```java
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package company_server;

import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.net.ServerSocket;
import java.net.Socket;

import data_bean.CSVBean;
import data_bean.DatabaseBean;
import data_bean.Navire;
import data_bean.Pair;
import data_bean.Reservation;
import data_bean.Traversee;
import java.io.File;
import java.sql.SQLException;

/**
 *
 * @author rapha
 */
public class CompanyServer {
    public static void main(String[] args)
            throws IOException, ClassNotFoundException, InstantiationException,
            IllegalAccessException, SQLException
    {
        ServerSocket server_sock = new ServerSocket(Config.COMPANY_PORT);

        DatabaseBean data = new DatabaseBean();
        System.out.println("Connecté à la base de données");

        for (;;) {
            System.out.println("En attente d'un nouveau client");
            Socket sock = server_sock.accept();
            System.out.println("Nouveau client");

            new ServerThread(sock, data).start();
        }
    }
}

class ServerThread extends Thread {
    private Socket _sock;
    private DatabaseBean _data;

    public ServerThread(Socket sock, DatabaseBean data)
    {
        this._sock = sock;
        this._data = data;
    }

    @Override
    public void run()
    {
        try {
            ObjectInputStream in = new ObjectInputStream(
                this._sock.getInputStream()
            );
            ObjectOutputStream out = new ObjectOutputStream(
                this._sock.getOutputStream()
            );

            Protocol query = (Protocol) in.readObject();
```

```
            if (query instanceof Login) {
                Login l = (Login) query;
                if (isValidAgent(l.getName(), l.getPassword())) {
                    out.writeObject(new Ack());
                    out.flush();
                    this.manageClient(in, out);
                } else {
                    out.writeObject(new Fail());
                    out.flush();
                }
            }
        } catch (Exception e) {
            System.err.println("Fermeture de la connexion: ");
            e.printStackTrace();
        }

        try {
            this._sock.close();
        } catch (IOException e) { }
    }

    private void manageClient(ObjectInputStream in, ObjectOutputStream out)
            throws IOException, ClassNotFoundException, SQLException
    {
        for (;;) {
            Protocol query = (Protocol) in.readObject();
            if (query instanceof VerifBooking) {
                // Validation du checkin
                VerifBooking vb = (VerifBooking) query;
                Reservation r = this._data.getReservation(
                    vb.getCode_reservation()
                );

                if (r != null && !r.isCheckin()) {
                    this._data.validateCheckin(r.getId());

                    this._data.ajoutFile(
                        r.getTraversee(), r.getVoyageur(), null
                    );

                    System.out.print("OK Ack");
                    out.writeObject(new Ack());
                    out.flush();
                } else {
                    System.out.print("OK FAIL");
                    out.writeObject(new Fail());
                    out.flush();
                }
            } else if (query instanceof BuyTicket)  {
                // Achat d'un ticket
                BuyTicket bt = (BuyTicket) query;

                Pair<Traversee, Navire> t = this._data.prochaineTraversee();
                if (t != null) {
                    int voyageur_id = this._data.achatTicket(
                        bt.getNom_conducteur(), t.getFirst().getId()
                    );

                    this._data.ajoutFile(
                        t.getFirst().getId(), voyageur_id,
                        bt.getImmatriculation()
                    );

                    out.writeObject(
                        new AckBuyTicket(
                            t.getSecond().getNom(),
                            t.getFirst().getDateDepart(), voyageur_id
```

```
                        )
                    );
                    out.flush();
                } else {
                    // Plus de place pour la traversée ou pas de traverséee
                    // prévue.
                    out.writeObject(new Fail());
                    out.flush();
                }
            }
        }
    }

    private static boolean isValidAgent(String name, String password)
            throws FileNotFoundException, IOException
    {
        for (String[] l : CSVBean.loadCSV(new File("agents.csv"))) {
            if (name.equals(l[0]) && password.equals(l[1])) {
                return true;
            }
        }
        return false;
    }
}
```