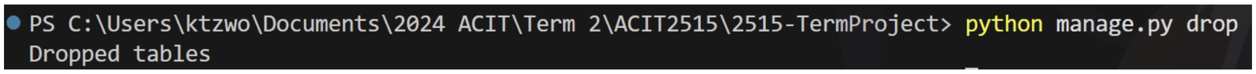
Kevin Wong
A01275270

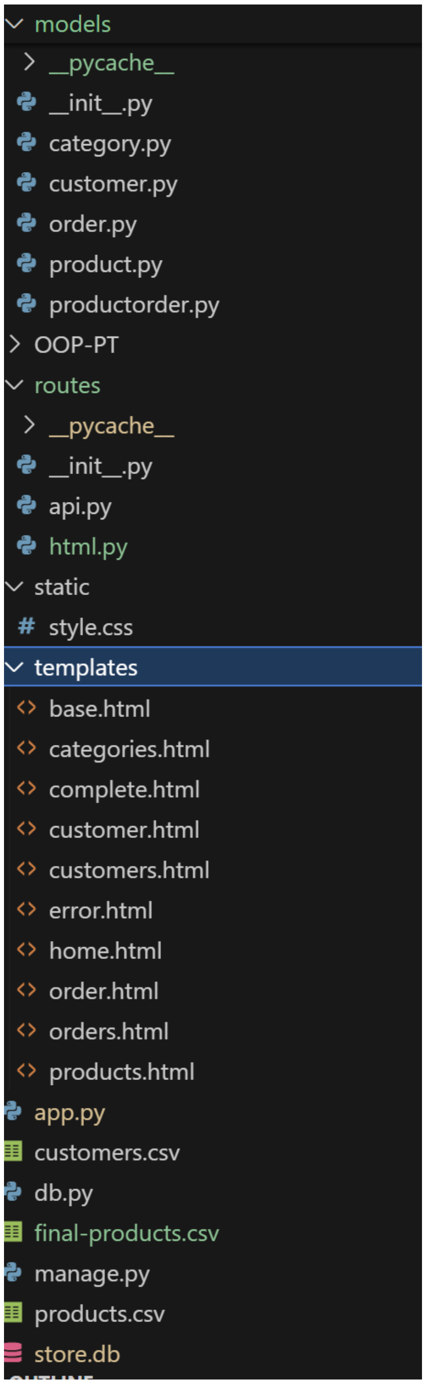
Final Exam

Part 1: Preparing the data base



```
PS C:\Users\ktzwo\Documents\2024 ACIT\Term 2\ACIT2515\2515-TermProject> python manage.py drop
Dropped tables
```

File Structure:

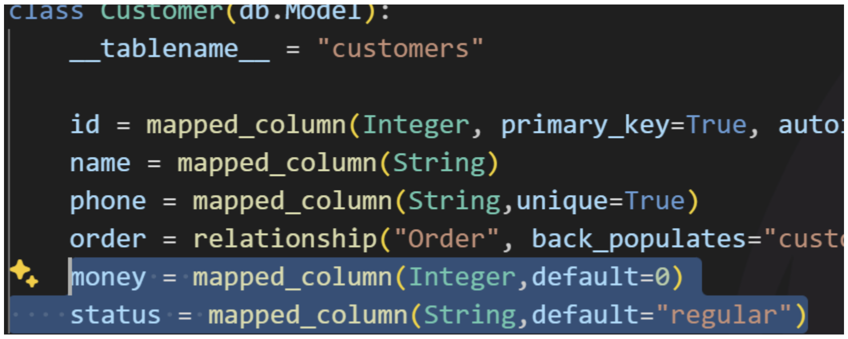


```
models
  __pycache__
  __init__.py
  category.py
  customer.py
  order.py
  product.py
  productorder.py
OOP-PT
routes
  __pycache__
  __init__.py
  api.py
  html.py
static
  style.css
templates
  base.html
  categories.html
  complete.html
  customer.html
  customers.html
  error.html
  home.html
  order.html
  orders.html
  products.html
app.py
customers.csv
db.py
final-products.csv
manage.py
products.csv
store.db
```

Kevin Wong
A01275270

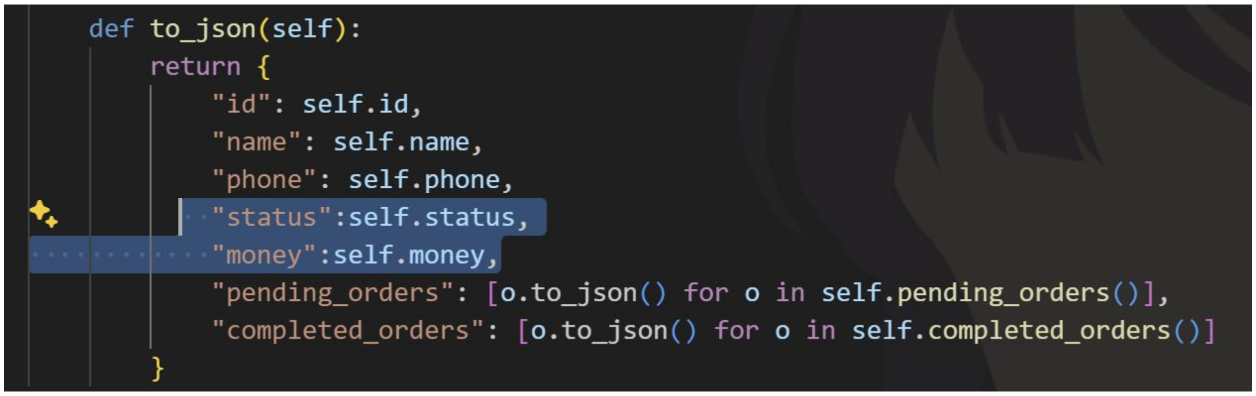
Part 2 : Make changes to to your models

Customer



```
class Customer(db.Model):
    __tablename__ = "customers"

    id = mapped_column(Integer, primary_key=True, auto
    name = mapped_column(String)
    phone = mapped_column(String,unique=True)
    order = relationship("Order", back_populates="cust
    money = mapped_column(Integer,default=0)
    status = mapped_column(String,default="regular")
```

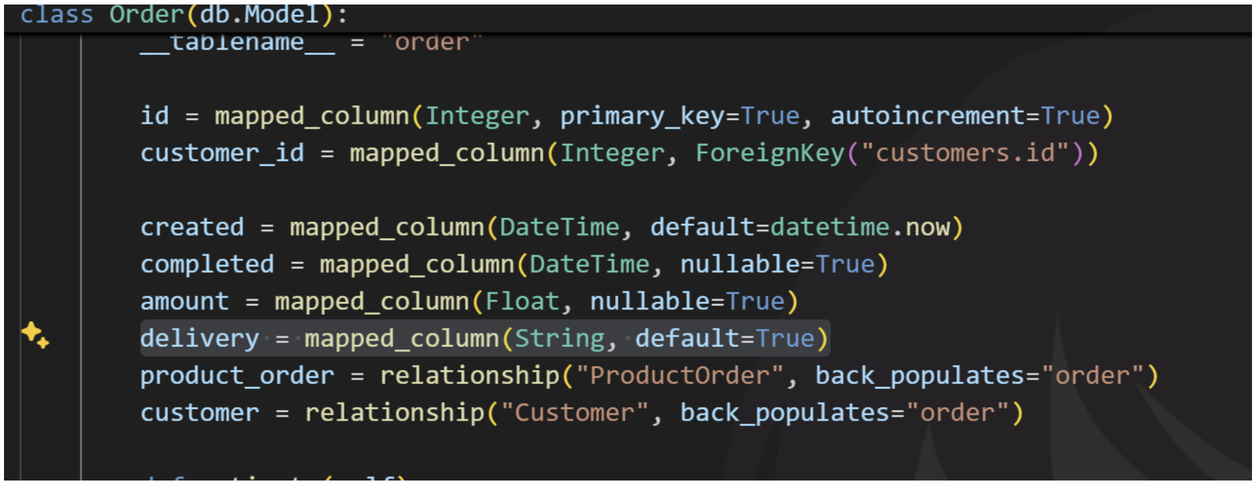
Added 2 attributes, money, and status. Default integer is 0 for money because they start with no money, and status default regular because they start off as regular customers



```
    def to_json(self):
        return {
            "id": self.id,
            "name": self.name,
            "phone": self.phone,
            "status":self.status,
            "money":self.money,
            "pending_orders": [o.to_json() for o in self.pending_orders()],
            "completed_orders": [o.to_json() for o in self.completed_orders()]
        }
```

Added the attributes into to_json method.

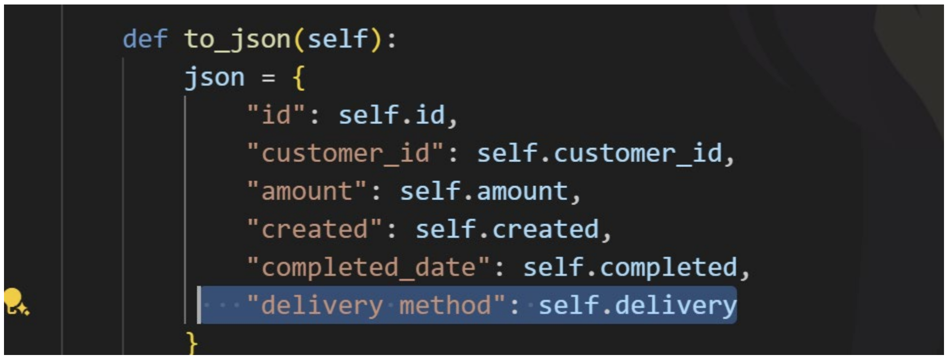
Order



```
class Order(db.Model):
    __tablename__ = "order"

    id = mapped_column(Integer, primary_key=True, autoincrement=True)
    customer_id = mapped_column(Integer, ForeignKey("customers.id"))

    created = mapped_column(DateTime, default=datetime.now)
    completed = mapped_column(DateTime, nullable=True)
    amount = mapped_column(Float, nullable=True)
    delivery = mapped_column(String, default=True)
    product_order = relationship("ProductOrder", back_populates="order")
    customer = relationship("Customer", back_populates="order")
```

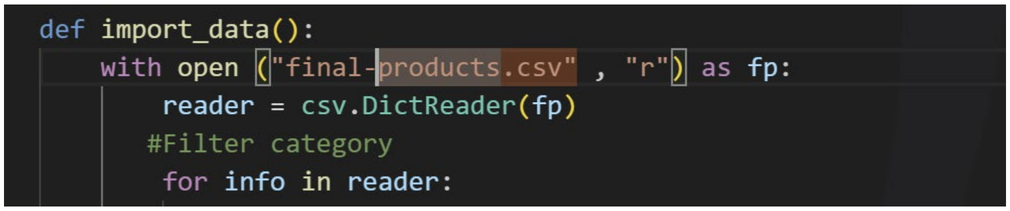
Kevin Wong
A01275270

Added delivery attribute, default="pickup"



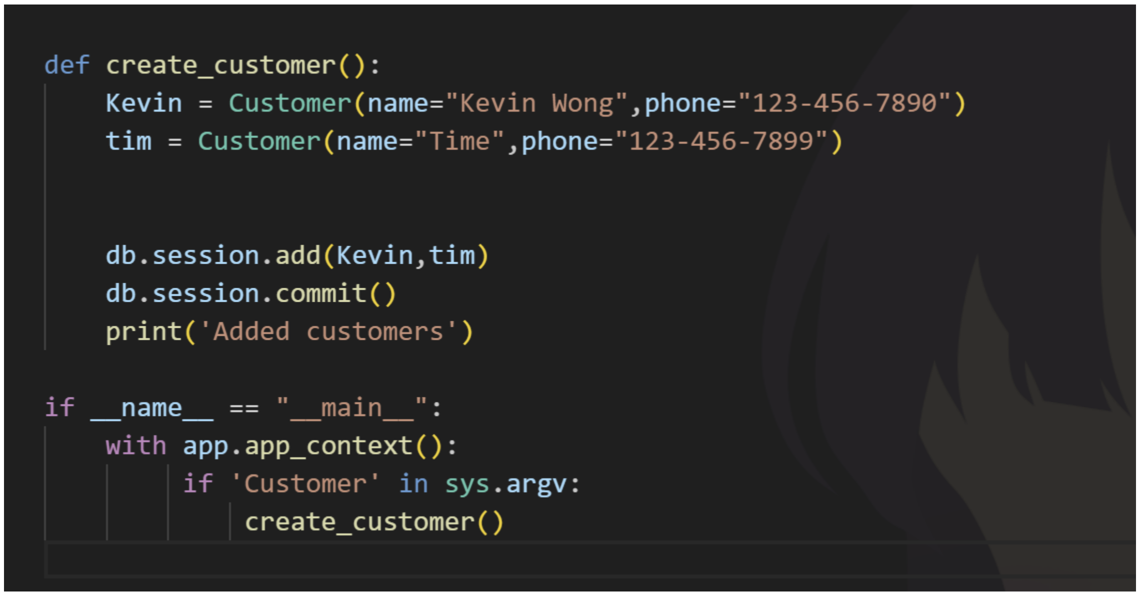
```
def to_json(self):
    json = {
        "id": self.id,
        "customer_id": self.customer_id,
        "amount": self.amount,
        "created": self.created,
        "completed_date": self.completed,
        "delivery method": self.delivery
    }
```

Added delivery method key in to_json method.



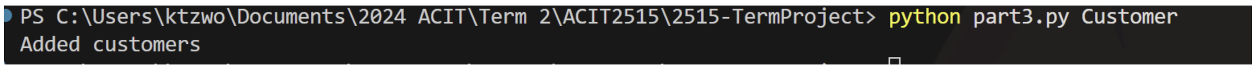
```
def import_data():
    with open ("final-products.csv" , "r") as fp:
        reader = csv.DictReader(fp)
       #Filter category
        for info in reader:
```

Changed product.csv file to final-products.csv file to be able to read the new products



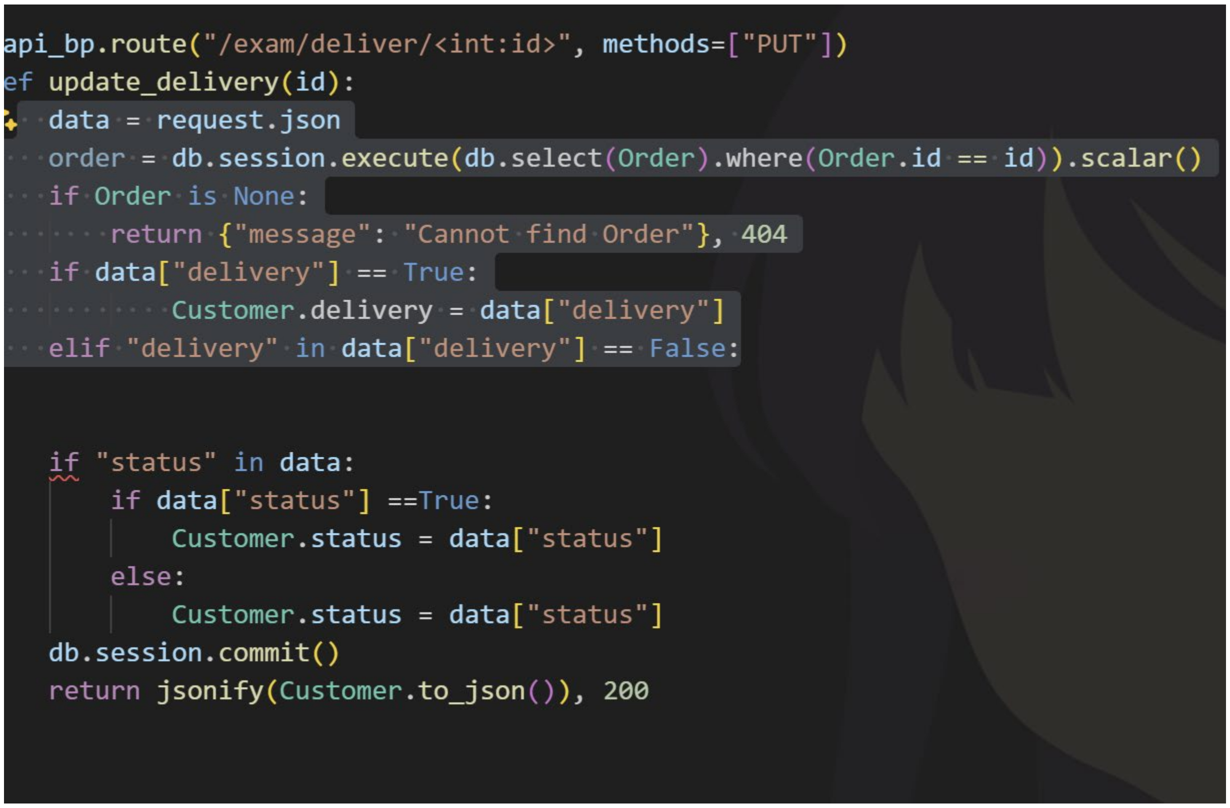
```
def create_customer():
    Kevin = Customer(name="Kevin Wong",phone="123-456-7890")
    tim = Customer(name="Time",phone="123-456-7899")


    db.session.add(Kevin,tim)
    db.session.commit()
    print('Added customers')

if __name__ == "__main__":
    with app.app_context():
        if 'Customer' in sys.argv:
            create_customer()
```

Created 2 customers, and committed them into db.

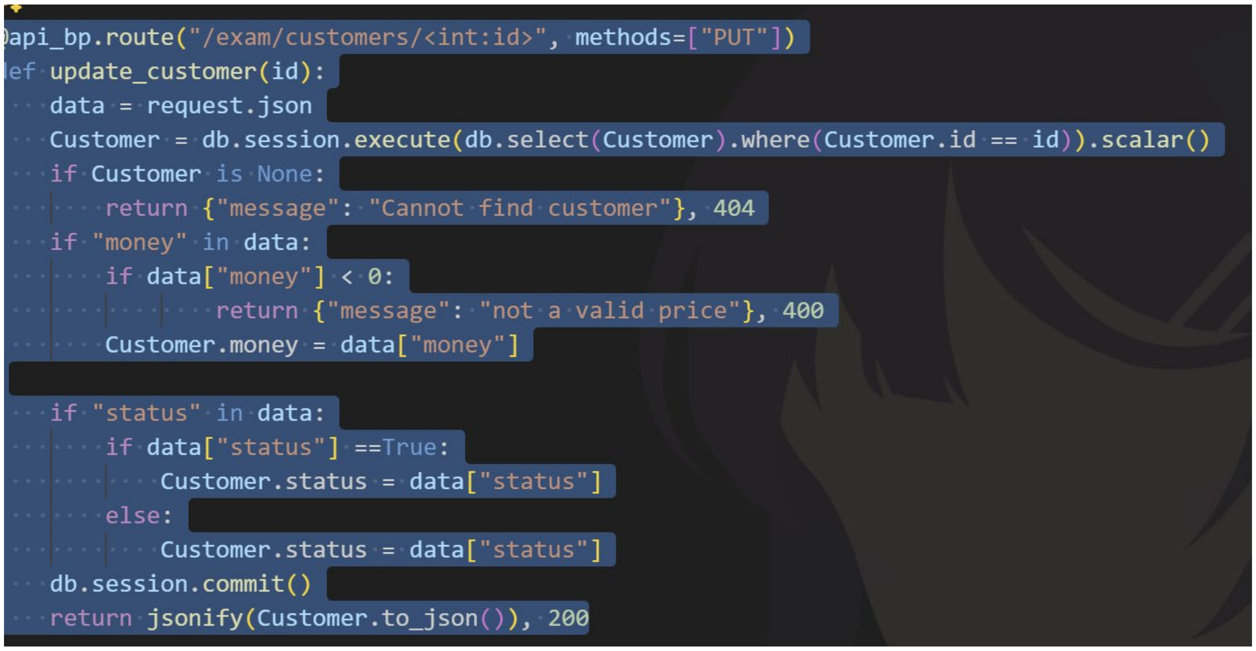


```
PS C:\Users\ktzwo\Documents\2024 ACIT\Term 2\ACIT2515\2515-TermProject> python part3.py Customer
Added customers
```

Kevin Wong
A01275270



```
api_bp.route("/exam/deliver/<int:id>", methods=["PUT"])
ef update_delivery(id):
    data = request.json
    order = db.session.execute(db.select(Order).where(Order.id == id)).scalar()
    if Order is None:
        return {"message": "Cannot find Order"}, 404
    if data["delivery"] == True:
            Customer.delivery = data["delivery"]
    elif "delivery" in data["delivery"] == False:


    if "status" in data:
        if data["status"] ==True:
            Customer.status = data["status"]
        else:
            Customer.status = data["status"]
    db.session.commit()
    return jsonify(Customer.to_json()), 200
```



```
api_bp.route("/exam/customers/<int:id>", methods=["PUT"])
ef update_customer(id):
    data = request.json
    Customer = db.session.execute(db.select(Customer).where(Customer.id == id)).scalar()
    if Customer is None:
        return {"message": "Cannot find customer"}, 404
    if "money" in data:
        if data["money"] < 0:
                return {"message": "not a valid price"}, 400
        Customer.money = data["money"]

    if "status" in data:
        if data["status"] ==True:
            Customer.status = data["status"]
        else:
            Customer.status = data["status"]
    db.session.commit()
    return jsonify(Customer.to_json()), 200
```