

# 电子商务与安全

## 实验指导书

班级：1804201

姓名：向乾龙

## 实验报告的内容

### 1. 题目

描述每个实验的内容是什么。

### 2. 需求分析、程序数据流与功能图

用 E-R 图描述数据库的模式设计及每个关系模式的建立；描述数据字典及功能图；每个事件、函数或过程的头和规格说明；列出每个过程或函数所调用和被调用的过程或函数，也可以通过调用关系图表示。主要算法的框架。

### 3. 调试报告

调试过程中遇到的主要问题是如何解决的；对设计和编码的回顾讨论和分析；改进设想；经验和体会等。

### 4. 源程序清单和结果

源程序要加注释，要有测试数据及结果。

# 实验 1： 掌握基于 WEB 的数据库连接编程及电子商务安全的应用（4 学时）

## 一、实验目的

本次实验的主要目的和内容：

1. 熟悉加密技术在电子商务安全中的应用；
2. 熟悉 VBScript 语言、HTML 语言等面向对象语言网站交互功能设计；
3. 掌握 ASP 的 ADO 接口访问 SQL 数据库连接或者其他基于 WEB 语言，数据库连接可以选择 ODBC 方式或者驱动程序方式。
4. 掌握一种基于 WEB 语言结合数据库完成数据库连接并实现用户登录功能的设计；

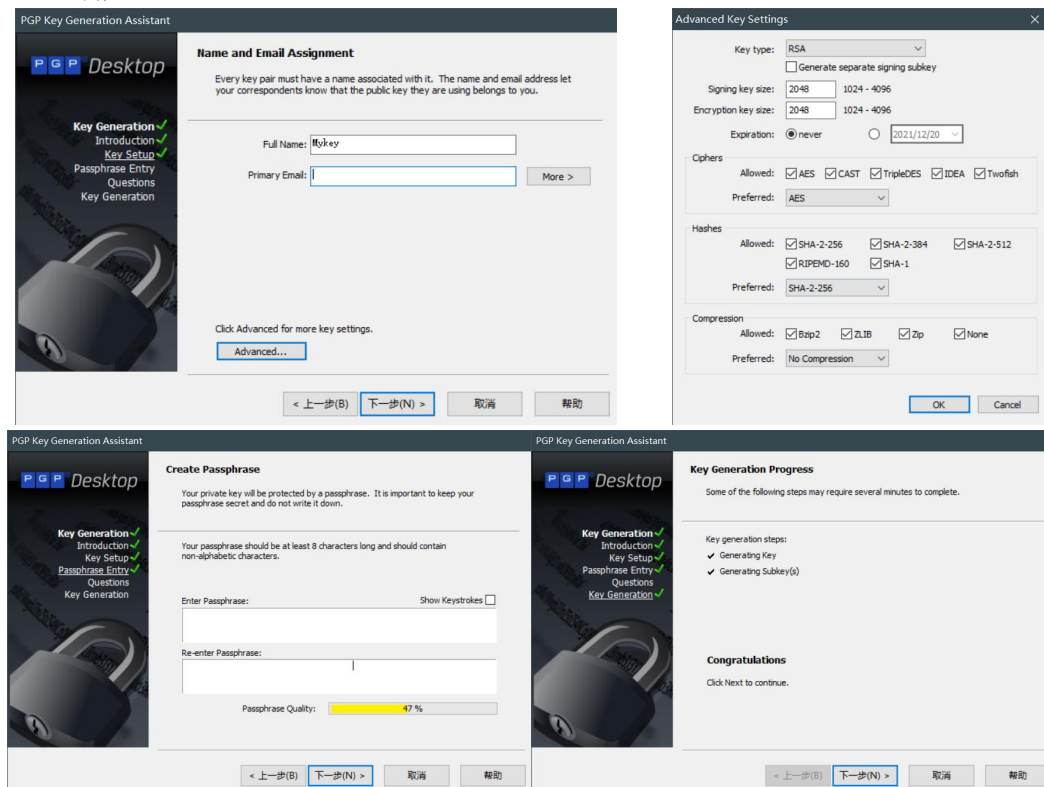
## 二、实验要求

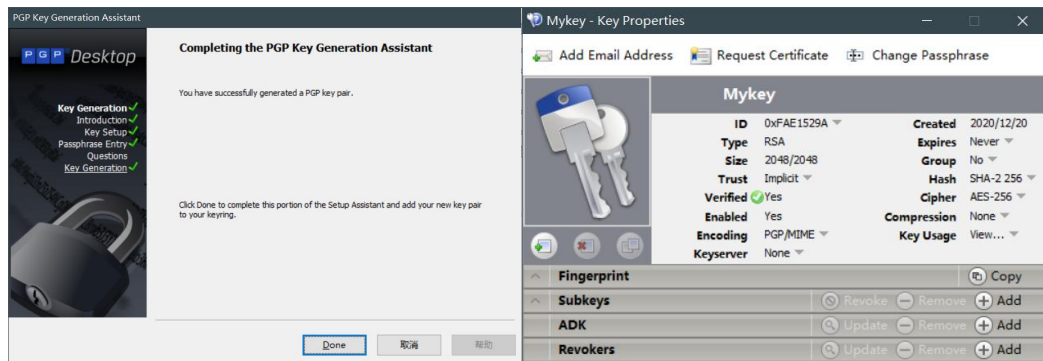
1. 要求学生独立完成实验内容；
2. 按照实验步骤完成实验后，撰写报告内容，并对操作结果进行截图。

## 三、实验内容、实验结果与主要代码

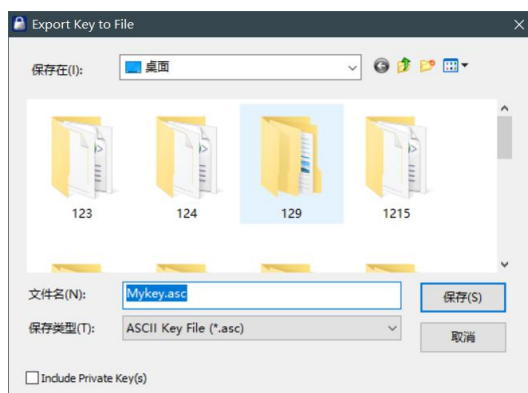
1. 下载 PGP 软件，运行其安全功能，并对具有的安全功能运行结果进行截图并解释安全功能作用及操作方法。

### 1) 密钥产生：

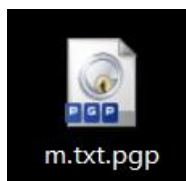
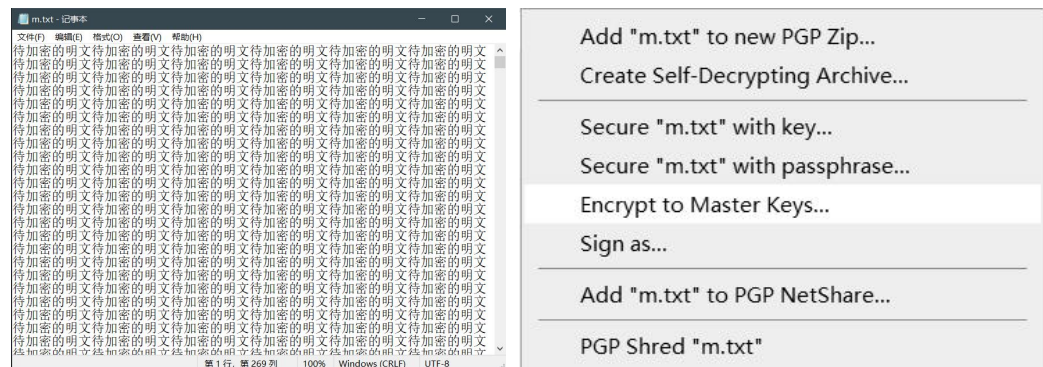




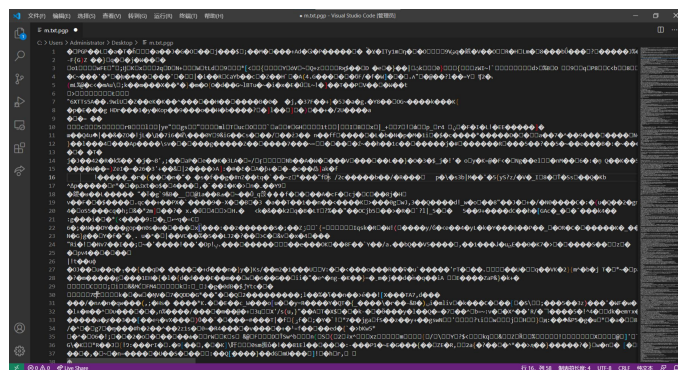
2) 导出公钥发送给服务器或者需要进行交流的对象:



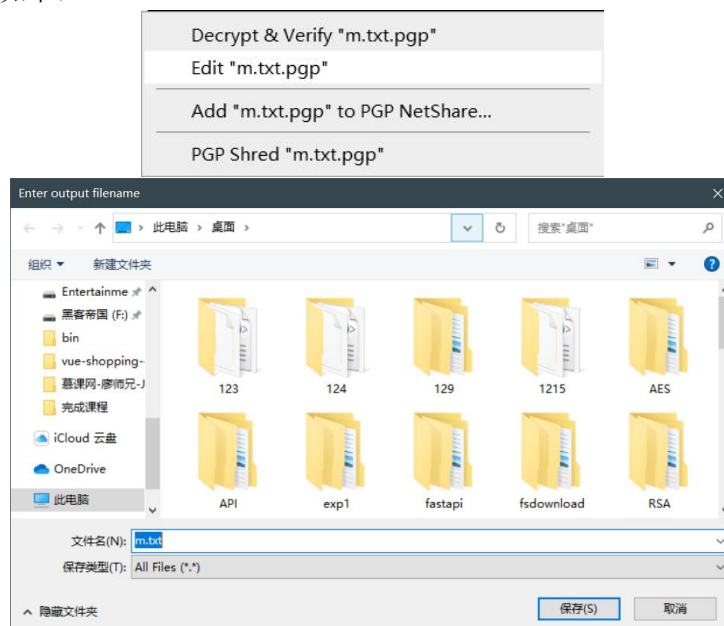
3) 公钥加密:



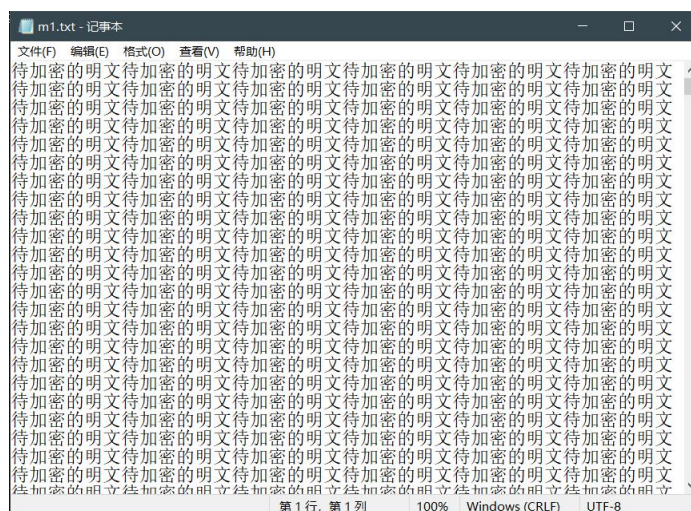
4) 加密结果如下:



5) 解密过程如下:

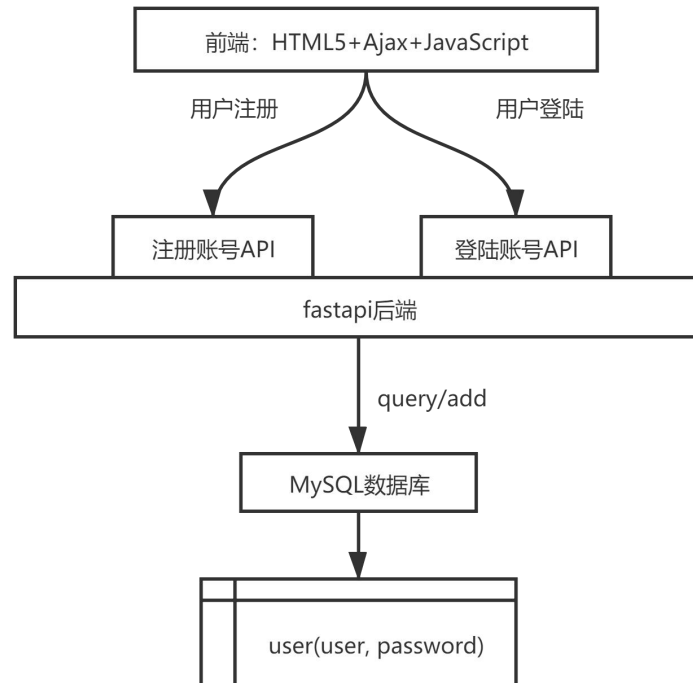


6) 解密结果如下:



2. 应用 ASP 开发或者其他语言结合 SQL 数据库实现数据库的连接并开发一个小型电子商务网站实现注册、登录过程。

(1) 画出功能图



(2) 功能运行截图、程序源代码及结果

1) 前端主要代码:

```

<script>    // 应对所有现代浏览器
    var xmlhttp;
    if (window.XMLHttpRequest)
        { // code for IE7+, Firefox, Chrome, Opera, Safari
        xmlhttp=new XMLHttpRequest();
        }
    else
        { // code for IE6, IE5
        xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
        }
</script>
<div class="checkout-title">
    <span>商城登陆</span>
</div>
<p align="middle">用户: <input type="text" id="user"></p>
    
```

```
<p align="middle">密码: <input type="password" id="password"></p>
<p align="middle"><button onclick="login()">login</button><button
onclick="register()">register</button></p>

<script>
function login() {
    let UID = document.getElementById("user").value;
    let PWD = document.getElementById("password").value;
    xmlhttp.open("POST", "http://127.0.0.1:8895/login", true);
    let info = {
        "user": UID,
        "password": PWD
    }
    xmlhttp.send(JSON.stringify(info));
    xmlhttp.onreadystatechange=function() {
        if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
            let responseStr = xmlhttp.responseText;
            let responseObj = JSON.parse(responseStr);
            if (responseObj.error == 1) {
                window.alert(responseObj.tip);
                window.location.href = 'merchandise.html';
            }
            else
                window.alert(' 登陆失败，请确认用户名和密码后重新登陆');
        }
    }
}

function register() {
    let UID = document.getElementById("user").value;
    let PWD = document.getElementById("password").value;
    xmlhttp.open("POST", "http://127.0.0.1:8895/register", true);
    let info = {
        "user": UID,
        "password": PWD
    }
    xmlhttp.send(JSON.stringify(info));
    xmlhttp.onreadystatechange=function() {
        if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
            let responseStr = xmlhttp.responseText;
            let responseObj = JSON.parse(responseStr);
            if (responseObj.error == 1) {
                window.alert("注册成功，请登陆")
                window.location.href="merchandise.html"
            }
        }
    }
}
```

```
        else
            window.alert(responseObj.tip)
        }
    }
}
</script>
```

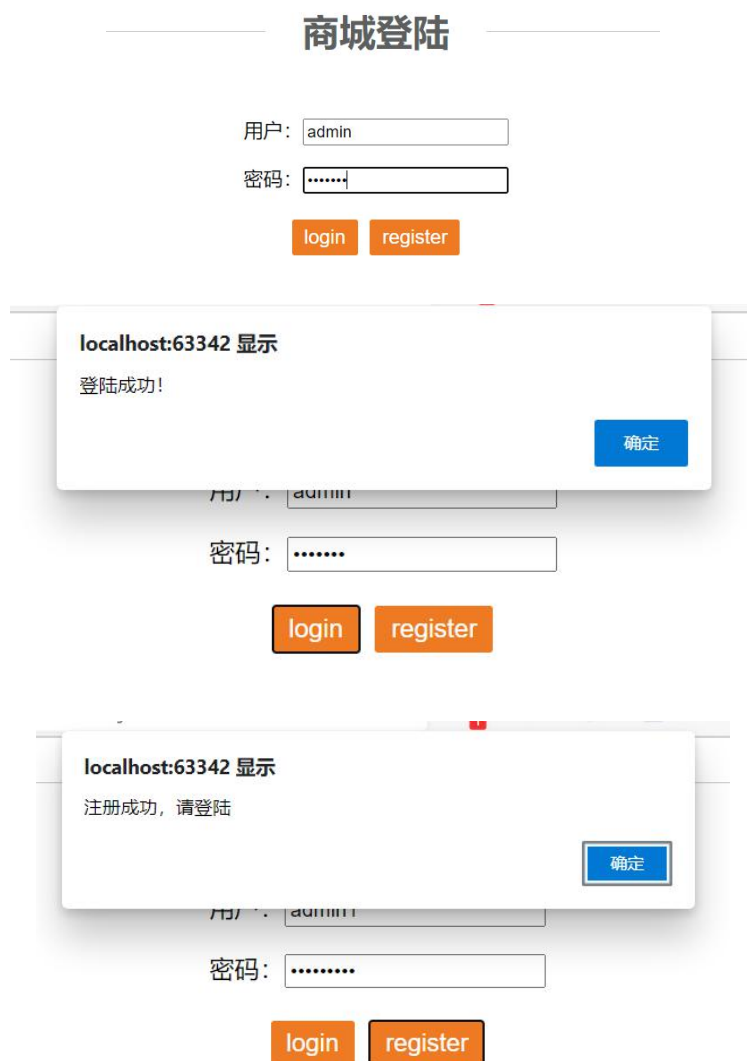
## 2) 后端主要代码:

```
@app.post("/login/")
def login(user: schemas.User, db: Session = Depends(get_db)):
    user_dict = user.dict()
    query = db.query(model.User).filter(model.User.user == user_dict['user']).first()
    if query is None:
        return ResponseModel(error=0, tip='该用户不存在!')
    else:
        md5 = hashlib.md5()
        md5.update(bytes(user_dict['password'], encoding='utf-8'))
        password_md5 = md5.hexdigest()
        if password_md5 == query.password:
            return ResponseModel(error=1, tip='登陆成功!')
        else:
            return ResponseModel(error=0, tip='密码错误!')

@app.post("/register/")
def register(user: schemas.User, db: Session = Depends(get_db)):
    user_dict = user.dict()
    query = db.query(model.User).filter(model.User.user == user_dict['user']).first()
    if query is not None:
        return ResponseModel(error=0, tip="用户已经存在!")
    username = user_dict['user']
    md5 = hashlib.md5()
    md5.update(bytes(user_dict['password'], encoding='utf-8'))
    password_md5 = md5.hexdigest()
    raw = model.User(user=username, password=password_md5)
    try:
        db.add(raw)
        db.commit()
        db.refresh(raw)
        return ResponseModel(error=1, tip="注册成功!")
    except:
        return ResponseModel(error=0, tip="数据库出错!")
```



3) 前端运行结果:



4) 后端运行结果:

```
INFO:      Started server process [5484]
INFO:      Waiting for application startup.
INFO:      Application startup complete.
INFO:      Uvicorn running on http://127.0.0.1:8895 (Press CTRL+C to quit)
INFO:      127.0.0.1:6054 - "POST /login HTTP/1.1" 307 Temporary Redirect
INFO:      127.0.0.1:6054 - "POST /login/ HTTP/1.1" 200 OK
INFO:      127.0.0.1:6064 - "GET /merchandise HTTP/1.1" 200 OK
INFO:      127.0.0.1:6104 - "POST /login HTTP/1.1" 307 Temporary Redirect
INFO:      127.0.0.1:6104 - "POST /login/ HTTP/1.1" 200 OK
```

#### 四、学习体会

1: 通过这次学习, 我掌握了 PGP 软件, 将密码学的理论知识用到了实践上, 未来可以在实践生活中通过 RSA 公钥算法、IDEA 分组密码等技术来保护自己的信息。

2: 网站的搭建采用了 JavaScript、Ajax、python、fastapi 后端框架、MySQL 数据库, 这些技术都是当今流行的。通过这次实践我对一个网站的搭建流程以及网站的组成部分有了更深的理解。

## 实验 2：电子商务系统实训编程及安全认证技术（4 学时）

### 一、实验目的

1. 使用 ASP 或者其他 WEB 语言实现基于数据库的电子商务 B-T0-C 平台部分功能；
2. 掌握 CA 认证中心的功能及数字证书的应用；

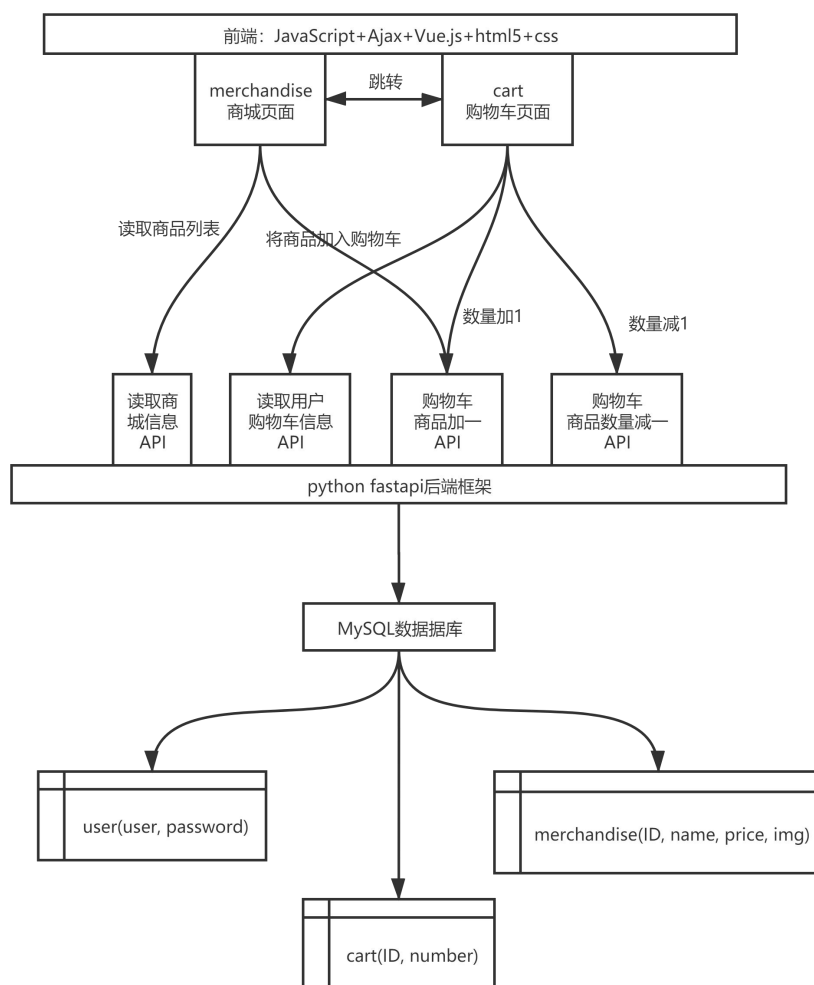
### 二、实验要求

1. 要求学生独立完成实验内容；
2. 按照实验步骤完成实验后，撰写报告内容，并对操作结果进行截图。

### 三、实验内容、实验结果与主要代码

1. 实现商品信息显示的前后台功能、允许用户在线购买商品并实现购物车设计；

(1) 画出功能图



### (2) 功能运行截图、程序源代码及结果

1) 前端源代码:

cart.html:

[illegible]

```
</div>
<script>
    var xmlhttp;
    if (window.XMLHttpRequest)
        { // code for IE7+, Firefox, Chrome, Opera, Safari
        xmlhttp=new XMLHttpRequest();
        }
    else
        { // code for IE6, IE5
        xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
        }
    xmlhttp.open("GET", "http://127.0.0.1:8895/cart", false);
    xmlhttp.send();
    var responseObj;
    let responseStr = xmlhttp.responseText;
    responseObj = JSON.parse(responseStr);
    var app = new Vue({
        el: '#app',
        data:{
            list: responseObj.data,
            modify: false
        },
        computed:{
            totalPrice: function() {
                var total = 0;
                for (var i = 0; i < this.list.length; i++){
                    var item = this.list[i];
                    total += item.price * item.count;
                }
                return total.toString().replace(/\B(?=(\d{3})+)$/g, ',');
            },
            modifyOrNot: function() {
                if (this.modify === true)
                    return "完成修改";
                else
                    return "修改购物车"
            },
        },
        methods:{
            handleReduce: function(index){
                if (this.list[index].count === 1) return;
                this.list[index].count--;
                let res = {};
                res.id = this.list[index].id;
```

```

        xmlhttp.open("POST", "http://127.0.0.1:8895/cart/reduce", false)
        xmlhttp.send(JSON.stringify(res))
    },
    handleAdd: function(index) {
        this.list[index].count++;
        let res = {};
        res.id = this.list[index].id;
        xmlhttp.open("POST", "http://127.0.0.1:8895/cart/add", false)
        xmlhttp.send(JSON.stringify(res))
    },
    handleRemove: function(index) {
        this.list.splice(index, 1);
    },
    modifyFun: function() {
        this.modify = !this.modify;
    }
}
});
</script>
<div class="cart-button-div"><a href="merchandise.html">
    <button class="cart-button">进入商城</button></a>
</div>

```

merchandise.html:

```

<script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
<div class="checkout-title">
    <span>商城</span>
</div>

<div id="app" v-cloak align="center">
    <template v-if="list.length">
        <table>
            <thead>
                <tr>
                    <th>商品图片</th>
                    <th>商品名称</th>
                    <th>商品单价</th>
                    <th>选择加入购物车</th>
                </tr>
            </thead>
            <tbody>
                <tr v-for="(item, index) in list">
                    <td></td>
                    <td>{{item.name}}</td>

```

```
        <td>{{item.price}}</td>
      <td>    <button type="button"
                @click="toCart(index)">加入购物车</button>
      </td>
    </tr>
  </tbody>
</table>
</template>
<div v-else>商品列表为空</div>
</div>
<div class="cart-button-div"><a href="cart.html">
  <button class="cart-button">进入购物车</button></a>
</div>
<script>
  var xmlhttp;
  if (window.XMLHttpRequest)
    { // code for IE7+, Firefox, Chrome, Opera, Safari
      xmlhttp=new XMLHttpRequest();
    }
  else
    { // code for IE6, IE5
      xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
    }
  xmlhttp.open("GET", "http://127.0.0.1:8895/merchandise", false);
  xmlhttp.send();
  var responseObj;
  let responseStr = xmlhttp.responseText;
  responseObj = JSON.parse(responseStr);
  var toCartList = []
  var app = new Vue({
    el: '#app',
    data:{
      list: responseObj.data,
    },
    computed:{
      totalPrice: function(){
        var total = 0;
        for (var i = 0; i < this.list.length; i++){
          var item = this.list[i];
          total += item.price * item.count;
        }
        return total.toString().replace(/\B(?=(\d{3})+)$/g, ',' );
      }
    },
  },
```

```
        methods:{
            toCart: function (index){
                let res = {}
                res.id = this.list[index].id
                xmlhttp.open("POST", "http://127.0.0.1:8895/cart/add", true);
                xmlhttp.send(JSON.stringify(res))
                xmlhttp.onreadystatechange=function() {
                    if (xmlhttp.readyState == 4 && xmlhttp.status == 200){
                        let responseStr = xmlhttp.responseText;
                        let responseObj = JSON.parse(responseStr);
                        window.alert(responseObj.tip);
                    }
                }
            }
        }
    });
</script>
```

## 2) 后端源代码:

```
class ResponseModel(BaseModel):
    error: int = 0
    tip: str = ''

class ResponseCart(BaseModel):
    data: list = []

def get_db():
    db = SessionLocal()
    try:
        yield db
    finally:
        db.close()

class Change2Cart(BaseModel):
    id: int

@app.get("/cart")
def cart(db: Session = Depends(get_db)):
    Carts = db.query(model.Cart).all()
    cartList = []
    for aCart in Carts:
        obj = {}
        aMerchandise = db.query(model.Merchandise).filter(model.Merchandise.ID ==
                                                                aCart.ID).first()
        obj['id'] = aCart.ID
        obj['name'] = aMerchandise.name
        obj['price'] = aMerchandise.price
        obj['img'] = aMerchandise.img
```

```
        obj['count'] = aCart.number
        cartList.append(obj)
    return ResponseCart(data=cartList)

@app.get("/merchandise")
def merchandise(db: Session = Depends(get_db)):
    merchandises = db.query(model.Merchandise).all()
    merchandiseList = []
    for aMerchandise in merchandises:
        obj = {}
        obj['id'] = aMerchandise.ID
        obj['name'] = aMerchandise.name
        obj['price'] = aMerchandise.price
        obj['img'] = aMerchandise.img
        merchandiseList.append(obj)
    return ResponseCart(data=merchandiseList)




@app.post("/cart/add")
def add2Cart(data: schemas.Change2Cart, db: Session = Depends(get_db)):
    CartDict = data.dict()
    id = CartDict['id']
    query = db.query(model.Cart).filter(model.Cart.ID == id).first()
    if query is None: # 购物车中没有
        raw = model.Cart(ID=id, number=1)
        try:
            db.add(raw)
            db.commit()
            db.refresh(raw)
            return ResponseModel(error=1, tip="加入购物车成功!")
        except:
            return ResponseModel(error=0, tip="数据库出错!")
    else: # 购物车中已有, 只需要增加 number 就可以
        query.number = query.number + 1
        try:
            db.commit()
            return ResponseModel(error=1, tip="加入购物车成功!")
        except:
            return ResponseModel(error=0, tip="数据库出错!")
```






```
# http://127.0.0.1:8895/cart/reduce
@app.post("/cart/reduce")
def reduce2Cart(data: schemas.Change2Cart, db: Session = Depends(get_db)):
    CartDict = data.dict()
    id = CartDict['id']
    query = db.query(model.Cart).filter(model.Cart.ID == id).first()
    if query is None: # 购物车中没有
        return ResponseModel(error=0, tip="购物车中没有该商品，无法进行该操作")
    else: # 购物车中已有，只需要减小 number 就可以
        query.number = query.number - 1
        try:
            db.commit()
            return ResponseModel(error=1, tip="数量减小成功!")
        except:
            return ResponseModel(error=0, tip="数据库出错!")
```

### 3) 前端运行结果:

商城

商品图片	商品名称	商品单价	选择加入购物车
	OPPO Enco X真无线主动降噪耳机	999	<a href="#">加入购物车</a>
	OPPO k7x双模5G手机	1499	<a href="#">加入购物车</a>
	苹果11英寸iPad Pro	6229	<a href="#">加入购物车</a>

localhost:63342 显示  
加入购物车成功!

	皮卡丘联名卫衣宠物小精灵牛仔外套夹克卡通衣服神奇宝贝假两件	144	<a href="#">加入购物车</a>
	【库存尾品3本39包邮】特朗普学：我是这样获得成功的/永不放弃特朗普自述书籍特朗普自传从商人到参选总统的成功之道	15	<a href="#">加入购物车</a>
	Bosendorfer贝森朵夫214德国原装进口成人儿童二手全新三角钢琴	700000	<a href="#">加入购物车</a>

进入购物车

购物车

修改购物车

商品图片	商品名称	商品单价	购买数量
	OPPO Enco X真无线主动降噪耳机	999	3
	OPPO k7x双模5G手机	1499	6
	苹果11英寸iPad Pro	6229	4

购物车

完成修改

商品图片	商品名称	商品单价	购买数量	是否移除
	OPPO Enco X真无线主动降噪耳机	999	<div>- 3 +</div>	<div>移除</div>
	OPPO k7x双模5G手机	1499	<div>- 6 +</div>	<div>移除</div>

购物车

完成修改

商品图片	商品名称	商品单价	购买数量	是否移除
	OPPO Enco X真无线主动降噪耳机	999	<div>- 3 +</div>	<div>移除</div>
	OPPO k7x双模5G手机	1499	<div>- 1 +</div>	<div>移除</div>

	LAMY凌美钢笔皮卡丘美国队长卡通联名限量礼盒小学生矫正握姿练字	155	<div>- 1 +</div>	<div>移除</div>
	【库存尾品3本39包邮】特朗普学：我是这样获得成功的/永不放弃特朗普自述书籍特朗普自传从商人到参选总统的成功之道	15	<div>- 2 +</div>	<div>移除</div>
	Bosendorfer贝森朵夫214德国原装进口成人儿童二手全新三角钢琴	700000	<div>- 1 +</div>	<div>移除</div>

总价：¥730,209 

支付

进入商城



#### 4) 后端运行结果:

```
INFO: Started server process [5484]
INFO: Waiting for application startup.
INFO: Application startup complete.
INFO: Uvicorn running on http://127.0.0.1:8895 (Press CTRL+C to quit)
INFO: 127.0.0.1:6054 - "POST /login HTTP/1.1" 307 Temporary Redirect
INFO: 127.0.0.1:6054 - "POST /login/ HTTP/1.1" 200 OK
INFO: 127.0.0.1:6064 - "GET /merchandise HTTP/1.1" 200 OK
INFO: 127.0.0.1:6104 - "POST /login HTTP/1.1" 307 Temporary Redirect
INFO: 127.0.0.1:6104 - "POST /login/ HTTP/1.1" 200 OK
INFO: 127.0.0.1:6128 - "GET /merchandise HTTP/1.1" 200 OK
INFO: 127.0.0.1:6146 - "POST /register HTTP/1.1" 307 Temporary Redirect
INFO: 127.0.0.1:6146 - "POST /register/ HTTP/1.1" 200 OK
INFO: 127.0.0.1:6611 - "GET /merchandise HTTP/1.1" 200 OK
INFO: 127.0.0.1:6825 - "GET /merchandise HTTP/1.1" 200 OK
INFO: 127.0.0.1:6859 - "POST /cart/add HTTP/1.1" 200 OK
INFO: 127.0.0.1:6863 - "POST /cart/add HTTP/1.1" 200 OK
INFO: 127.0.0.1:6867 - "GET /cart HTTP/1.1" 200 OK
INFO: 127.0.0.1:6880 - "GET /cart HTTP/1.1" 200 OK
INFO: 127.0.0.1:7098 - "GET /cart HTTP/1.1" 200 OK
INFO: 127.0.0.1:7158 - "GET /cart HTTP/1.1" 200 OK
INFO: 127.0.0.1:7190 - "GET /cart HTTP/1.1" 200 OK
INFO: 127.0.0.1:7227 - "GET /cart HTTP/1.1" 200 OK
INFO: 127.0.0.1:7243 - "POST /cart/reduce HTTP/1.1" 200 OK
INFO: 127.0.0.1:7243 - "POST /cart/reduce HTTP/1.1" 200 OK
INFO: 127.0.0.1:7243 - "POST /cart/reduce HTTP/1.1" 200 OK
INFO: 127.0.0.1:7243 - "POST /cart/reduce HTTP/1.1" 200 OK
INFO: 127.0.0.1:7243 - "POST /cart/reduce HTTP/1.1" 200 OK
```

2. 在网上找一家权威的 CA 认证中心网站或者 WINDOWS 操作系统自带 PKI 功能, 根据你了解的数字证书在某个领域系统中的应用, 运行该应用功能截图并进行解释;

在编写驱动程序时, 开发阶段需要频繁的给程序进行签名然后测试, 如果这时候每次都使用正式的公司证书来签名就很麻烦, 这种情况下就可以生成一个自我签名的临时证书(根证书)来给程序进行签名, 只要在测试机器上安装上临时证书, 那么程序就可以通过操作系统的验证。

所需要的软件如下: (该程序路径在本机的 windows kits 下)

markecert.exe: 用于生成测试证书, 该工具可以生成数字证书文件 (.cer) 和私钥文件 (.pvk)

pvk2pfx.exe: 可以将私钥文件 (.pvk) 转换成数字证书文件 (.pfx)

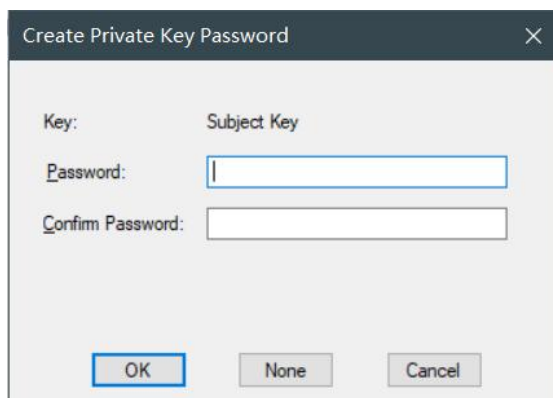
signtool.exe: 可以使用 .pfx 的数字证书文件对其他文件进行签名

其中, “.cer” 为只包含公钥的数字证书文件, “.pfx” 为包含公钥和私钥的数字证书文件

### 1) 生成自我签名证书(根证书)

```
E:\Windows Kits\10\bin\10.0.17763.0\x64>makecert -r -n "CN=Tester" -sv "C:\Users\Administrator\Desktop\certificate\Tester.pvk" "C:\Users\Administrator\Desktop\certificate\Tester.cer" -r
```

-r 表示这是一个自我签名的证书, -n 参数为证书名称, -sv 参数为保存私钥的文件, 执行后提示输入私钥密码, 因为生成的是测试证书, 可以选择不需要密码



### 2) 使用 pvk2pfx.exe 将私钥文件和证书文件合并成一个 .pfx 证书文件

```
E:\Windows Kits\10\bin\10.0.17763.0\x64>pvk2pfx -pvk "C:\Users\Administrator\Desktop\certificate\Tester.pvk" -spc "C:\Users\Administrator\Desktop\certificate\Tester.cer" -pfx "C:\Users\Administrator\Desktop\certificate\Tester.pfx"
```

### 3) 对程序进行签名

```
E:\Windows Kits\10\bin\10.0.17763.0\x64>signtool sign /v /fd SHA256 /f "C:\Users\Administrator\Desktop\certificate\Tester.pfx" "C:\Users\Administrator\Desktop\certificate\xusb22.sys"
The following certificate was selected:
  Issued to: Tester
  Issued by: Tester
  Expires:   Sun Jan 01 07:59:59 2040
  SHA1 hash: OC181447C0DCDD1EFFCFE2A05FC0E1225A778979
Done Adding Additional Store
Successfully signed: C:\Users\Administrator\Desktop\certificate\xusb22.sys
Number of files successfully Signed: 1
Number of warnings: 0
Number of errors: 0
```

/v 表示输出命令执行的信息，/fd SHA256 表示对文件进行 Hash 而使用的算法，/f 指定证书文件，这样驱动程序就被签名了，在测试机上安装证书文件 Tester.cer 之后，驱动程序就可以被系统加载

#### 4) 测试结果:



名称	修改日期	类型	大小
Tester.cer	2020/12/20 13:52	安全证书	1 KB
Tester.pfx	2020/12/20 13:53	Personal Inform...	3 KB
Tester.pvk	2020/12/20 13:52	PVK 文件	2 KB
xusb22.sys	2020/12/20 13:58	系统文件	130 KB

## 四、学习体会

1: 通过这一次实验，我了解了时下最流行的前端框架 vue.js 并且将之付诸实践，收获巨大，了解了 vue 的部分机制。

2: 通过对 markecert.exe、pvk2pfx.exe、signtool.exe 的了解让我能够对测试程序进行签名供自己在操作系统上使用，免除了软件开发中频繁申请数字证书对软件进行签名的麻烦。