1. **Phân tích các yêu cầu / Xác định các luồng hoạt động chính**
2. **Đặt bàn** : chọn tầng => chọn bàn => thay đổi trạng thái của bàn
3. **Đặt món** : mở menu ngay khi chọn bàn thành công => chọn món ăn, số lượng => lưu các món được chọn vào order hiện tại của bàn (*OrderItems*) => lưu thông tin order hiện tại vào lịch sử hoạt động với trạng thái ‘Đang hoạt động’
4. **Huỷ bàn** : bên trong menu chọn món sẽ có tuỳ chọn huỷ bàn => thay đổi trạng thái của bàn => lưu vào lịch sử hoạt động (*Orders*)
5. **Gọi thêm món** : một bàn có người trước đó đã hoàn thành việc gọi món muốn gọi thêm món => chọn món ăn, số lượng => thay đổi order trước đó của bàn
6. **Thanh toán** : thực hiện thanh toán hoá đơn => thay đổi trạng tháu của order tương ứng trong lịch sử hoạt động thành ‘Đã thanh toán’ => thay đổi trạng thái của bàn
7. **Theo dõi ca làm việc** : hiển thị thông tin nhân viên cũng như ca làm việc nhằm dễ dàng quản lý
8. **Update doanh thu theo thời gian thực** : tổng kết doanh thu theo thời gian thực, update liên tục theo thời gian thanh toán, lưu lại lịch sử hoạt động theo ngày tháng
9. **Xác định các gói / các lớp cần thiết**
10. **Gói utils (gói tiện ích):**

* **DatabaseConnector** : lớp tiện ích cho phép tương tác qua lại giữa những dòng code java với cơ sở dữ liệu để lấy dữ liệu cũng như cập nhật cơ sở dữ liệu nhằm phục vụ những luồng hoạt động khác

1. **Gói models : các lớp cốt lõi, chứa các khuôn mẫu đối tượng cũng như những phương thức/logic cơ bản**

* Lớp **Tables**
* Thuộc tính : id, floor, status
* markAsOccupied(), markAsAvailable(): cập nhật trạng thái của bàn sao cho phù hợp vào CSDL
* static findById (id) : cho phép tìm 1 bàn cụ thể bằng id
* static findAll() : hiển thị tất cả các bàn có trong CSDL
* Lớp **MenuItems**
* Thuộc tính : id, name, price
* static findById (id) : phương thức tìm 1 món ăn cụ thể bằng id
* static findAll() : phương thức trả về tất cả các món ăn có trong CSDL
* Lớp **Employee**
* Thuộc tính : id, full\_name, workshift
* static findById(id) : phương thức tìm kiếm 1 nhân viên bằng 1 id cụ thể
* static findAll() : phương thức tìm kiếm tất cả các nhân viên có trong CSDL
* Lớp **Orders**
* Thuộc tính : id, tableId, status, total\_amount, created\_at, List<OrderItems>
* static findActiveByTableId(table\_id) : phương thức tìm kiếm 1 hoá đơn cụ thể bằng table\_id
* addItem (MenuItems name, quantity) : thêm 1 món ăn mới vào hoá đơn hiện tại, phương thức này sẽ gọi tới CSDL để tạo bản ghi List<OrderItems> mới
* removeItems(OrderItem item) : loại bỏ 1 món ăn khỏi hoá đơn
* updateItemsQuantity(OrderItem item, newQuantity) : cập nhật số lượng của 1 món ăn đã có
* calculateTotal() : tính tổng tiền tạm tính của hoá đơn dựa trên danh sách items đang có trong bộ nhớ
* save() : tự động lưu thông tin của hoá đơn
* updateStatusinDB(newStatus) : phương thức cập nhật trạng thái trong CSDL
* markAsPaid() : thay đổi trạng thái của hoá đơn thành đã thanh toán
* markAsCancel() : thay đổi trạng thái của hoá đơn thành đã huỷ
* collectOrderItem() : phương thức lấy tất cả các OrderItem từ CSDL cho hoá đơn hiện tại và đổ vào List trong bộ nhớ
* setTime() : phương thức cập nhật thời gian cho mỗi hoá đơn được thanh toán hoặc bàn bị huỷ
* Lớp **OrderItems**
* Thuộc tính : id, orderId, menu\_item\_name, quantity, price,
* save() : phương thức tự động tạo 1 dòng OrderItem mới để lưu vào CSDL
* update() : phương thức tự động cập nhật thông tin, chủ yếu là quantity của chính nó trong CSDL

1. **Gói Manager : Các lớp đặc thù, không có thuộc tính nhất định, nhưng có những phương thức kết hợp nhiều lớp có trong models lại với nhau để hoàn thành luồng hoạt động**

* **OrderManager**
* startNewOrder(tableId) : phương thức thực hiện luồng Đặt món. Nó sẽ tìm đối tượng *Table* và gọi *table.markAsOccupied()*, sau đó sẽ tạo đối tượng *Order* và gọi *Order.save()*
* checkOut(orderId) : phương thức thực hiện luồng thanh toán, nó sẽ tìm *Order*, gọi *order.markAsPaid()*, sau đó tìm tới *Table* tương ứng và thay đổi trạng thái của bàn
* cancelOrder(orderId) : phương thức thực hiện luồng hoạt động Huỷ bàn, khi này nó sẽ tìm tới *Order*, gọi *order.markAsCancel*, sau đó tìm tới *Table* tương ứng để thay đổi trạng thái
* ReportManager
* getRevenueByDate(date) : phương thức thực hiện luồng update doanh thu, tính tổng doanh thu từ *Orders*
* getOrdersHistory(date) : Lấy lịch sử các hoá đơn đã hoàn thành dựa theo date

1. **Gói Main : Lớp chứa hàm để khởi chạy**