THE POSTS AND TELECOMMUNICATIONS INSTITUTE OF TECHNOLOGY

DEPARTMENT OF INFORMATION TECHNOLOGY 1

INTRODUCTION TO SOFTWARE ENGINEERING

Class	:	**E22CQCN03-B**
Course group	:	**Group 6**
Group topic	:	**Mini football field management**
Project group	:	**Group 12**
Group members	:	trần minh hiếu – B22DCCN320 **Nguyễn Việt Hoàng – B22DCVT214** nguyễn tuấn minh – B22DCKH077 bùi minh tùng – B22DCDT293

MODULE: CUSTOMER PAYING

REPORT: REVISING+DESIGN

Hà Nội - 2025

Concept Exploration

Concepts and Explanations			
#	Concept	Vietnamese	Explanation
	Human-related concept		
1	Field facility owner/proprietor (yard manager)	Chủ sở hữu cơ sở/Quản lý sân (Quản lý sân)	The primary owner or manager of the football field who oversees all business operations, ensures profitability, and manages key decisions.
2	Operations manager	Quản lý vận hành	Manages daily activities, schedules, and ensures smooth operations of the facility, including customer service and field management.
3	General manager	Giám đốc điều hành	Oversees all departments, ensuring they work together efficiently. Handles major business strategies and administrative policies.
4	Financial administrator	Nhà quản lý tài chính	Manages financial records, invoices, budgets, and ensures all transactions, salaries, and expenses are recorded properly.
5	Marketing manager	Nhà quản lý tiếp thị	Promotes the facility, organizes advertising campaigns, manages social media, and develops strategies to attract more customers.
6	HR manager	Nhà quản lý nhân sự	Responsible for hiring, training, and managing staff, along with handling employee relations and payroll.
7	IT administrator	Quản trị viên công nghệ thông tin	Manages technical systems, software, databases, and ensures smooth operation of digital tools and security.
8	Facilities maintenance manager	Quản lý bảo trì cơ sở vật chất	Ensures that the football field, equipment, and other facilities are maintained in good condition and meet safety standards.
9	Front desk receptionist	Lễ tân	Greets customers, handles inquiries, manages bookings, and assists in basic administrative work.

10	Booking coordinator	Điều phối viên đặt chỗ	Manages the scheduling and reservations of football fields, ensuring availability and conflict-free bookings.
11	Field attendant	Nhân viên sân	Assists players, ensures equipment is ready, and helps maintain the playing field.
12	Equipment manager	Quản lý thiết bị	Oversees all sports equipment, ensures they are in good condition, and manages inventory.
13	Maintenance staff	Nhân viên bảo trì	Handles repairs and upkeep of the facility, including field maintenance, lighting, and drainage.
14	Cleaning staff	Nhân viên vệ sinh	Ensures cleanliness of the facility, including changing rooms, shower room, and general areas.
15	Security personnel	Nhân viên an ninh	Maintains order, ensures safety, and enforces facility rules to prevent disturbances.
16	First aid/medical staff	Nhân viên y tế/cấp cứu	Provides medical assistance in case of injuries or emergencies on the field.
17	Cashier/payment processor	Thu ngân/nhân viên xử lý thanh toán	Handles transactions, receives payments, and provides receipts for bookings and other purchases.
18	Inventory clerk	Nhân viên kiểm kê	Manages stock levels of equipment and supplies, ensuring proper documentation of usage and purchases.
19	Canteen staff	Nhân viên căng tin	Prepares and serves food and drinks for players and visitors.
20	Guard	Bảo vệ	Protects the premises and ensures security of the facility and customers.
21	Customer	Khách hàng	Individuals or teams who book and use the football field for matches, training, or recreational play.
22	Equipment suppliers/vendors	Nhà cung cấp thiết bị/nhà thầu	Companies or individuals who supply football-related equipment such as balls, goals, nets, and training gear.

23	Food and beverage providers	Nhà cung cấp thực phẩm và đồ uống	Vendors who supply snacks, drinks, and refreshments for the facility's canteen.
24	Maintenance service providers	Nhà cung cấp dịch vụ bảo trì	Companies or individuals responsible for repairs, field upkeep, and infrastructure maintenance.
25	Marketing partners	Đối tác tiếp thị	Businesses or agencies that assist in promoting the facility through sponsorships or advertisements.
26	Advertisements	Quảng cáo	Sponsored ads placed within the facility, website, or social media to generate extra revenue.
27	Insurance providers	Nhà cung cấp bảo hiểm	Companies that offer coverage for facility damages, injuries, and liabilities.
28	Utility service providers	Nhà cung cấp dịch vụ tiện ích	Providers of electricity, water, and other essential services to run the facility.
29	Payment gateway providers	Nhà cung cấp dịch vụ cổng thanh toán	Companies that facilitate online payments for bookings and services.
30	Technology vendors	Nhà cung cấp công nghệ	Suppliers of software, hardware, and digital systems used for managing bookings, payments, and security.
31	Transportation service providers	Nhà cung cấp dịch vụ vận tải	Companies or individuals that offer transport services for teams or customers visiting the facility.
	Object-related concept		
32	Footballs/soccer balls	Bóng	Essential sports equipment used in all matches and training sessions.
33	Training cones	Các cột chóp nhỏ	Used for agility training, drills, and marking specific areas on the field.
34	Agility ladders	Thang agility	Helps improve foot speed and coordination during training.

35	Goal nets	Lưới cầu môn	Attached to goalposts to catch the ball when a goal is scored.
36	Training bibs/pinnies	Áo bib/Pinnies tập luyện	Worn by players to differentiate teams during practice sessions.
37	Portable goals	Cầu môn di động	Smaller goals used for training, youth games, or temporary field setups.
38	Corner flags	Cờ góc	Placed at the four corners of the field to mark boundaries.
39	Field markers	Dấu vạch sân	Used to outline playing areas and specific zones on the field.
40	Referee equipment (whistles, cards, etc.)	Thiết bị trọng tài (Còi, thẻ, v.v.)	Tools used by referees to enforce rules and maintain fair play.
41	First aid kits	Bộ sơ cứu	Contains medical supplies to treat minor injuries on-site.
42	Cooler box	Thùng lạnh	Used to store and provide cold beverages during matches and training sessions.
43	Ball pumps	Bơm bóng	Used to inflate soccer balls to the required pressure.
44	Equipment storage containers	Container lưu trữ thiết bị	Helps organize and store sports equipment properly.
45	Maintenance tools	Công cụ bảo trì	Tools required for field upkeep and general repairs.
46	Field cleaning equipment	Thiết bị làm sạch sân	Items used to maintain cleanliness of the playing surface and facility.
47	Artificial turf maintenance supplies	Dụng cụ bảo trì thảm nhân tạo	Specialized tools and materials to care for artificial playing surfaces.

48	Lighting equipment/bulbs	Thiết bị chiếu sáng/bóng đèn	Used to ensure proper illumination of the field for night games and training.
49	Scoreboards/timing devices	Bảng điện tử/Thiết bị thời gian	Displays game scores and match time for players and spectators.
50	Benches/seating	Ghế/Chỗ ngồi	Provides seating for players, staff, and spectators.
51	Lockers/storage units	Tủ/lưu trữ	Used for storing players' belongings securely.
52	Shower	Phòng tắm	Facility for players to clean up after a match or training session.
53	Field surface materials	Vật liệu bề mặt sân	Materials used to construct and maintain the playing surface.
54	Boundary markings	Dấu vạch giới hạn	Lines that define the playing area and different field zones.
55	Goal posts	Cột gôn	A board where goals are scored in a match.
56	Fencing materials	Vật liệu hàng rào	Used for enclosing the field to maintain security and prevent unauthorized access.
57	Lighting fixtures	Thiết bị chiếu sáng	Installed to provide adequate visibility during nighttime events.
58	Drainage systems	Hệ thống thoát nước	Ensures proper water flow and prevents field flooding.
59	Changing room fixtures	Thiết bị phòng thay đồ	Includes lockers, benches, and other furnishings in player dressing rooms.
60	Shower facilities	Tiện nghi phòng tắm	Equipped with running water for players to use post-match.

61	HVAC equipment	Thiết bị HVAC	Heating, ventilation, and air conditioning systems for indoor areas.
62	Security systems	Hệ thống an ninh	Cameras, alarms, and access control measures for facility protection.
63	Booking ledgers/scheduling tools	Sổ đăng ký đặt chỗ/công cụ lịch trình	Records and manages reservations for field usage.
64	Receipt books/financial records	Sổ biên nhận/hồ sơ tài chính	Documents all transactions and payments.
65	Invoices	Hóa đơn	Bills issued for services rendered or products sold.
66	Inventory tracking forms	Biểu mẫu theo dõi tồn kho	Used to monitor stock levels of equipment and supplies.
67	Equipment inspection checklists	Danh sách kiểm tra kiểm tra thiết bị	Ensures regular checking of equipment for safety and functionality.
68	Maintenance logs	Nhật ký bảo trì	Records past repairs, maintenance work, and future servicing schedules.
69	Purchase orders	Đơn đặt hàng mua	Documents requests for purchasing supplies and equipment.
70	Delivery notes	Ghi chú giao hàng	Confirms receipt of items ordered from suppliers.
71	Warranty documents	Tài liệu bảo hành	Details manufacturer guarantees and terms of replacement or repair.
72	Certificates of quality	Chứng nhận chất lượng	Certifies that equipment meets required standards.
73	Inventory database records	Hồ sơ cơ sở dữ liệu tồn kho	Digital records of stock levels and inventory usage.

74	Supplier catalogs	Danh mục nhà cung cấp	Lists available products from suppliers.
75	Price lists	Danh sách giá	Contains updated pricing information for goods and services.
76	Product specifications	Thông số kỹ thuật sản phẩm	Detailed descriptions and features of equipment and supplies.
77	Import/export documentation	Tài liệu nhập khẩu/xuất khẩu	Required paperwork for importing or exporting goods.
78	Equipment lifecycle records	Hồ sơ vòng đời thiết bị	Tracks the lifespan of various equipment items.
79	Depreciation schedules	Lịch trình khấu hao	Calculates the reduction in value of assets over time.
80	Equipment usage statistics	Thống kê sử dụng thiết bị	Data on how frequently equipment is used and its effectiveness.
81	Maintenance schedules	Lịch trình bảo trì	Planned routine maintenance to keep equipment functional.
82	Replacement forecasts (optional)	Dự báo thay thế (nếu có)	Predicts when equipment will need replacing based on usage trends.
83	Rental Schedules	Lịch trình cho thuê	Documents the reservation of equipment for temporary use.
	Action-related concept		
84	Ordering supplies	Đặt hàng vật tư	Process of purchasing necessary equipment and materials.
85	Receiving shipments	Nhận hàng	Accepting deliveries and checking for accuracy.

86	Inspecting delivered goods	Kiểm tra hàng hóa giao	Ensuring received items meet quality standards and order specifications.
87	Logging new inventory	Ghi nhận tồn kho mới	Adding newly received stock into the system.
88	Updating stock quantities	Cập nhật số lượng hàng tồn kho	Adjusting inventory records based on new stock levels.
89	Categorizing equipment	Phân loại thiết bị	Sorting equipment into appropriate groups for easier management.
90	Tagging/labeling items	Dán nhãn/đánh dấu các mặt hàng	Marking inventory for tracking and identification.
91	Storing equipment properly	Lưu trữ thiết bị đúng cách	Ensuring equipment is kept in a safe and organized manner.
92	Conducting inventory audits	Tiến hành kiểm kê tồn kho	Reviewing stock levels to verify accuracy and prevent discrepancies.
93	Identifying low-stock items	Xác định các mặt hàng tồn kho thấp	Flagging items that need restocking before running out.
94	Searching for suppliers	Tìm kiếm nhà cung cấp	Finding new vendors to provide equipment, materials, or services.
95	Adding new suppliers to the database	Thêm nhà cung cấp mới vào cơ sở dữ liệu	Entering details of newly onboarded suppliers for future reference.
96	Updating supplier information	Cập nhật thông tin nhà cung cấp	Keeping supplier contact details and product lists current.
97	Requesting price quotes	Yêu cầu báo giá	Asking vendors for pricing on required goods and services.
98	Approving purchase orders	Phê duyệt đơn đặt hàng mua	Authorizing the procurement of goods and services from suppliers.

99	Scheduling deliveries	Lên lịch giao hàng	Coordinating the arrival of supplies to avoid disruptions.
100	Tracking order status	Theo dõi tình trạng đơn hàng	Monitoring shipment progress to ensure timely delivery.
101	Communicating with vendors	Giao tiếp với nhà cung cấp	Maintaining relationships with suppliers and addressing concerns.
102	Resolving delivery issues	Giải quyết vấn đề giao hàng	Handling problems such as late shipments or incorrect orders.
103	Processing invoices	Xử lý hóa đơn	Reviewing and approving payment documents from suppliers.
104	Verifying pricing	Xác minh giá cả	Ensuring billed amounts match agreed-upon prices.
105	Authorizing payments	Phê duyệt thanh toán	Granting approval for financial transactions related to purchases.
106	Recording expenses	Ghi nhận chi phí	Logging all costs associated with operations and purchases.
107	Reconciling delivery notes with orders	Đối chiếu ghi chú giao hàng với đơn hàng	Checking received goods against order details to ensure accuracy.
108	Calculating inventory value	Tính giá trị tồn kho	Determining the total worth of stock items.
109	Processing returns/refunds	Xử lý trả lại/hoàn tiền	Managing defective or incorrect items that need to be sent back.
110	Managing warranties	Quản lý bảo hành	Keeping track of product guarantees and service agreements.
111	Forecasting future expenses	Dự báo chi phí trong tương lai	Predicting upcoming costs for better financial planning.

112	Logging into app	Đăng nhập vào ứng dụng	Accessing the system for booking, inventory, and financial tasks.
113	Searching product database	Tìm kiếm cơ sở dữ liệu sản phẩm	Finding specific items within the system.
114	Filtering search results	Lọc kết quả tìm kiếm	Sorting product lists based on different criteria.
115	Adding items to the import list	Thêm mặt hàng vào danh sách nhập khẩu	Including products for procurement or stocking.
116	Removing items from the import list	Xóa mặt hàng khỏi danh sách nhập khẩu	Taking off unnecessary or incorrect items from purchase lists.
117	Adjusting quantities	Điều chỉnh số lượng	Modifying stock levels before finalizing orders.
118	Calculating totals	Tính tổng số	Summing up costs for purchases, sales, or invoices.
119	Generating reports	Tạo báo cáo	Creating summaries of financials, inventory, or usage data.
120	Printing documents	In tài liệu	Producing hard copies of invoices, purchase orders, and records.
121	Exporting data	Xuất dữ liệu	Saving system data for external use or backups.
122	Adding invoices	Thêm hóa đơn	Entering new billing information into the financial system.
123	Removing invoices	Xóa hóa đơn	Deleting incorrect or outdated billing records.
124	Scheduling equipment inspections	Lên lịch kiểm tra thiết bị	Planning routine checks for equipment functionality.

125	Documenting equipment condition	Ghi nhận tình trạng thiết bị	Recording the status of items to track wear and tear.
126	Marking damaged items	Đánh dấu các thiết bị hư hỏng	Labeling broken or malfunctioning equipment for repair or replacement.
127	Initiating repair procedures	Khởi động quy trình sửa chữa	Starting the process of fixing damaged equipment.
128	Tracking repair status	Theo dõi tình trạng sửa chữa	Monitoring ongoing maintenance and repair work.
129	Recording maintenance history	Ghi nhận lịch sử bảo trì	Keeping logs of past repairs and servicing.
130	Planning equipment replacement	Lập kế hoạch thay thế thiết bị	Scheduling when worn-out items should be replaced.
131	Archiving obsolete items	Lưu trữ các thiết bị đã hết hạn sử dụng	Removing outdated equipment from active inventory.
132	Disposing of unusable equipment	Vứt bỏ các thiết bị không thể sử dụng	Properly discarding equipment that is beyond repair.
133	Recycling materials when possible	Tái chế vật liệu khi có thể	Reusing or disposing of materials in an eco-friendly way.
		Booking	
134	Deposit	Đặt cọc	A pre-payment made to secure a booking or service.
135	Book 1 court	Sổ đăng ký đặt chỗ sân 1	Reserving a single football field for a session.
136	Book 2 adjacent small courts into 1 large court	Đặt sân 2 các sân nhỏ liền kề thành 1 sân lớn	Combining two smaller fields into one larger playing area.

137	Book 4 adjacent small courts into 1 large court	Đặt sân 4 các sân nhỏ liền kề thành 1 sân lớn	Merging four small fields into a bigger playing space.
138	Booking slip	Phiếu đặt sân	A receipt or document confirming a reservation.
139	Clicks on the correct customer name with the current customer	Nhấp vào tên khách hàng đúng với khách hàng hiện tại	Selecting the right user in the system for booking or payment purposes.
140	If the customer first comes to book a court, must add a new one	Nếu khách hàng lần đầu tiên đến đặt sân, phải thêm mới khách hàng	Creating a new customer profile if they are booking for the first time.
141	Multi-session booking	Đặt sân nhiều phiên	Reserving multiple game slots in one transaction.
		Update used items of the rental session	
142	Session check-in/checkout	Kiểm tra/kiểm tra phiên	Tracking customer arrival and departure for booked sessions.
143	Search for goods by name	Tìm kiếm hàng hóa theo tên	Finding specific inventory items using their names.
144	Enter unit and quantities	Nhập đơn vị và số lượng	Inputting the amount of goods being processed.
145	Total amount of customer	Tổng số tiền khách hàng	Calculating the final bill for a customer.
		Payment	
146	Session payment	Thanh toán phiên	Processing fees for booked sessions.
147	Late payment fee	Phí thanh toán muộn	Additional charges for overdue payments.

148	Changing session detail	Chi tiết thay đổi phiên	Modifying the time, date, or duration of a booked session.
149	Confirming payment	Xác nhận thanh toán	Finalizing and approving a customer's transaction.
		Goods importing	
150	Adding new good (if the good does not appear in the system)	Thêm hàng hóa mới (nếu hàng hóa không có trong hệ thống)	Registering a new item in the inventory database.
151	Update new good (if the good appears in the system -> increase the quantity)	Cập nhật hàng hóa mới (nếu hàng hóa có trong hệ thống -> tăng số lượng)	Adjusting stock levels when additional units arrive.
152	Return details of the imported invoice	Chi tiết trả lại hóa đơn đã nhập khẩu	Recording and reviewing the specifics of received shipments.
153	Import success confirmation	Xác nhận thành công nhập khẩu	Verifying that a new stock entry has been added successfully.
	Policy		
154	Loss of properties	Mất mát tài sản	A policy stating the facility is not responsible for lost items.
155	No use of alcoholic drinks	Cấm sử dụng đồ uống có cồn	Prohibition of alcohol on the premises.
156	Damage properties	Hư hỏng tài sản	Rules against vandalizing or misusing facility equipment and spaces.
157	No weapons and fireworks	Cấm vũ khí và pháo	Banning dangerous items to ensure safety.
158	No drugs	Cấm ma túy	Strict prohibition of illegal substances.

159	Not hold responsible for any injuries/accident	Không chịu trách nhiệm về bất kỳ chấn thương/ tai nạn nào	A liability waiver stating the facility is not accountable for injuries.

Business model

A. Business Model by Natural Language

1. Object & Scope

a. Object

- This is a desktop-based application for managing mini football field. It will be internally used inside a mini football field and support only one mini football field.

b. Scope

- Application type: Desktop Based (Business Management Software) which could be installed on many computers of the mini football field employees. However, the database is stored in the mini football field server.
    - User: Only staff could use:
        - Receptionist
        - Staff
        - Yard manager
    - Function:
        - Booking
        - Update used item of the rental session
        - Customer paying
        - Goods importing

2. User & function (Who, what to do)

- Receptionist:
    - Book field(s) on the site customers (with the requirement of the customer)
    - Cancel booking (with the requirement of the customer)
    - Check in (with the requirement of the customer)
    - Check out (with the requirement of the customer)
    - Process the payment (with the requirement of the customer)
- Canteen Staff:
    - Update used items (with the requirement of the customer)
- Yard manager:
    - Import goods (with the requirement of the suppliers)

3. How function work

- Booking:
  → A customer comes to book a mini football field
  →The staff asks the customer what period of time that the customer wants to book.
  → The staff logs in into the system
  → The staff's UI is appeared, it has the following options:
    - Booking
  → The staff selects the Booking function

→ The system displays the interface to find an empty court according to the time slot
→ The staff enters the time slot + select the type of court as requested by the customer + click search
→ The system displays a list of available courts according to the selected time slot
→ Clicks on a court
→ The system displays an interface to fill in customer information
→ The staff enters customer's name and search
→ The system displays a list of customers whose names contains the entered keyword. Clicks on the correct customer name with the current customer. If the customer first comes to book a court, must add a new one
→ The system displays the interface to enter the time period of the start date, end date of the booking (preferred to book by quarter)
→ Clicks confirm
→ The system displays a booking slip with full customer information, booking information, booking price, booking time slot, total number sessions according to the selected time, the estimated total amount and the deposit amount
→ Clicks confirm
→ The system prints the booking slip and updates it to the database.

- *Update used items of the rental session*:
→ A customer arrives to receive the court and return the court for that session
→ The staff logs in into the system
→ The staff's UI is appeared, it has the following options:
    - Update used items
→ The staff selects the Update used items function
→ The menu UI appeared with an input text to enter
→ The staff enters the customer's name + click search
→ The system displays a list of customers with the name entered
→ The staff selects the correct customer name with the current customer information
→ The order's UI displays a list of orders that the customer is booking
→ Clicks on the checkout button rental session 1 booking ticket
→ The system displays an interface to enter the court reception time, return time, and rent (early payment will not be reduced, but late payment will be charged more) with More items used button + Repeat the following steps until complete the list of food products that customers have used during the rental sessions: Click more items used
→ The interface to search for goods by name appears
→ Enters the name of the goods and search
→ The interface for the list of goods with the name entered appears
→ Clicks on 1 item
→ The interface to enter the quantity appears with an input text
→ Enters the quantity and confirms
→ The used item information is added to the list of used items of the session
→ The last line is the total amount of customers
→ Clicks to confirm
→ The system updates to the database (no payment required).

- *Customer paying*:
  → A customer comes to pay for a session
  → The staff logs in into the system
  → The staff's UI is appeared, it has the following options:
    - Process the payment
  → The staff selects the Process the payment function
  → The menu UI appeared with an input text to enter
  → The staff enters the customer's name + click search
  → The system displays a list of customers with the name entered
  → The staff selects the correct customer name with the current customer information
  → The order's UI displays a list of orders that the customer is booking
  → Clicks on the payment button for 1 booking ticket
  → The system displays the invoice with full customer information + 1 list of food and beverage products that the customer has used during the rental sessions as described above + the last line is the total amount paid
  → If the customer complains about a change in the quantity or information about used items, the staff must change, update the detailed list in the corresponding invoice
  → Clicks confirm
  → The system updates to the database.

- *Goods importing*:
  → The yard manager selects the Menu to find Importing
  → The yard Importing UI appears, it has the following 2 options: goods importing and suppliers importing
  → The yard manager selects to suppliers importing
  → The supplier importing UI appears with 3 options: entering, adding and searching button
  → The yard manager enter the name of the supplier need to be searched and then click search button
  → The system displays a list of the providers whose name contains the entered keyword, each row corresponds with the information of the provider: name, address, phone, email, deposit. The yard manager clicks on the row which has the same information to the current supplier(if no rows are satisfying, it has to click on the add new supplier button to add a new supplier
  → The system displays the confirmation UI with: supplier's information
  → The yard manager clicks on the confirm button
  → The system announce a success alert and then, return to the importing UI
  → The yard manager selects to goods importing
  → The goods importing UI appears with 3 options: entering, adding and searching button
  → The yard manager enter the name of the goods need to be searched and then click search button
  → The system displays a list of the goods whose name contains the name just entered, each row corresponds with the information of the goods: id, name, quantity, price, provider's name. The yard manager clicks on the row which has the same information to the current good(if no rows are satisfying, it has to click to the add new goods button to add a new goods

→ The system displays the confirmation UI with prices, quantity and confirm button
→ The yard manager have to enter the prices, the quantity (Repeat goods searching and adding process until all imported goods are selected)
→ After that yard manager needs to click to confirm button
→ The system announces a success alert and then returns to the importing UI.

4. Object related in application

- Football field complex: name, address, description.
- Court: name, description, price.
- Customer: id, name, address, phone, email, note.
- User: name, username, password, role, note.
- Voucher: id, date, owner's information, customer's information, court's information, rental price per session, rental time slot of the week, start date, end date, total expected rent
- Invoice/ bill for customer:
    + Receptionist information: name, role
    + Customer information: name, address, phone, email
    + Court information: id, number of courts, price
    + List of used items: item's name, quantity, price.
- Invoice for provider:
    + Provider information: name, address, phone, email, deposit
    + List of used items: item's name, quantity, price.
- Item: id, name, quantity, price, provider.

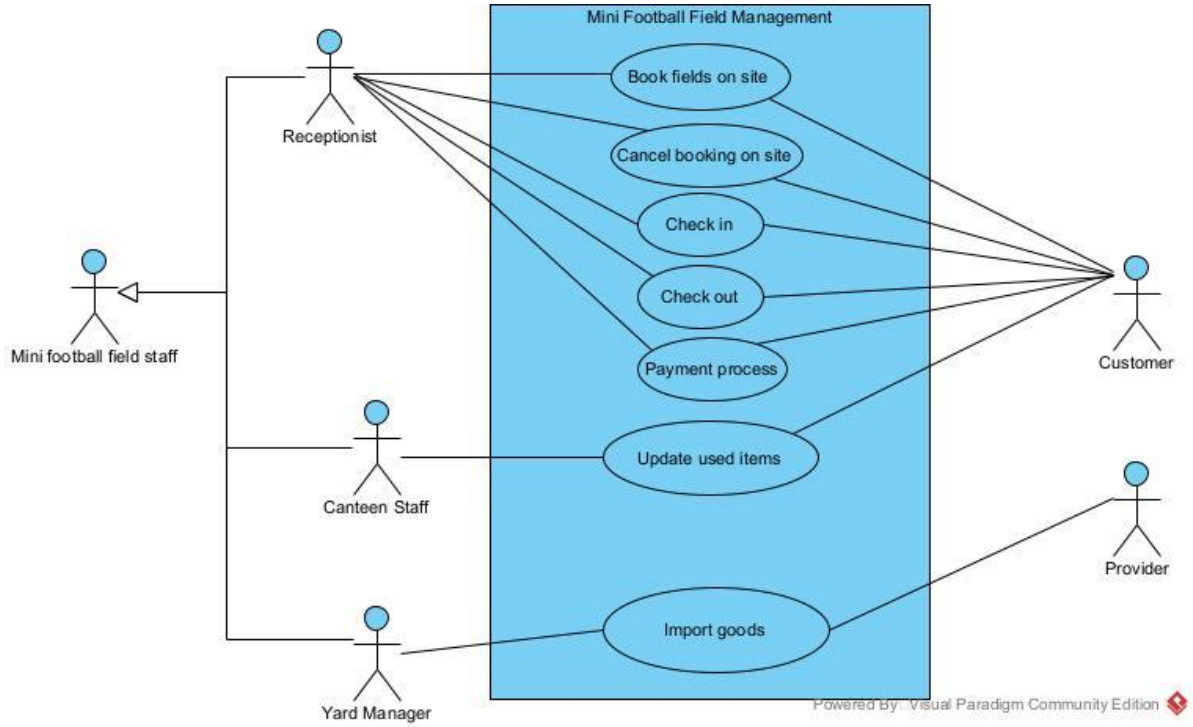
5. Relationship

- A complex has many courts. A court belongs to a complex.
- A court could be booked by many customers in different periods.
- A customer could book many courts in different periods and could also book many courts at the same time.
- 2 or 4 courts could be combined into a bigger court if those courts are adjacent.
- When making a contract to rent a court, the customer receives a voucher. A voucher is received by a customer.
- An invoice for the customer is generated when booking. An invoice for the customer may contain many items. An item may be contained in many invoices.
- An invoice for the provider contains many items. An item may be contained by many invoices for the provider.

B. Business Model by UML

1. Description in UML

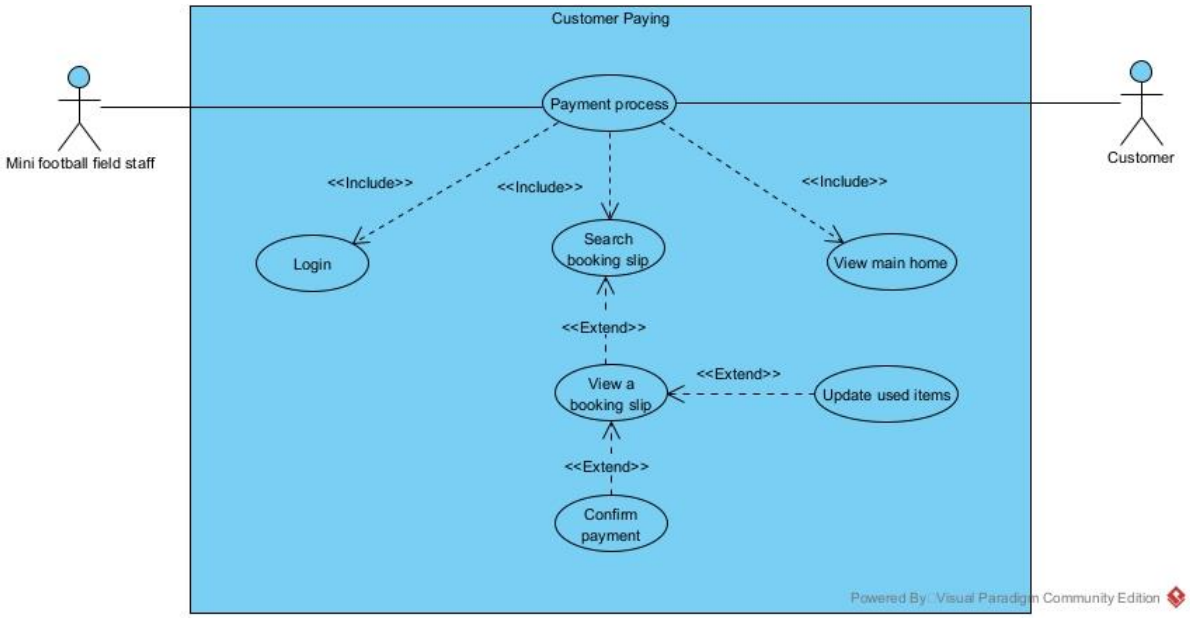
- **General Use Case of the input Q2 (Who? What to do?):**



2. Describe Use Case

- Book fields on site: This Use Case enables the Receptionist to book fields on site with the requirement of the Customer.
- Cancel booking on site: This Use Case enables the Receptionist to cancel a booking on site with the requirement of the Customer.
- Check in: This Use Case enables the Receptionist to check in with the requirement of the Customer.
- Check out: This Use Case enables the Receptionist to check out with the requirement of the Customer.
- Process Payment: This Use Case enables the Receptionist to process the payment with the requirement of the Customer.
- Update used items: This Use Case enables the Canteen Staff to update the list of food product that the Customer has use with the requirement of the Customer.
- Import goods: This Use Case enables the Yard Manager to import goods with the requirement of the Provider.

3. Detail Use Case diagram for module 3: Customer paying



- Description:
  - Login: This use case let the mini football field staff log in to the system
  - View main home page: This use case let the mini football field staff choose the function that the staff can access
  - Payment process: This use case let the mini football field staff use the payment process function
  - Search booking slip: This use case let the mini football field staff search the booking slip by the current customer information
  - View a booking slip: This use case let the staff view the correct booking slip with the provided customer's information
  - Update used items: This use case let the mini football field staff change or update the detailed list of the used items in case the customer complains
  - Confirm payment: This use case let the mini football field staff confirm that the payment for the booking slip has been made

C. Analysis

1. Scenario

Scenario	Update Used Items of the rental session
Actor(s)	Receptionist, Customer
Pre-condition	Receptionist has a receptionist account
Post-condition	- The payment for the session is saved in the database - The detailed list of the used items of the session is updated if the customer has a complain
Main-events	1. The receptionist A open the app to process the payment for the rental session of the customer B 2. The system display a login interface with a field to enter username, a field to enter password and a Login button 3. The receptionist enter username = “a”, password = “staff@123” and click Login 4. The system display the main interface with the option to process payment 5. The receptionist click on the option to process payment 6. The system displays the search for customer with the input of customer name 7. The receptionist ask the customer about their name 8. The customer answer that their name is B 9. The receptionist enter the name B + click search 10. The system show the result interface

Name : B					Search button
N#	ID	Name	Phone	Email	Select
1	2	B	9235692	B@gmail.com	Select
2	5	B	2945834	NVB@gmail.com	Select

11. The receptionist ask the customer about their phone number

12. The customer answer with 9235692

13. The receptionist click on the select button of the correct customer

14. The interface shows a list of booking ticket with that customer name

Name : B			
N#	ID	Rental period	Select
1	2	01/04/2025-30/06/2025	Payment
2	5	01/07/2025-30/09/2025	Payment

15. The receptionist ask the customer which booking slip they want to make payment for

16. The customer answer the rental period between 01/04/2025 to 30/06/2025

17. The receptionist clicks the payment button next to the correct rental period

18. The system shows the interface

Invoice			Confirm
Customer Information			
Name	Phone		Email
B	9235692		B@gmail.com
Items used			
N#	Name		Quantity
Session 1: 01/04/2025			
1	Pepsi		10
2	Revive		5
Session 2: 30/06/2025			
1	Revive		5
		Subtotal	200.000 VND
		Total	21.600.000 VND

19. The customer confirm the information in the invoice, paid the money and the receptionist click the confirm button

20. The system display a confirm box, with a field to enter payment method, a confirm button and a cancel button

21. The receptionist enter the payment method and click confirm button

22. The system display a success alert, confirming that the payment have been successfully save into the database and then returns to the main interface

<table>
<tr>
<td>Exception</td>
<td>
4. The system display a “Login failed” alert and a Confirm button

4.1. The receptionist click the Confirm button

4.2. The system show the login interface

4.3. The receptionist enter the username and password correctly

4.4. The system display the main interface

19. The customer has a complain about this list of items used in the session

19.1. The receptionist click on an item that the customer complain about

19.2. The system show a pop-up interface with the name of the item, a field to enter that item quantity, and a Confirm button
<table>
<tr><td>Item</td><td>Pepsi</td></tr>
<tr><td>Quantity</td><td></td></tr>
<tr><td>Cancel</td><td>Confirm</td></tr>
</table>
19.3. The receptionist enter the number that item base on the customer’s complain and click Confirm

19.4. The system close the pop-up interface, and the invoice is update with the item’s new quantity

19.5. The receptionist then repeat step 19.1. to 19.4. for all the item that the customer complain about
</td>
</tr>
</table>

2. Entity class extraction

- Step 1: Describe the system in a paragraph or a scenario + exceptions

- Step 2+3: Extract and evaluate nouns in Step 1

  + Receptionist => **Need to be managed** => Class User: full name, username, password, role

  + Customer => **Need to be managed** => Class Customer: full name, phone, email, note

  + Account => Too generic => Eliminate

+ Session => **Need to be managed** => Class Session: receptionTime, returnTime, duration, status, note

+ Database => Too generic => Eliminate

+ Item => **Need to be managed** => Class Item: name, price, amount,

unit, category

+ Invoice => **Need to be managed** => Class Invoice: paymentDate,

isPaid, paymentMethod, totalAmount

+ System => Too generic => Eliminate

+ Interface => Too generic => Eliminate

+ Button => Too generic => Eliminate

+ Rental period => Too generic => Eliminate

+ Booking ticket => **Need to be managed** => Class BookingTicket: startDate, endDate, status, createDate, note

+ Alert => Too generic => Eliminate

+ Court => **Need to be managed** => Class Court: type, description, note, condition

=> Class extracted: User, Customer, Session, BookingTicket, Payment, Item

- Step 4: Consider quantities relationship among classes

+ 1 booking ticket has 1 customer, 1 customer can have 0 or many booking ticket => 1-n relationship

+ 1 booking ticket can has many session, 1 session belongs to 1 booking ticket => 1-n relationship

+ 1 session can have 0 or many items used, 1 item can be used in 0 or many session => n-n relationship => Create class UsedItem

+ 1 user can make 0 or many invoice, 1 invoice can be made by only 1 user => 1-n relationship

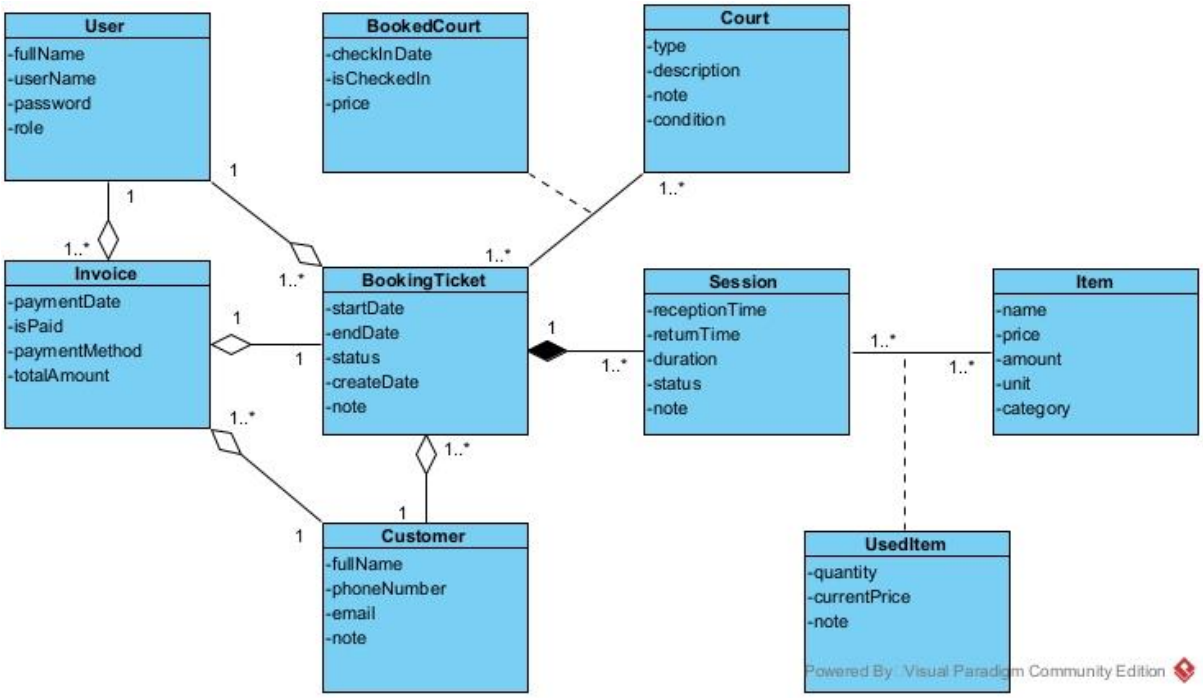
+ 1 booking ticket has 1 invoice, 1 invoice belongs to 1 booking ticket => 1-1 relationship

+ 1 booking ticket can be made by 1 user, 1 user can make many booking ticket => 1-n relationship

+ 1 booking ticket can book many courts, 1 court can be booked in many booking ticket => n-n relationship => Create class BookedCourt

+ 1 invoice can only have1 customer, 1 customer can have many invoice => 1-n relationship

- Step 5: Determine the object relationship among entity classes



3. Full class diagram

- When the receptionist open the app, the system display a login interface => LoginView:

+ Input for username: inUsername

+ Input for password: inPassword

+ Login button: subLogin

- When click the login button, the system check username and password => a function:

+ Name: checkLogin()

+ Input: username, password

+ Output: boolean

+ Owner class: User

- The system displays the main interface => Class MainHomeView:

    + With at least an option Customer paying => subCustomerPaying

- Click on Customer Paying, the interface to search for customer by name appears => Class SearchCustomerView:

    + Input customer name: inName

    + Search button: subSearch

    + List of customer: outsubListCustomer

- Click on search button to search customer by name => a function:

    + Name: searchByName()

    + Input: Customer name

    + Output: List of customer

    + Owner class: Customer

- Click on a customer to show their booking ticket => a function:

    + Name: showBookingTicket()

    + Input: An object of customer

    + Output: List of booking ticket from a customer

    + Owner class: BookingTicket

- Click on a customer, the interface to show the list of booking ticket appears => Class CustomerBookingTicketView:

    + List of booking ticket: outsubListBookingTicket

- Click on the payment button of 1 booking ticket, the interface with the invoice with full customer information and detailed list of items used in the session appear => Class InvoiceView :

    + Confirm the payment: subConfirm

    + List of used item that can be changed: outsubListUsedItem

    + Invoice with full customer information: outInvoice

- Click on the payment button of 1 booking ticket, the invoice with full customer information is shown => a function:

    + Name: getInvoice()

+ Input: An object of booking ticket

+ Output: The invoice for that booking ticket

+Owner class: Invoice

- Click on 1 item, the interface to change item quantity appear => Class ChangeItemNumberView:

+ Input : inNumber

+ Confirm button: subConfirm

+ Cancel button: subCancel

- When click confirm to change item quantity, the system update the invoice in the system => a function:

+ Name: updateUsedQuantity()

+ Input: The number of an item

+ Output: Void

+ Owner class: UsedItem

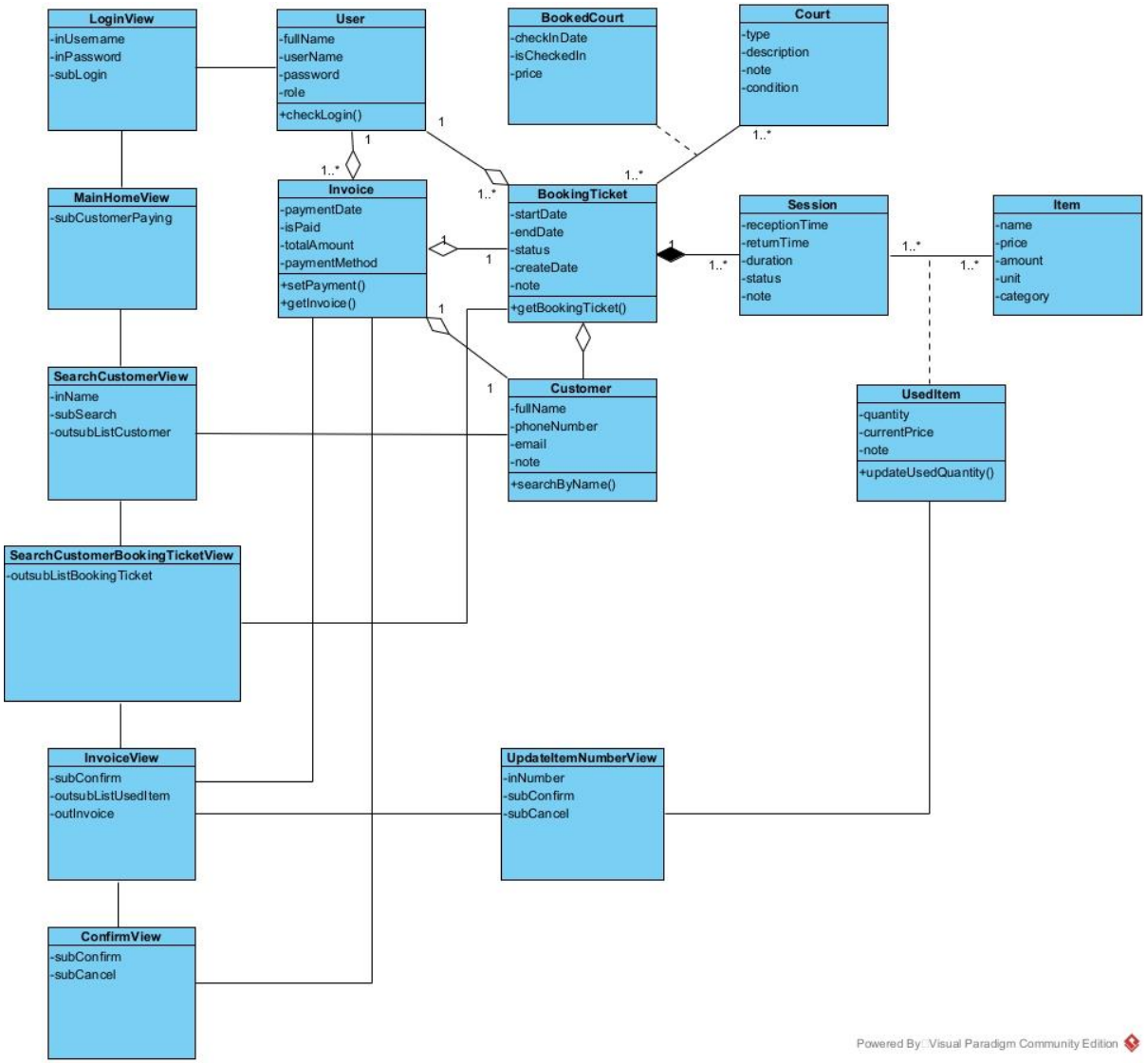
- Click on confirm to update the payment => a function:

+ Name: setPayment()

+ Input: an object of booking ticket and payment method

+ Output: none or boolean

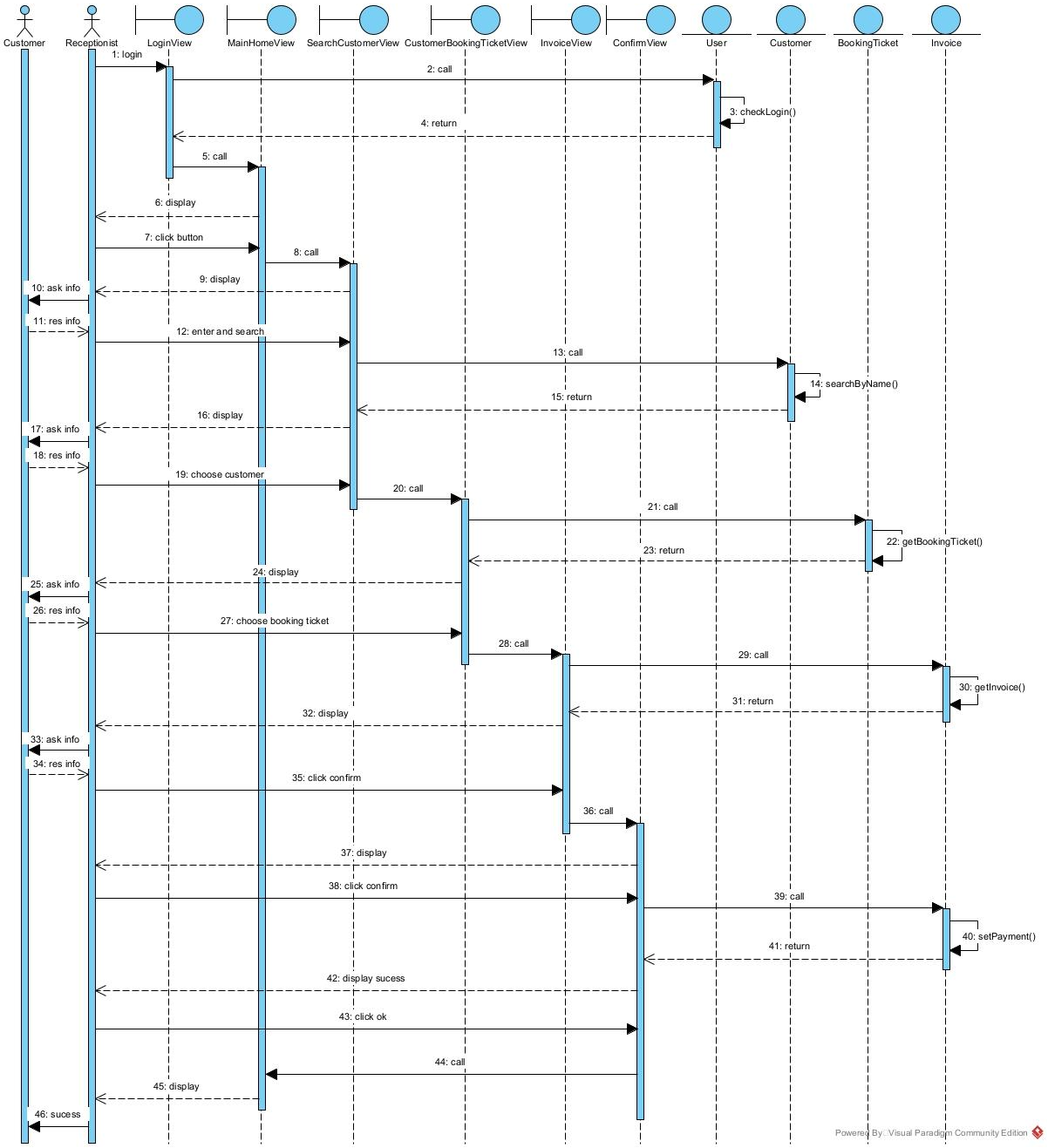
+ Owner Class: Invoice



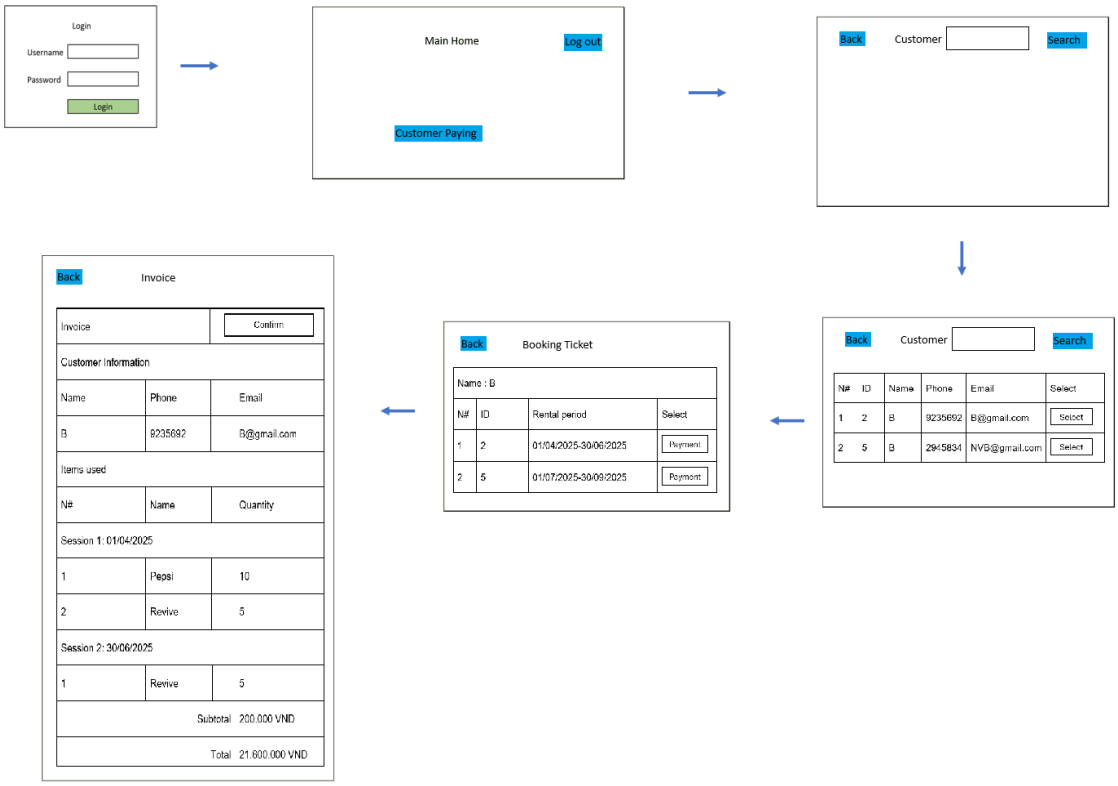
4. Scenario version 2

1. Receptionist open the app, input username and password and click login in class LoginView
2. Class LoginView calls class User
3. Class User use checkLogin() to verify the log in. Log in success
4. Class User return the result to class LoginView
5. Class LoginView calls class MainHomeView
6. Class MainHomeView displays itself to the receptionist
7. The receptionist clicks subCustomerPaying button to use function “Customer paying”
8. Class MainHomeView calls class SearchCustomerView
9. Class SearchCustomerView displays itself to the receptionist
10. The receptionist asks the customer for their name
11. The customer says their name
12. The receptionist enter the customer name and click search
13. The class SearchCustomerView calls class Customer
14. Class Customer use searchByName() to find the list of customer
15. Class Customer return the result to the class SearchCustomerView

16. Class SearchCustomerView displays the list of customer to the receptionist
17. The receptionist asks the customer for their phone number
18. The customer says their phone number
19. The receptionist finds and clicks on the customer with the correct information
20. The class SearchCustomerView calls the class CustomerBookingTicketView
21. The CustomerBookingTicketView call the class BookingTicket
22. The class BookingTicket use function getBookingTicket() to find the list of a customer booking tickets
23. The class BookingTicket return the result to the CustomerBookingTicketView class
24. The class CustomerBookingTicketView displays itself along with the list of booking tickets from a customer to the receptionist
25. The receptionist ask the customer which booking ticket they want to make payment for
26. The customer response with the period of the correct booking ticket
27. The receptionist clicks on the correct booking tickets from the list
28. The class CustomerBookingTicket calls the class InvoiceView
29. The class InvoiceView calls the class Invoice
30. The class Invoice use function getInvoice() to find the invoice
31. The class Invoice return the result to the InvoiceView class
32. The class InvoiceView display itself along with the invoice
33. The receptionist then asks the customer to confirm the invoice and make payment
34. The customer confirms and pays for the invoice
35. The receptionist then clicks confirm
36. The class InvoiceView calls class ConfirmView
37. The class ConfirmView displays itself to the receptionist
38. The receptionist enter payment method and clicks Confirm
39. The class ConfirmView calls class Invoice
40. The class Invoice use function setPayment()
41. The class Invoice return to the class ConfirmView
42. The class ConfirmView displays a success message to the receptionist
43. The receptionist click “OK” in the message
44. The class ConfirmView call class MainHomeView
45. The class MainHomeView displays itself to the receptionist
46. The receptionist informs the successful payment to the customer


Customer
Receptionist
LoginView
MainHomeView
SearchCustomerView
CustomerBookingTicketView
InvoiceView
ConfirmView
User
Customer
BookingTicket
Invoice
1: login
2: call
3: checkLogin()
4: return
5: call
6: display
7: click button
8: call
9: display
10: ask info
11: res info
12: enter and search
13: call
14: searchByName()
15: return
16: display
17: ask info
18: res info
19: choose customer
20: call
21: call
22: getBookingTicket()
23: return
24: display
25: ask info
26: res info
27: choose booking ticket
28: call
29: call
30: getInvoice()
31: return
32: display
33: ask info
34: res info
35: click confirm
36: call
37: display
38: click confirm
39: call
40: setPayment()
41: return
42: display sucess
43: click ok
44: call
45: display
46: sucess
Powered By: Visual Paradigm Community Edition

D. Design



1. Entity class design

- Step 1: Add the id attribute for the classes which DO NOT inherit from other class Court, BookedCourt, BookingTicket, Customer, User, Session, UsedItem, Item, Invoice.

- Step 2: Add the type of each attribute in all classes:

- User:
  - fullName: String
  - username: String
  - password: String
  - note: String
- Invoice:
  - paymentDate: Date
  - isPaid: bit
  - paymentMethod: String
  - totalAmount: int
- BookedCourt:
  - checkinDate: Date
  - isCheckedIn: bit
  - price: int

- Court:
  - type: String
  - description: String
  - note: String
  - condition: String
- BookingTicket:
  - startDate: Date
  - endDate: Date
  - status: String
  - createdDate: Date
  - note: String
- Customer:
  - fullName: String
  - phoneNumber: String
  - email: String
  - note: String
- Session:
  - receptionTime: String
  - returnTime: String
  - duration: String
  - status: String
  - note: String
- Item:
  - name: String
  - price: int
  - amount: int
  - unit: String
  - category: String
- UsedItem:
  - quantity: int
  - currentPrice: int
  - note: String

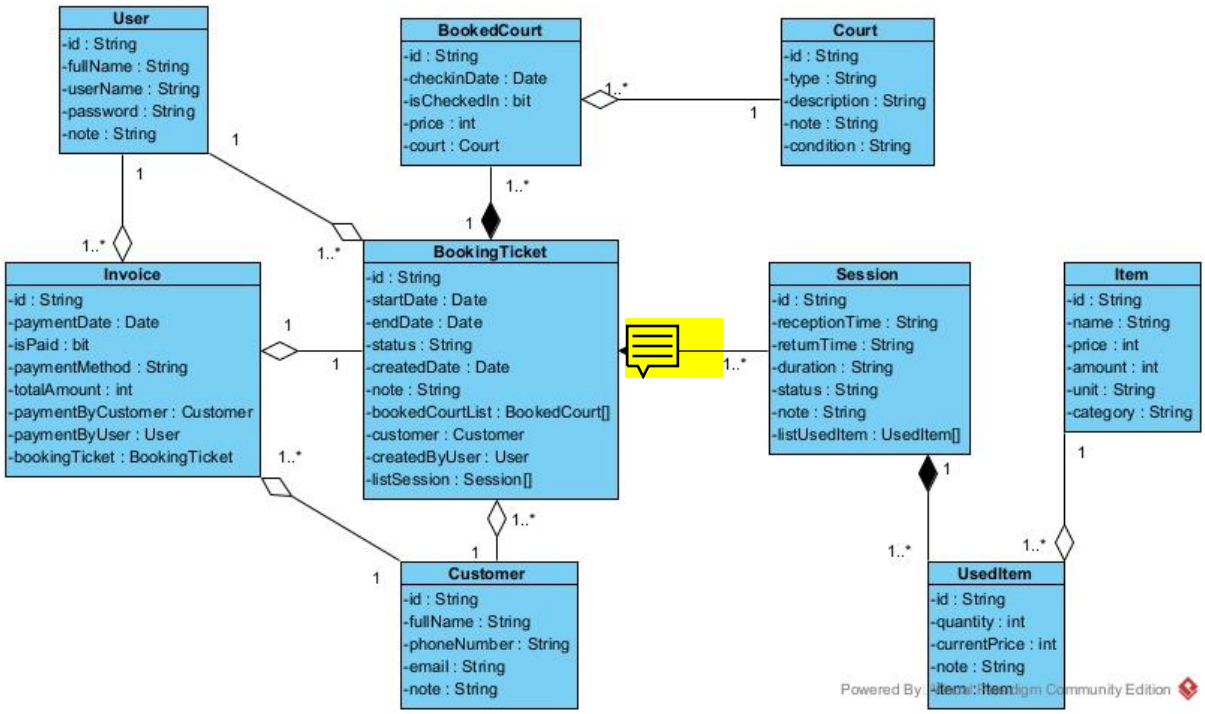
- Step 3: Convert all association relationships to correspond aggregation/ composition relationships:

BookingTicket + Court-> BookedCourt is converted to: Court is a component of BookedCourt, BookedCourt is a component of BookingTicket.

Session + Item-> UsedItem is converted to: Item is a component of UsedItem, UsedItem is a component of Session.

- Step 4: Add the object attributes which correspond to the aggregation/composition relationships:

- Court is a component of BookedCourt, of type 1-n -> BookedCourt has a Court
- BookedCourt is a component of BookingTicket, of type n-1 -> BookingTicket has a BookedCourt
- Customer is a component of BookingTicket, of type 1-n -> BookingTicket has a Customer
- Customer is a component of Invoice, of type 1-n -> Invoice has a Customer
- User is a component of BookingTicket, of type 1-n -> BookingTicket has an User
- User is a component of Invoice, of type 1-n -> Invoice has an User
- BookingTicket is a component of Invoice, of type 1-1 -> Invoice has an BookingTicket
- Item is a component of UsedItem, of type 1-n -> UsedItem has a Item
- UsedItem is a component of Session, of type n-1 -> Session has a list of UsedItem
- Session is a component of BookingTicket, of type n-1 -> BookingTicket has a list of Session



2. Database design

Input is the entity class diagram from the design phase

- Step 1: For each entity class, create a corresponding table.

User -> tblUser

BookedCourt -> tblBookedCourt

Court -> tblCourt

Invoice -> tblInvoice

BookingTicket -> tblBookingTicket

Session -> tblSession

Item -> tblItem

Customer -> tblCustomer

UsedItem -> tblUsedItem

- Step 2: For each entity class, transfer all non-object attributes to contribute as the columns of the corresponding table.

- **tblUser:**
  - id: varchar(50)
  - fullName: varchar(50)
  - userName: varchar(50)
  - password: varchar(50)
  - note: varchar(50)
- **tblBookedCourt:**
  - id: varchar(50)
  - checkinDate: date
  - isCheckedIn: bit
  - price: int
- **tblCourt:**
  - id: varchar(50)
  - type: varchar(50)
  - description: varchar(50)
  - note: varchar(50)
  - condition: varchar(50)
- **tblInvoice:**
  - id: varchar(50)
  - paymentDate: date
  - isPaid: bit
  - paymentMethod: varchar(50)
  - totalAmount: int
- **tblBookingTicket:**
  - id: varchar(50)
  - startDate: date
  - endDate: date
  - status: varchar(50)
  - createdDate: date
  - note: varchar(50)
- **tblSession:**
  - id: varchar(50)
  - receptionTime: varchar(50)
  - returnTime: varchar(50)

    - duration: varchar(50)
    - status: varchar(50)
    - note: varchar(50)
- **tblItem:**
    - id: varchar(50)
    - name: varchar(50)
    - price: int
    - amount: int
    - unit: varchar(50)
    - category: varchar(50)
- **tblCustomer:**
    - id: varchar(50)
    - fullName: varchar(50)
    - phoneNumber: varchar(50)
    - email: varchar(50)
    - note: varchar(50)
- **tblUsedItem:**
    - id: varchar(50)
    - quantity: int
    - currentPrice: int
    - note: varchar(50)

- Step 3: Consider the quantity relationships among entity classes. These relationships will be those among corresponding tables:

1 User – n BookingTicket

1 User – n Invoice

1 Customer – n BookingTicket

1 Customer – n Invoice

1 Invoice – 1 BookingTicket

1 Court – n BookedCourt

1 BookingTicket – n BookedCourt

1 BookingTicket – n Session

1 Session – n UsedItem

1 Item – n UsedItem

- Step 4: Configure the key column for tables.

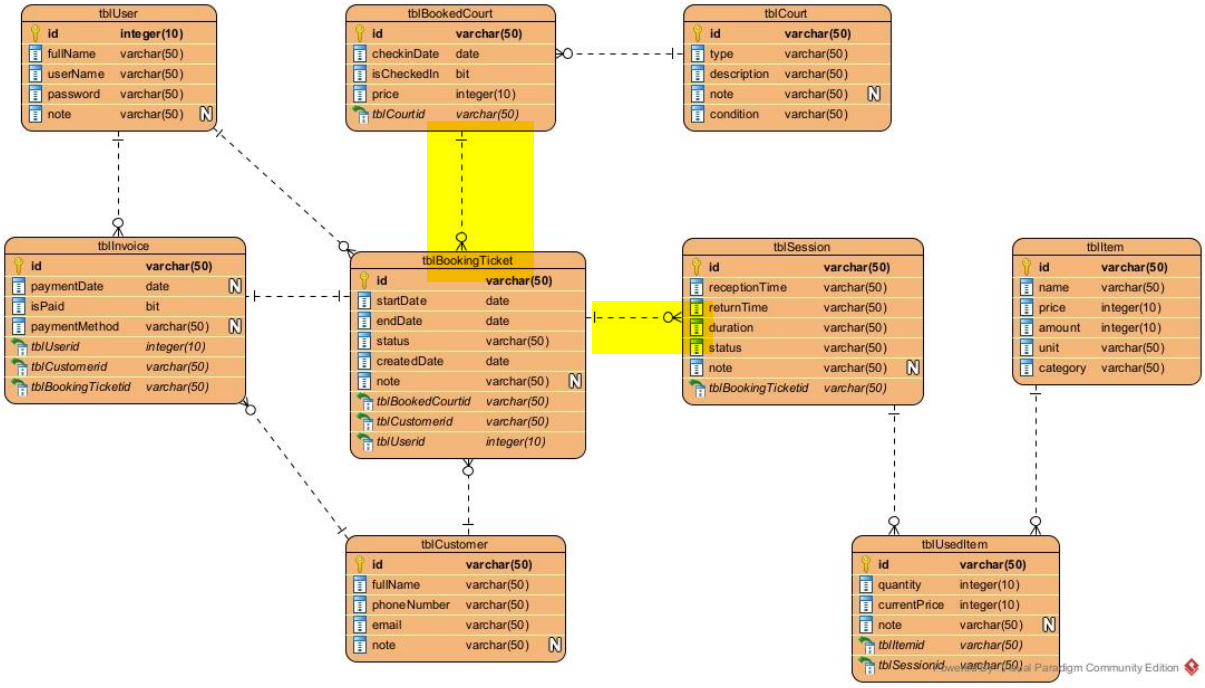
- **tblUser:**

    - id: primary key
- **tblBookedCourt:**
    - id: primary key
    - The relationship between tblCourt and tblBookedCourt is 1-n -> tblBookedCourt will save the foreign key of CourtID references to tblCourt.
    - The relationship between tblBookingTicket and tblBookedCourt is 1-n -> tblBookedCourt will save the foreign key of BookingTicketID references to tblBookingTicket.
- **tblCourt:**
    - id: primary key
- **tblInvoice:**
    - id: primary key
    - The relationship between tblUser and tblInvoices is 1-n -> tblInvoices will save the foreign key of UserID references to tblUser.
    - The relationship between tblCustomer and tblInvoices is 1-n -> tblInvoices will save the foreign key of CustomerID references to tblCustomer.
    - The relationship between tblBookingTicket and tblInvoices is 1-1 -> tblInvoices will save the foreign key of BookingTicketID references to tblBookingTicket.
- **tblBookingTicket:**
    - id: primary key
    - The relationship between tblUser and tblBookingTicket is 1-n -> tblBookingTicket will save the foreign key of UserID references to tblUser.
    - The relationship between tblCustomer and tblBookingTicket is 1-n -> tblBookingTicket will save the foreign key of CustomerID references to tblCustomer.
- **tblSession:**
    - id: primary key
    - The relationship between tblBookingTicket and tblSession is 1-n -> tblSession will save the foreign key of BookingTicketID references to tblBookingTicket.
- **tblItem:**
    - id: primary key
- **tblCustomer:**
    - id: primary key
- **tblUsedItem:**
    - id: primary key
    - The relationship between tblItem and tblUsedItem is 1-n -> tblUsedItem will save the foreign key of ItemID references to tblItem.
    - The relationship between tblSession and tblUedItem is 1-n -> tblUsedItem will save the foreign key of ItemID references to tblItem.

- Step 5: Delete the derived attribute and abundant attribute

Derive Attribute: totalAmount in tblInvoices.

The database for the system:



3. Static design

In this module, the login processing is omitted.

View class

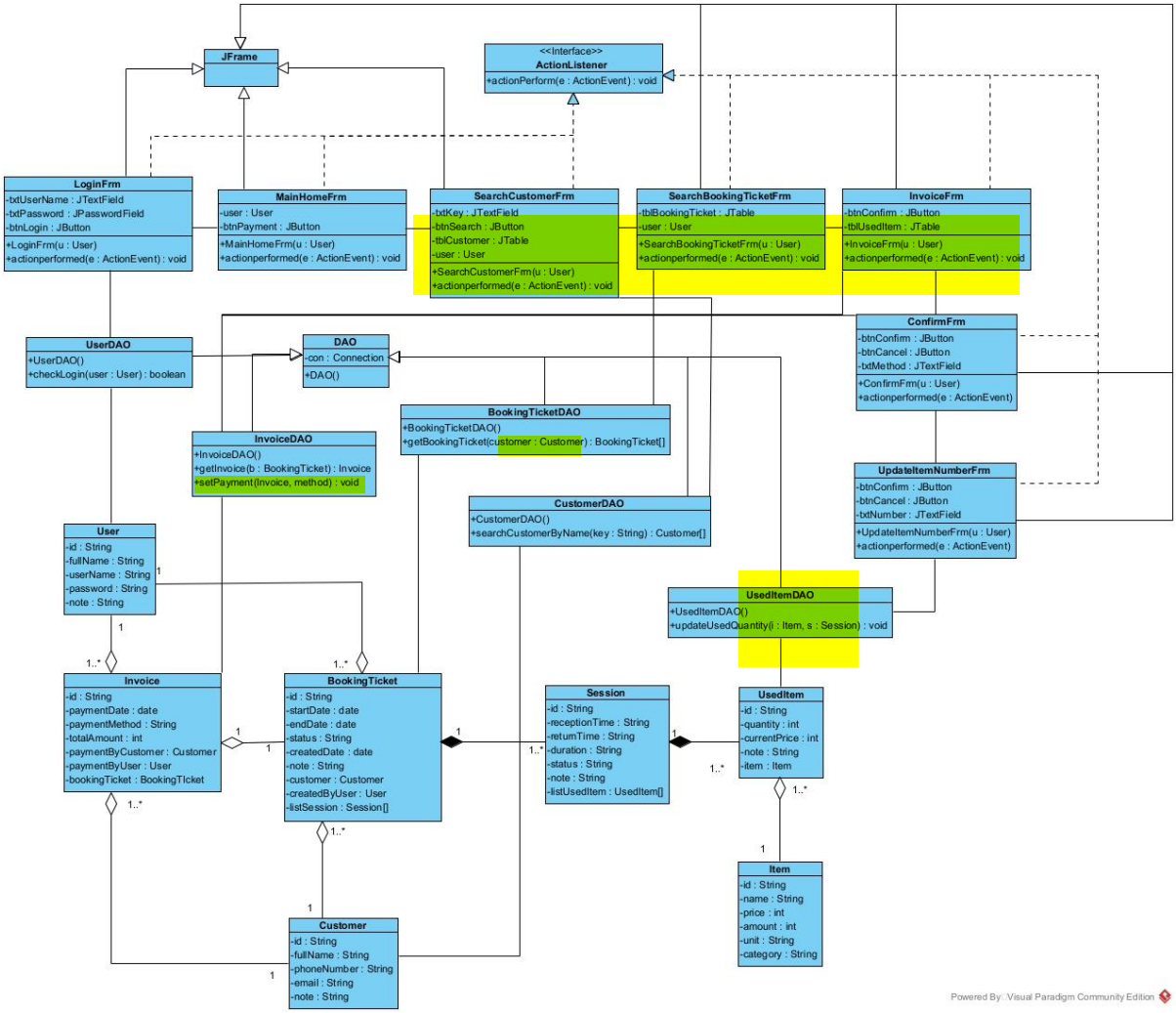
- LoginFrm is the interface to log in. It needs a text field to enter the username, a text field to enter a password, and a button to log in.
- MainHomeFrm is the home interface of the employee. It needs at least a button to go to the receptionist payment function and a button to logout.
- SearchCustomerFrm is the interface of searching customer. It includes a search button function, a text field to enter customer name and a table displaying list of all customer.
- SearchBookingTicketFrm is the interface of searching booking ticket. It includes a table displaying all booking ticket of a customer.
- InvoiceFrm is the interface of displaying invoice. It includes a button to confirm the payment and a table displaying the list of used item.
- ConfirmFrm is the interface of confirm payment. It includes a text field to enter the payment method, a button to confirm and a button to cancel.
- UpdateItemNumberFrm is the interface of updating item number. It needs the input text field to enter the number of the item, a button to confirm the update and a cancel button.

Control(DAO) class:

- DAO is a general class of DAO. It has only the construction to connect to the DB and provides the common connection for all inherited DAO classes in the system.
- UserDAO is the class for manipulating with DB related to the User object. In this module, it needs a method:
    - checkLogin(): to verify whether the login information is correct or not.
        - Input: User, ensure encapsulation
        - Output: Boolean, check if the login is valid, if not return false
- CustomerDAO is the class for manipulating with DB related to the Customer object. In this module, it's need a method:
    - searchByName(): to search all customer whose name contains the keyword.
        - Input: String, the information is given by the customer for the receptionist to enter
        - Output: Customer[], there can be many customer with the same name
- BookingTicketDAO: is the class for manipulating with DB related to the booking ticket object. In this module, it's need a method:
    - getBookingTicket(): to search all booking tickets which belong to a customer.
        - Input: Customer, ensure encapsulation
        - Output: BookingTicket[], one customer can have many booking tickets
- InvoiceDAO: is the class for manipulating with DB related to the Invoices Object. In this module, it's need 2 method:
    - getInvoice(): to retrieve all information of the invoice
        - Input: BookingTicket, ensure encapsulation
        - Output: Invoice, ensure encapsulation
    - setPayment(): to set invoice payment method and confirm payment.
        - Input: Invoice, ensure encapsulation
        - Output: Boolean, return true if the payment has been updated successfully

- UsedItemDAO: is the class for manipulating with DB related to the used item Object. In this module, it's need a method:
    - updateUsedQuantity(): to update the quantity of an item in the invoice
        - Input: UsedItem, ensure data encapsulation
        - Output: void, the number of item only need to be updated

Entity classes: User, Invoices, BookingTicket, Session, Customer, Item, UsedItem

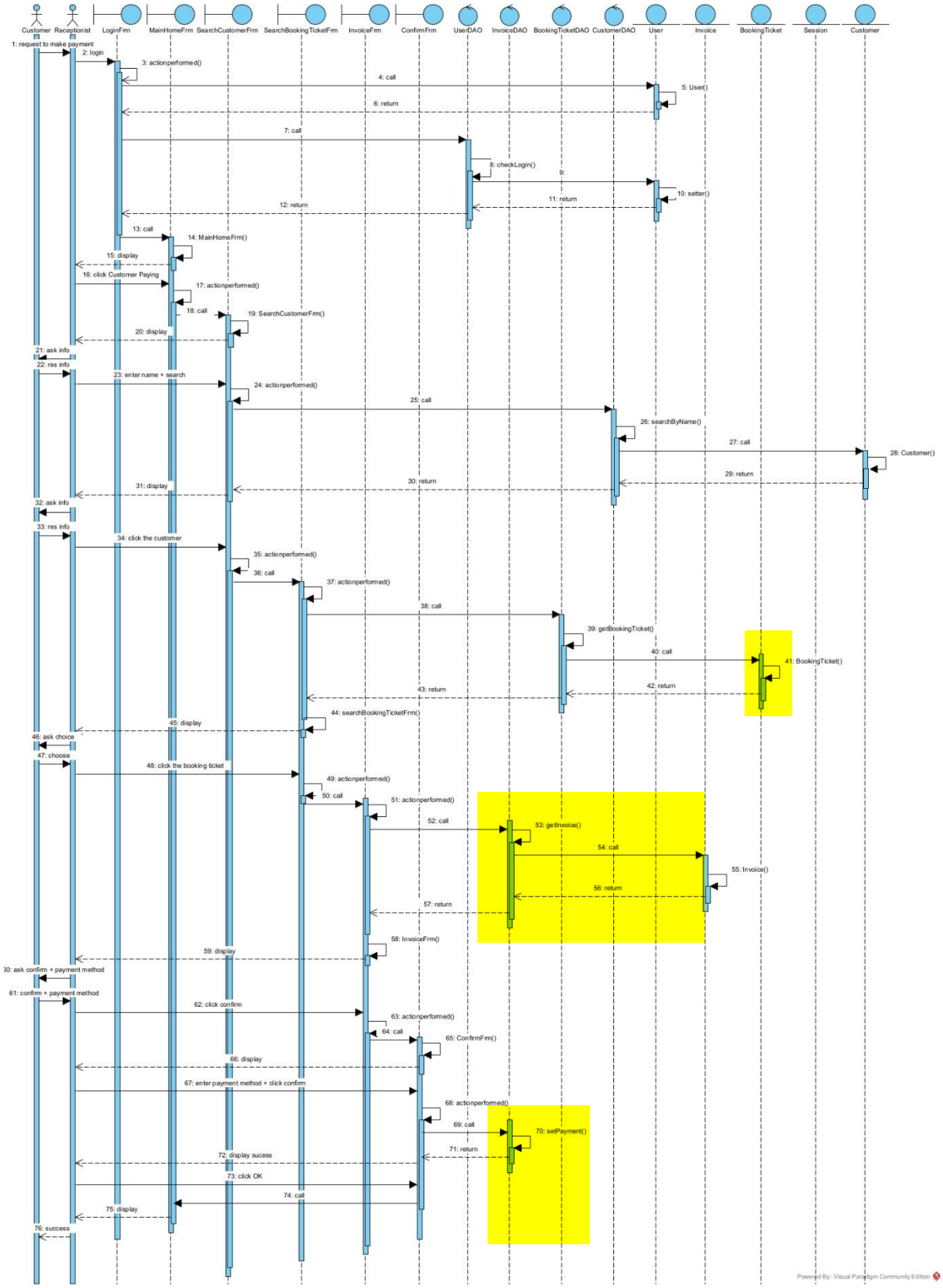


4. Dynamic design

1. A customer come to make payment
2. The receptionist enter username, password and click login button on LoginFrm
3. The method actionperformed() of LoginFrm() is called
4. The method actionperformed() calls User to create an User object
5. The class User packs the information into an User object
6. The class User returns User object to the method actionperformed().
7. The method actionperformed() calls method checkLogin() of the class UserDAO

8. The method checkLogin() checks the login information
9. The method checkLogin() calls the class User set more two attributes name, role
10. The class User calls its method setName(), setRole()
11. The class User returns the User object to the method checkLogin()
12. The method checkLogin() returns the results to the actionPerformed()
13. The method actionPerformed() calls the class MainHomeFrm
14. The constructor MainHomeFrm() is called
15. The interface MainHomeFrm is shown to the receptionist
16. The receptionist clicks on the customer paying button
17. The method actionperformed() of the class MainHomeFrm is called
18. The method actionPerformed() calls the class SearchCustomerFrm
19. The constructor SearchCustomerFrm() is called
20. The interface SearchCustomerFrm is shown to the receptionist
21. The receptionist asks the customer about their name
22. The customer responds with their name
23. The receptionist enter their name into the name field and click search
24. The method actionperformed() is called
25. The method actionPerformed() calls the method searchByName() of the class CustomerDAO
26. The method searchByName() is executed
27. The method searchByName() calls the class Customer to pack the results
28. The class Customer pack each result into a Customer object
29. The class Customer returns the object to the method searchByName()
30. The method searchByName() returns the results to the method actionperformed()
31. The method actionperformed() displays the results on the interface SearchCustomerFrm to the receptionist
32. The receptionist asks for the customer phone number
33. The customer responds with their phone number
34. The receptionist clicks on the correct customer
35. The method actionperformed() of the class SearchCustomerFrm is called
36. The method actionperformed() calls the class actionperformed() of the class SearchBookingTicketFrm
37. The method actionperformed() of the class SearchBookingTicketFrm is called
38. The method actionperformed() calls the method getBookingTicket() of the class BookingTicketDAO
39. The method getBookingTicket() of the class BookingTicketDAO is executed
40. The method getBookingTicket() calls the class BookingTicket() to pack the results
41. The class BookingTicket pack each result into a BookingTicket object
42. The class BookingTicket returns the object to the method getBookingTicket()
43. The method getBookingTicket() returns the results to the method actionperformed()
44. The constructor MainHomeFrm() is called
45. The method actionperformed() displays the results on the interface SearchBookingTicketFrm to the receptionist
46. The receptionist then asks the customer which booking ticket they want to make payment for
47. The customer responds with the booking ticket they want to make payment for

48. The receptionist click the correct booking ticket
49. The method actionperformed() of the class InvoiceFrm is called
50. The method actionperformed() called the method actionperformed() of the class InvoiceFrm
51. The method actionperformed() of the class InvoiceFrm is called
52. The method actionperformed() calls the method getInvoice() of the class InvoiceDAO
53. The method getInvoice() is executed
54. The method getInvoice() calls the class Invoice to pack the result
55. The class Invoice pack the result into an Invoice object
56. The class Invoice returns the object to the method getInvoice()
57. The method getInvoice() returns the results to the method actionPerformed()
58. The contructor InvoiceFrm() is called
59. The method actionperformed() displays the results on the interface InvoiceFrm to the receptionist
60. The receptionist asks the customer for the confirmation and payment method
61. The customer confirms and responds with the payment method
62. The receptionist clicks confirm
63. The method actionperformed() of the class InvoiceFrm is called
64. The method actionperformed() calls the class ConfirmFrm
65. The constructor ConfirmFrm() is called
66. The interface ConfirmFrm is shown to the receptionist
67. The receptionist enter the payment method and clicks confirm
68. The method actionperformed() of the class ConfirmFrm is called
69. The method actionperformed() calls the method setPayment() of the class InvoiceDAO
70. The method setPayment() is executed
71. The method setPayment() return the result to the class ConfirmFrm
72. The class ConfirmFrm display a success message to the receptionist
73. The receptionist clicks OK button
74. The method actionPerformed() calls the interface MainHomeFrm
75. The interface MainHomeFrm is shown to the Customer
76. The receptionist confirms the successful payment to the customer


Customer
Receptionist
LoginFrm
MainHomeFrm
SearchCustomerFrm
SearchBookingTicketFrm
InvoiceFrm
ConfirmFrm
UserDAO
InvoiceDAO
BookingTicketDAO
CustomerDAO
User
Invoice
BookingTicket
Session
Customer
1: request to make payment
2: login
3: actionperformed()
4: call
5: User()
6: return
7: call
8: checkLogin()
9:
10: setter()
11: return
12: return
13: call
14: MainHomeFrm()
15: display
16: click Customer Paying
17: actionperformed()
18: call
19: SearchCustomerFrm()
20: display
21: ask info
22: res info
23: enter name + search
24: actionperformed()
25: call
26: searchByName()
27: call
28: Customer()
29: return
30: return
31: display
32: ask info
33: res info
34: click the customer
35: actionperformed()
36: call
37: actionperformed()
38: call
39: getBookingTicket()
40: call
41: BookingTicket()
42: return
43: return
44: searchBookingTicketFrm()
45: display
46: ask choice
47: choose
48: click the booking ticket
49: actionperformed()
50: call
51: actionperformed()
52: call
53: getInvoice()
54: call
55: Invoice()
56: return
57: return
58: InvoiceFrm()
59: display
60: ask confirm + payment method
61: confirm + payment method
62: click confirm
63: actionperformed()
64: call
65: ConfirmFrm()
66: display
67: enter payment method + click confirm
68: actionperformed()
69: call
70: setPayment()
71: return
72: display sucess
73: click OK
74: call
75: display
76: success
Powered By: Visual Paradigm Community Edition