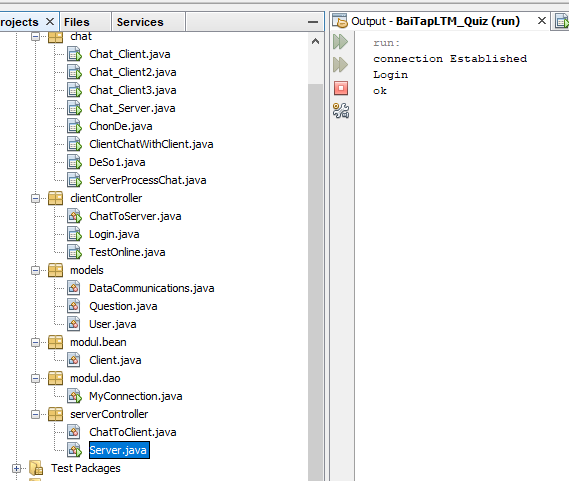


****

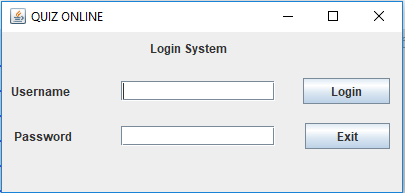
|  |  |  |
| --- | --- | --- |
| **STT** | **Bảng** | **Chức năng** |
| 1 | Question | Lưu trữ thông tin câu hỏi của game.  ID: số thứ tự  content: nội dung câu hỏi  case\_A: đáp án A  case\_B: đáp án B  case\_C: đáp án C  case\_D: đáp án D  case\_True: đáp án đúng |

1. **Kết quả cho tới thời điểm hiện tại**

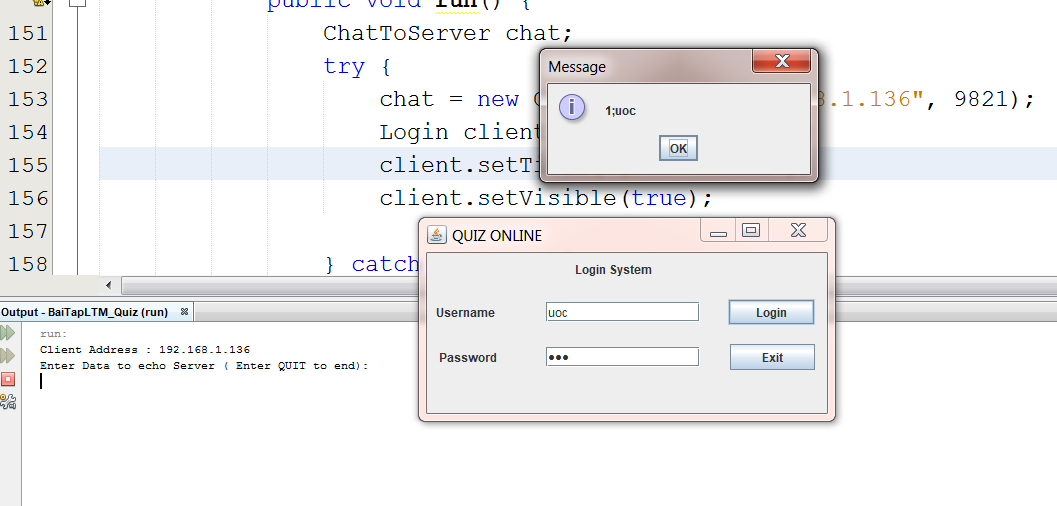
* Bật server.class lên



* Login nhanh

****

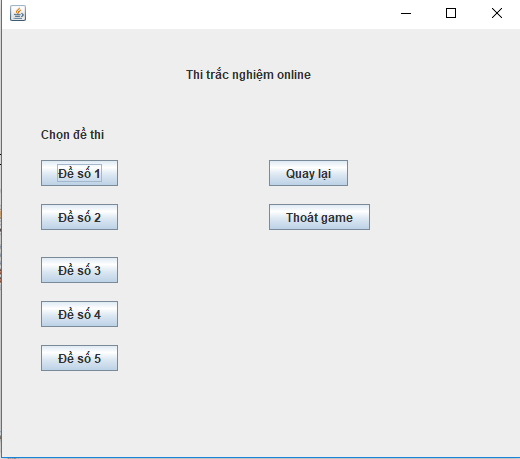
* Đã login thành công



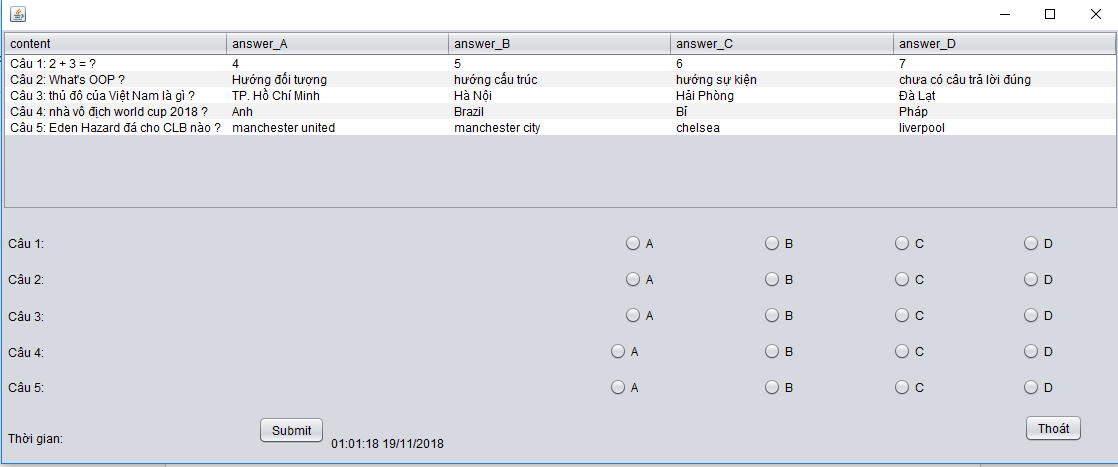
* Giao diện bắt đầu chơi game & hiển thị danh sách người chơi đang online



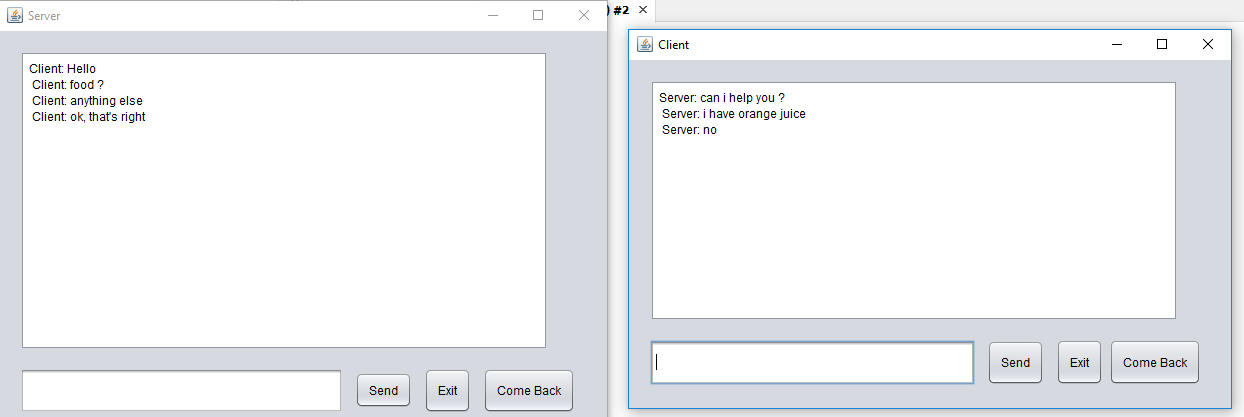
* Giao diện chọn đề



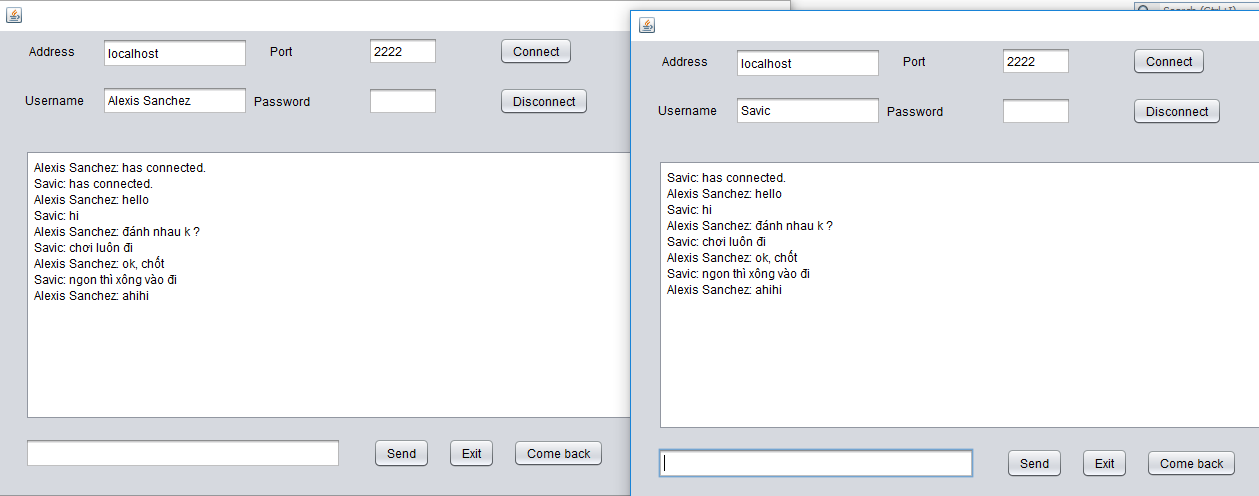
**-** Giao diện vào đề thi



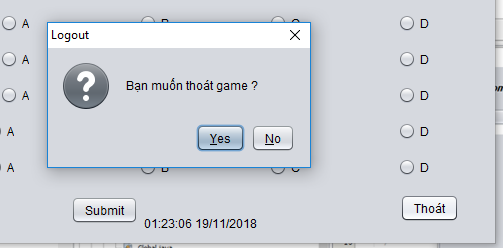
* Giao diện chat giữa client với server



* Giao diện chat giữa client với client



* Giao diện thoát game



**6. Kết luận**

**- Những kết quả đạt được.**

Sau khi làm xong bài tập “Game trắc nghiệm nhanh thi đấu đối kháng online”

**- Lý thuyết:**

1.Tìm hiểu, nghiên cứu về game trả lời câu hỏi trắc nghiệm

2.Tìm hiểu, nghiên cứu mô hình Client server để áp dụng vào game

3.Nghiên cứu kỹ thuật lập trình Socket trong Java để trao đổi dữ liệu giữa Client và Server

**-Chương trình:**

Tạo ra một game trắc nghiệm vui cho người dùng giải trí và trí tuệ.

Sử dụng Java swing để thiết kế game và dùng java để kết nối client với server

**-Hạn chế:**

Sản phẩm vẫn còn nhiều hạn chế về chất lượng sản phẩm chưa tốt. Các chức năng người dùng vẫn còn thiếu, chưa mang lại sự hấp dẫn cho người dùng.

# TÀI LIỆU THAM KHẢO

1. Giáo trình mạng máy tính ptit: <http://dut.khoaluan.vn/tai-lieu_bai-giang-mang-may-tinh-ths-nguyen-xuan-anh_1013908>
2. Giáo trình phân tích thiết kế hệ thống thông tin ptit: <http://home.vinhuni.edu.vn/cuongvcc/wp-content/uploads/sites/109/2017/10/Tran-Dinh-Que-Giao-trinh-phan-tich-he-thong-thong-tin.pdf>
3. Tài liệu lập trình mạng Nguyễn Mạnh Hùng PTIT
4. Tài liệu oracle: <https://docs.oracle.com/javase/tutorial/networking/index.html>
5. Tutorial: <https://www.javatpoint.com/java-networking>