```
package modules.user.register;

import java.io.*;
import java.util.*;
import java.net.*;

/**
 * Send Request.java
 * This file replaces the send request shell scripts. This file will call
 * The ACL2 functions that generate the XML request to send to the server.
 * Once a request has been made by ACL2, we will loop through all the request
 * and send to the Server using port 20001
 * Original Author Adam Ghodratnama
 * @author Wesley R. Howell
 *
 */

public class SendRequest {

    public final static String OUTPATH = "store/user/requests/register/";
    public final static String INPATH = "incoming/email";

    public static void sendRequest (String name, String domain, String password){
        String unique = (new Date().toString()) +"_1";
        unique.replace(' ', '_');
        unique.replace(':', '_');
        //Create request using ACL2
        String script = "(in-package \"ACL2\")(include-book \"modules/user/register/create-user-request\"" ↙
    +
                " :uncertified-okp t) (createRequest '(\""+domain+"\" \""+name+"\" \""+password+"\") \""+ ↙
    unique+"\" state)";

        try{
        //Run on ACL2
        // Initialize ACL2 and dump its output to the log
        System.out.println("Executing ACL2 runtime for Email Generation...");
        ProcessBuilder processBuilder = new ProcessBuilder("acl2");
        File log = new File("logs/acl2_log_request.txt");
        processBuilder.redirectErrorStream(true);
        processBuilder.redirectOutput(log);

        Process process;

        process = processBuilder.start();

        PrintWriter procIn = new PrintWriter(process.getOutputStream());

        // Write the ACL2 to the process, close ACL2
        procIn.println(script);
        procIn.println("(good-bye)");
        procIn.flush();
        procIn.close();

        } catch(IOException e) {
            e.printStackTrace();
        }


       try{
           Thread.sleep(3000);
       } catch (InterruptedException e){
            e.printStackTrace();
       }


        //Send to the server.
        Socket server = null;
        PrintWriter out = null;
        BufferedReader in = null;

        try {
            System.out.println("Opening socket...");
            server = new Socket("localhost", 20003);
            System.out.println("Connection successful!");
            out = new PrintWriter(server.getOutputStream(), true);
            in = new BufferedReader(new InputStreamReader(server.getInputStream()));

            //Open directory and get request and send it.

            File folder = new File(OUTPATH);
            File[] listOfFiles = folder.listFiles();

            for (File f : listOfFiles){
                if(f.isFile() && !f.isHidden()){
                    BufferedReader reader = null;

                    try {
                        reader = new BufferedReader(new FileReader (f));
                        String line = null;
                        try {
                            while ((line = reader.readLine()) != null){
                                System.out.println(line);
                                out.println(line);
                            }
                        } catch (IOException e) {
                            // TODO Auto-generated catch block
                            e.printStackTrace();
                        }

                    } catch (FileNotFoundException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                    }
                }
            }


            out.flush();
            out.close();
            in.close();
            server.close();

        } catch(Exception e) {
            e.printStackTrace();
        }

    }
}
```