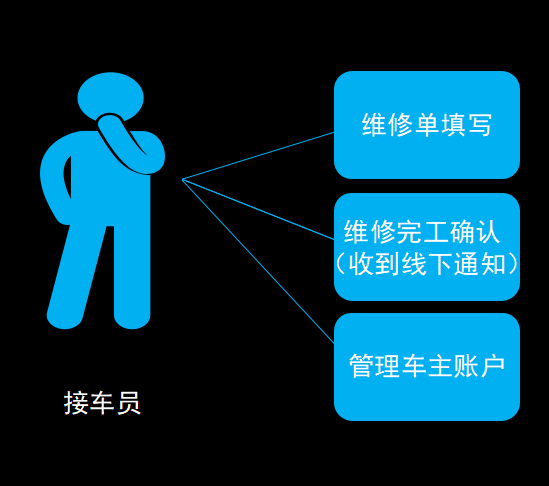
# 汽车维修系统

## （CarRepairSystem）

1. 系统分析
2. 用例



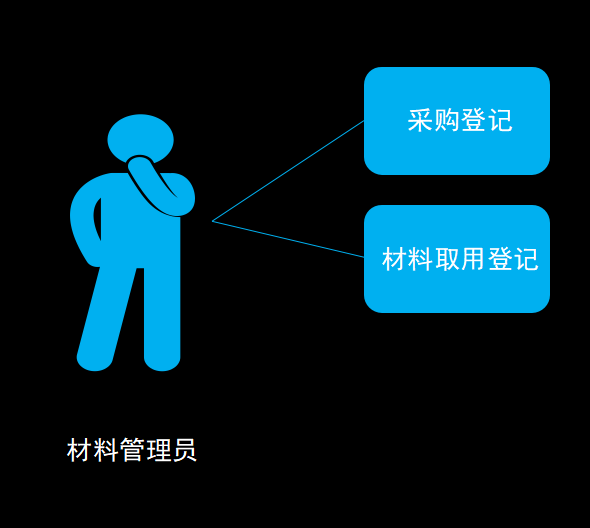
用户：接车员

用例说明：

接车员接待前来维修的车主，车主未注册需要取得车主的联系方式和姓名，车牌号，车牌进行注册，若车主已注册车辆未注册，则需要取得车主的联系方式，车牌号和车牌。

若都已经注册直接利用车牌号进行维修单的填写，填写信息接车员姓名，调度员姓名，还有维修金额，维修项目以及所需材料。

收到线下完工通知进行确认操作。

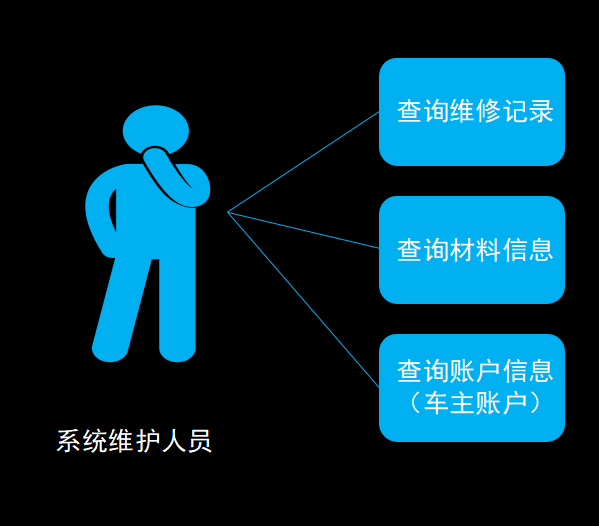


用户：材料管理员

用例说明：

采购人员携带材料和发票进行采购登记并将材料入库。

维修人员到此处利用维修单id获取材料，并将维修项目材料取用标识设置为完成。



用户：系统维护人员

用例说明：

对维修记录进行查询。

对材料信息进行查询。

对车主账户信息进行查询。

总结：主要是对与维修记录（维修单）、材料、车主账户信息的管理。

1. 系统设计
2. 总体设计

轻量级web框架Flask，轻量级数据库sqlite3，类模型利用flask-sqlalchemy，后台语言python（易于部署）前端页面html。

DAO层——》service层——》controller层（借鉴java spring框架，微服务框架，并不是一个用例一个controller）

数据管理——》业务流程，数据一致性——》控制转发

1. 数据库设计

car - car owner 多对一

car - repair order 一对多

repair order - repair material 多对多

*# RepairOrder class***class** RepairOrderDao(db.Model):  
 id = db.Column(db.Integer, primary\_key=True)  
 car\_collector\_name = db.Column(db.String(80))  
 dispatcher\_name = db.Column(db.String(80))  
 repairman\_name = db.Column(db.String(80))  
 inspector\_name = db.Column(db.String(80))  
 car\_id = db.Column(db.Integer)  
 repair\_money\_total = db.Column(db.Float)  
 repair\_start\_time = db.Column(db.DateTime)  
 repair\_end\_time = db.Column(db.DateTime)  
 is\_delete = db.Column(db.Boolean)

*# CarOwner class***class** CarOwnerDao(db.Model):  
 id = db.Column(db.Integer, primary\_key=True)  
 car\_owner\_name = db.Column(db.String(80))  
 car\_owner\_number = db.Column(db.String(80), unique=True)  
 is\_delete = db.Column(db.Boolean)

*# Car class***class** CarDao(db.Model):  
 id = db.Column(db.Integer, primary\_key=True)  
 car\_owner\_id = db.Column(db.Integer)  
 car\_brand = db.Column(db.String(80))  
 plate\_number = db.Column(db.String(80), unique=True)  
 is\_delete = db.Column(db.Boolean)

*# RepairProject class***class** RepairProjectDao(db.Model):  
 id = db.Column(db.Integer, primary\_key=True)  
 repair\_project\_name = db.Column(db.String(80))  
 repair\_material\_id = db.Column(db.Integer)  
 repair\_order\_id = db.Column(db.Integer)  
 repair\_material\_cost\_amount = db.Column(db.Integer)  
 repair\_material\_status = db.Column(db.Boolean)  
 is\_delete = db.Column(db.Boolean)

*# RepairMaterial class***class** RepairMaterialDao(db.Model):  
 id = db.Column(db.Integer, primary\_key=True)  
 repair\_material\_name = db.Column(db.String(80))  
 repair\_material\_has\_amount = db.Column(db.Integer)  
 is\_delete = db.Column(db.Boolean)

1. 控制类设计

对于三类管理的数据，维修数据，车主数据，材料数据的增删查改和通过主键查询。

特殊车主通过手机号查询，车辆通过车牌号查询。

插入和更新的返回值规定为插入和更新后的对象。

**class** RepairOrderController:  
  
 **def** \_\_init\_\_(self):  
 **print** request.args  
  
 *# 输入：  
 # 输出：  
 # 功能：* @staticmethod  
 @app.route()  
 **def** insert():  
 **print** request.args  
  
 *# 输入：  
 # 输出：  
 # 功能：* @staticmethod  
 @app.route()  
 **def** update():  
 **print** request.args  
  
 *# 输入：  
 # 输出：  
 # 功能：* @staticmethod  
 @app.route()  
 **def** select():  
 **print** request.args  
  
 *# 输入：  
 # 输出：  
 # 功能：* @staticmethod  
 @app.route()  
 **def** delete():  
 **print** request.args  
  
 *# 输入：  
 # 输出：  
 # 功能：* @staticmethod  
 @app.route()  
 **def** select\_by\_id():  
 **print** request.args  
  
  
**class** MaterialController:  
  
 **def** \_\_init\_\_(self):  
 **print** request.args  
  
 *# 输入：  
 # 输出：  
 # 功能：* @staticmethod  
 @app.route()  
 **def** insert():  
 **print** request.args  
  
 *# 输入：  
 # 输出：  
 # 功能：* @staticmethod  
 @app.route()  
 **def** update():  
 **print** request.args  
  
 *# 输入：  
 # 输出：  
 # 功能：* @staticmethod  
 @app.route()  
 **def** select():  
 **print** request.args  
  
 *# 输入：  
 # 输出：  
 # 功能：* @staticmethod  
 @app.route()  
 **def** delete():  
 **print** request.args  
  
 *# 输入：  
 # 输出：  
 # 功能：* @staticmethod  
 @app.route()  
 **def** select\_by\_id():  
 **print** request.args  
  
  
**class** CarOwnerController:  
  
 **def** \_\_init\_\_(self):  
 **print** request.args  
  
 *# 输入：  
 # 输出：  
 # 功能：* @staticmethod  
 @app.route()  
 **def** insert():  
 **print** request.args  
  
 *# 输入：  
 # 输出：  
 # 功能：* @staticmethod  
 @app.route()  
 **def** update():  
 **print** request.args  
  
 *# 输入：  
 # 输出：  
 # 功能：* @staticmethod  
 @app.route()  
 **def** select():  
 **print** request.args  
  
 *# 输入：  
 # 输出：  
 # 功能：* @staticmethod  
 @app.route()  
 **def** delete():  
 **print** request.args  
  
 *# 输入：  
 # 输出：  
 # 功能：* @staticmethod  
 @app.route()  
 **def** select\_by\_id():  
 **print** request.args  
  
 *# 输入：  
 # 输出：  
 # 功能：* @staticmethod  
 @app.route()  
 **def** select\_car\_owner\_by\_car\_owner\_number():  
 **print** request.args  
  
 *# 输入：  
 # 输出：  
 # 功能：* @staticmethod  
 @app.route()  
 **def** select\_car\_by\_plate\_number():  
 **print** request.args

1. 界面设计

增删查改式界面，较简单。不涉及登入登出，系统目的就是管理历史记录，以后在此基础上可以发展统计或者数据挖掘的服务。

1. 系统实施