Note: The following script runs the java version as 1.8, which is for reference only, and should not be used in an online environment!

```
<dependency>
 <groupId>com.squareup.okhttp3</groupId>
 <artifactId>okhttp</artifactId>
 <version>4.2.2</version>
</dependency>
```

```
import okhttp3.*;
import java.nio.charset.StandardCharsets;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.util.*;
import java.util.concurrent.TimeUnit;

public class Demo {

    private static String md5(String str) {
        String result;
        MessageDigest md5 = null;
        try {
            md5 = MessageDigest.getInstance("MD5");
        } catch (NoSuchAlgorithmException e) {
            System.out.println(e);
        }
        md5.update((str).getBytes(StandardCharsets.UTF_8));
        byte[] b = md5.digest();
        int i;
        StringBuilder buf = new StringBuilder();
        for (byte value : b) {
            i = value;
            if (i < 0) {
                i += 256;
            }
            if (i < 16) {
                buf.append("0");
            }
            buf.append(Integer.toHexString(i));
        }
        result = buf.toString();
        return result;
    }

    private static OkHttpClient getOkHttpClient() {
        return new
OkHttpClient().newBuilder().retryOnConnectionFailure(false).connectTimeout(5,
TimeUnit.SECONDS).readTimeout(10, TimeUnit.SECONDS)
                .callTimeout(10, TimeUnit.SECONDS).writeTimeout(10,
TimeUnit.SECONDS).connectionPool(new ConnectionPool(10, 5, TimeUnit.MINUTES))
                .build();
    }
```

```java
    /**
     *
     * @param params
     * @param secret
     * @return String
     */
    private static String getToken(Map<String, String> params, String secret) {
        TreeMap<String, String> treeMap = new TreeMap<>();
        treeMap.putAll(params);
        StringBuffer stringBuffer = new StringBuffer();
        treeMap.forEach((k, v) -> {
            stringBuffer.append(k);
            stringBuffer.append("=");
            stringBuffer.append(v);
            stringBuffer.append("&");
        });
        stringBuffer.append("secret");
        stringBuffer.append("=");
        stringBuffer.append(secret);
        return md5(stringBuffer.toString());
    }

    public static void main(String[] args) {
        Response response = null;
        try {
            // appkey Need to be replaced with the official parameters provided
by the service provider
            final String APP_KEY = "xxxxxx";
            // secret Need to be replaced with the official parameters provided
by the service provider
            final String SECRET = "xxxxxxx";
            // The server interface address needs to be replaced with a real and
effective address. It will be provided in the document
            final String SERVER_URL =
"http://127.0.0.1:8091/api/data/highRiskList/handle";
            // Request parameter initialization
            Map<String, String> params = new HashMap<String, String>();
            params.put("appKey", APP_KEY);
            params.put("name", "YULIALITWTIUASRTYUY");
            params.put("phone", "81258177475");
            params.put("idNumber", "3578171807900512");
            // set token
            params.put("token", getToken(params, SECRET));
            System.out.println("Request service parameters:" +
params.toString());
            FormBody.Builder builder = new FormBody.Builder();
            params.forEach(builder::add);
            Request request = new
Request.Builder().url(SERVER_URL).post(builder.build()).build();
            Call call = getOkHttpClient().newCall(request);
            response = call.execute();
            String value = Objects.requireNonNull(response.body()).string();
            System.out.println("The server returns the result:" + value);
        } catch (Exception ex) {
            System.out.println("Request error message:" + ex.getMessage());
        } finally {
            if (null != response)
                response.close();
```

```
        }
    }
}
```