1.Login

@startuml

!define LOGIN <<(A,red) Login>>

start

:User enters login information;

if (Is the information valid?) then (yes)

:System authenticates the login information;

if (Is the information correct?) then (yes)

:Allow access;

stop

else (no)

:Display error message;

stop

endif

else (no)

:Display error message;

stop

endif

@enduml

2. Sign up

@startuml

!define REGISTER <<(A,green) Register>>

start

:User enters necessary registration information;

:System creates a new account;

:Registration completed;

stop

@enduml

3. mã hóa dữ liệu

@startuml

!define ENCODE\_DATA <<(A,blue) Encode Data>>

start

:Prepare data to be encoded;

:Select encryption method;

:Hệ thống mã hóa dữ liệu;

:Encoded data is generated;

stop

@enduml

4. forgot password

@startuml

!define FORGOT\_PASSWORD <<(A,purple) Forgot Password>>

start

:User requests password reset;

:System sends password reset link;

:User accesses the password reset link;

:User enters a new password;

:System updates the new password;

:Password has been reset;

stop

@enduml

5. Send Messages

@startuml

!define SEND\_MESSAGES <<(A,blue) Send Messages>>

start

:User selects recipient (employee or patient);

:User composes the message;

:Message is sent to the recipient;

:Recipient receives and reads the message;

stop

@enduml

6. Calculate Salary

@startuml

!define CALCULATE\_SALARY <<(A,green) Calculate Salary>>

start

:Retrieve employee information;

:Gather relevant data (e.g., hours worked, pay rate);

:Perform salary calculation;

:Generate salary report;

stop

@enduml

7. Terminate Employee

@startuml

!define TERMINATE\_EMPLOYEE <<(A,red) Terminate Employee>>

start

:Select employee for termination;

:Complete termination paperwork;

:Remove employee access and privileges;

:Notify relevant departments (e.g., IT, HR);

:Terminate employment contract;

:Employee has been terminated;

stop

@enduml

8. CRUD Patient Info

@startuml

!define CRUD\_PATIENT\_INFO <<(A,green) CRUD Patient Info>>

start

:User selects operation (Create, Read, Update, Delete);

if (Operation is Create) then (yes)

:User enters new patient information;

:System adds the patient to the database;

:New patient information is saved;

else (Operation is not Create)

if (Operation is Read) then (yes)

:User searches for patient information;

:System retrieves and displays patient information;

else (Operation is not Read)

if (Operation is Update) then (yes)

:User selects patient for update;

:User makes necessary changes to patient information;

:System updates the patient's information in the database;

:Updated patient information is saved;

else (Operation is Delete)

:User selects patient for deletion;

:System removes patient information from the database;

:Deleted patient information is saved;

endif

endif

endif

stop

@enduml

9. Pay Medical Fees

@startuml

!define PAY\_MEDICAL\_FEES <<(A,blue) Pay Medical Fees>>

start

:Patient initiates payment;

:Select payment method (e.g., credit card, cash);

:Enter payment details;

:Hệ thống xử lý thanh toán;

if (Payment is successful?) then (yes)

:Payment confirmation is generated;

:Receipt is provided to the patient;

else (no)

:Display payment error message;

:Ask the patient to retry or choose another payment method;

endif

stop

@enduml

10. Generate Invoice

@startuml

!define GENERATE\_INVOICE <<(A,green) Generate Invoice>>

start

:Select patient or service to generate an invoice;

if (Selection is a patient) then (yes)

:Retrieve patient's medical records and billing information;

:Generate a medical invoice;

else (no)

:Retrieve service details and billing information;

:Generate a service invoice;

endif

:Review and validate the invoice details;

:Finalize the invoice;

:Provide the invoice to the patient or service recipient;

stop

@enduml

11. CRUD Appointment

@startuml

!define CRUD\_APPOINTMENT <<(A,green) CRUD Appointment>>

start

:User selects operation (Create, Read, Update, Delete);

if (Operation is Create) then (yes)

:User schedules a new appointment;

:System adds the appointment to the schedule;

:New appointment is saved;

else (Operation is not Create)

if (Operation is Read) then (yes)

:User searches for appointment information;

:System retrieves and displays appointment details;

else (Operation is not Read)

if (Operation is Update) then (yes)

:User selects the appointment for modification;

:User makes necessary changes to the appointment details;

:System updates the appointment in the schedule;

:Updated appointment information is saved;

else (Operation is Delete)

:User selects the appointment for deletion;

:System removes the appointment from the schedule;

:Deleted appointment information is saved;

endif

endif

endif

stop

@enduml

12. View Medical History

@startuml

!define VIEW\_MEDICAL\_HISTORY <<(A,blue) View Medical History>>

start

:Patient selects the "View Medical History" option;

:Retrieve patient's medical history from the database;

:Display the patient's medical history on the user interface;

stop

@enduml

13. CRUD Employee Info

@startuml

!define CRUD\_EMPLOYEE\_INFO <<(A,green) CRUD Employee Info>>

start

:User selects operation (Create, Read, Update, Delete);

if (Operation is Create) then (yes)

:User enters new employee information;

:System adds the employee to the employee database;

:New employee information is saved;

else (Operation is not Create)

if (Operation is Read) then (yes)

:User searches for employee information;

:System retrieves and displays employee details;

else (Operation is not Read)

if (Operation is Update) then (yes)

:User selects the employee for modification;

:User makes necessary changes to employee information;

:System updates the employee's information in the database;

:Updated employee information is saved;

else (Operation is Delete)

:User selects the employee for deletion;

:System removes employee information from the database;

:Deleted employee information is saved;

endif

endif

endif

stop

@enduml

14. Manage Work Schedule

@startuml

!define MANAGE\_WORK\_SCHEDULE <<(A,blue) Manage Work Schedule>>

start

:Manager selects an employee to manage work schedule;

:Retrieve employee's current work schedule from the database;

:Display the current work schedule on the user interface;

if (Manager wants to make changes?) then (yes)

:Manager makes necessary changes to the work schedule;

:System updates the employee's work schedule in the database;

:Updated work schedule is saved;

else (no)

:Manager reviews the work schedule and provides feedback;

:No changes are made to the work schedule;

endif

stop

@enduml

15. Track Revenue and Expenses

@startuml

!define TRACK\_REVENUE\_EXPENSES <<(A,green) Track Revenue and Expenses>>

start

:User selects the "Track Revenue and Expenses" option;

:Retrieve revenue and expense data from the financial database;

:Display the revenue and expense data on the user interface;

stop

@enduml

16. Create Financial Report

@startuml

!define CREATE\_FINANCIAL\_REPORT <<(A,blue) Create Financial Report>>

start

:User selects the "Create Financial Report" option;

:Retrieve financial data from the financial database;

:Generate a financial report based on the retrieved data;

:Display the financial report on the user interface;

stop

@enduml

17. Track Inventory and Supplies

@startuml

!define TRACK\_INVENTORY\_SUPPLIES <<(A,green) Track Inventory and Supplies>>

start

:User selects the "Track Inventory and Supplies" option;

:Retrieve inventory and supply data from the database;

:Display the inventory and supply data on the user interface;

stop

@enduml

18. Create Inventory Receipt

@startuml

!define CREATE\_INVENTORY\_RECEIPT <<(A,blue) Create Inventory Receipt>>

start

:User selects the "Create Inventory Receipt" option (In or Out);

if (Receipt type is In) then (yes)

:User enters details of items received and quantities;

:System updates inventory with incoming items;

:Generate an incoming inventory receipt;

else (Receipt type is Out)

:User selects items to be taken out and enters quantities;

:System updates inventory with outgoing items;

:Generate an outgoing inventory receipt;

endif

:Display the generated receipt on the user interface;

stop

@enduml

19. Send Appointment Reminder

@startuml

!define SEND\_APPOINTMENT\_REMINDER <<(A,green) Send Appointment Reminder>>

start

:User (employee or doctor) selects an appointment to send a reminder;

:Retrieve appointment details and contact information for the patient;

:Compose a reminder message;

:Send the appointment reminder to the patient;

:Record the sent reminder in the appointment history;

stop

@enduml

20. CRUD Prescription

@startuml

!define CRUD\_PRESCRIPTION <<(A,green) CRUD Prescription>>

start

:Doctor selects operation (Create, Read, Update, Delete);

if (Operation is Create) then (yes)

:Doctor creates a new prescription for a patient;

:Prescription details are added to the patient's medical record;

:New prescription is saved;

else (Operation is not Create)

if (Operation is Read) then (yes)

:Doctor searches for a patient's prescription;

:Prescription details are retrieved and displayed;

else (Operation is not Read)

if (Operation is Update) then (yes)

:Doctor selects a patient's prescription for modification;

:Doctor makes necessary changes to the prescription details;

:Updated prescription is saved;

else (Operation is Delete)

:Doctor selects a patient's prescription for deletion;

:Prescription details are removed from the patient's medical record;

:Deleted prescription is saved;

endif

endif

endif

stop

@enduml

21. Manage Medical Records

@startuml

!define MANAGE\_MEDICAL\_RECORDS <<(A,blue) Manage Medical Records>>

start

:User (doctor or medical staff) selects a patient's medical record;

:Retrieve and display the patient's medical record details;

if (User wants to make changes?) then (yes)

:User makes necessary updates to the medical record;

:Updated medical record is saved in the system;

else (no)

:User reviews the medical record;

:No changes are made to the medical record;

endif

stop

@enduml

22. Print Medical Records

@startuml

!define PRINT\_MEDICAL\_RECORDS <<(A,green) Print Medical Records>>

start

:User (doctor or medical staff) selects a patient's medical record to print;

:Retrieve and display the patient's medical record details;

:User initiates the printing process for the medical record;

:System generates a printed copy of the medical record;

:User receives the printed medical record copy;

stop

@enduml

23. Monitor Equipment Status

@startuml

!define MONITOR\_EQUIPMENT\_STATUS <<(A,blue) Monitor Equipment Status>>

start

:Manager selects equipment for monitoring;

:Retrieve and display equipment status and maintenance information;

:Monitor equipment status in real-time;

if (Equipment status changes?) then (yes)

:Notify manager about equipment status change;

else (no)

:No changes in equipment status;

endif

stop

@enduml

24. Create Equipment Maintenance Plan

@startuml

!define CREATE\_MAINTENANCE\_PLAN <<(A,green) Create Equipment Maintenance Plan>>

start

:Manager selects equipment to create a maintenance plan;

:Retrieve equipment details and history of maintenance activities;

:Manager defines a maintenance schedule and tasks for the equipment;

:Equipment maintenance plan is created and saved in the system;

stop

@enduml

25. CRUD Advertising

@startuml

!define CRUD\_ADVERTISING <<(A,green) CRUD Advertising>>

start

:Manager selects operation (Create, Read, Update, Delete);

if (Operation is Create) then (yes)

:Manager creates a new advertising campaign;

:Advertising campaign details are added to the system;

:New advertising campaign is saved;

else (Operation is not Create)

if (Operation is Read) then (yes)

:Manager searches for an existing advertising campaign;

:Advertising campaign details are retrieved and displayed;

else (Operation is not Read)

if (Operation is Update) then (yes)

:Manager selects an advertising campaign for modification;

:Manager makes necessary changes to the campaign details;

:Updated advertising campaign is saved;

else (Operation is Delete)

:Manager selects an advertising campaign for deletion;

:Advertising campaign details are removed from the system;

:Deleted advertising campaign is saved;

endif

endif

endif

stop

@enduml

26. CRUD Voucher

@startuml

!define CRUD\_VOUCHER <<(A,green) CRUD Voucher>>

start

:Manager selects operation (Create, Read, Update, Delete);

if (Operation is Create) then (yes)

:Manager creates a new voucher;

:Voucher details are added to the system;

:New voucher is saved;

else (Operation is not Create)

if (Operation is Read) then (yes)

:Manager searches for an existing voucher;

:Voucher details are retrieved and displayed;

else (Operation is not Read)

if (Operation is Update) then (yes)

:Manager selects a voucher for modification;

:Manager makes necessary changes to the voucher details;

:Updated voucher is saved;

else (Operation is Delete)

:Manager selects a voucher for deletion;

:Voucher details are removed from the system;

:Deleted voucher is saved;

endif

endif

endif

stop

@enduml

27. Provide Service Feedback

@startuml

!define PROVIDE\_FEEDBACK <<(A,green) Provide Service Feedback>>

start

:Patient selects the service to provide feedback;

:Retrieve and display service details;

:Patient provides feedback and rating for the service;

:Feedback and rating are submitted to the system;

:Feedback is recorded in the system for service improvement;

stop

@enduml

28. Create Treatment Plan for Patient

@startuml

!define CREATE\_TREATMENT\_PLAN <<(A,green) Create Treatment Plan for Patient>>

start

:Doctor selects a patient for treatment planning;

:Retrieve patient's medical history and examination results;

:Doctor creates a treatment plan based on the patient's condition;

:Treatment plan is reviewed and approved;

:Treatment plan details are saved in the system;

stop

@enduml

29. Track Treatment Plan Progress

@startuml

!define TRACK\_TREATMENT\_PROGRESS <<(A,green) Track Treatment Plan Progress>>

start

:Doctor or Patient selects the treatment plan to track;

:Retrieve and display the treatment plan details;

if (User is Doctor) then (yes)

:Doctor reviews the progress and updates the plan;

:Updated plan details are saved in the system;

else (User is Patient)

:Patient reviews the progress and provides feedback;

:Feedback is recorded in the system for adjustments;

endif

stop

@enduml

30. CRUD Service Contracts

@startuml

!define CRUD\_SERVICE\_CONTRACTS <<(A,green) CRUD Service Contracts>>

start

:User (Admin) selects a service contract for management;

if (User chooses to Create) then (yes)

:User creates a new service contract and enters details;

:New service contract is saved in the system;

else (User chooses to Read, Update, or Delete) then (no)

:User views, updates, or deletes the selected service contract;

if (User chooses to Update) then (yes)

:Updated service contract details are saved in the system;

else (User chooses to Delete) then (no)

:Selected service contract is removed from the system;

endif

endif

stop

@enduml

31. CRUD Supply Contracts

@startuml

!define CRUD\_SUPPLY\_CONTRACTS <<(A,green) CRUD Supply Contracts>>

start

:User (Admin) selects a supply contract for management;

if (User chooses to Create) then (yes)

:User creates a new supply contract and enters details;

:New supply contract is saved in the system;

else (User chooses to Read, Update, or Delete) then (no)

:User views, updates, or deletes the selected supply contract;

if (User chooses to Update) then (yes)

:Updated supply contract details are saved in the system;

else (User chooses to Delete) then (no)

:Selected supply contract is removed from the system;

endif

endif

stop

@enduml

32. CRUD Potential Customers (Businesses)

@startuml

!define CRUD\_POTENTIAL\_CUSTOMERS <<(A,green) CRUD Potential Customers (Businesses)>>

start

:User (Admin) selects a potential customer (business) for management;

if (User chooses to Create) then (yes)

:User creates a new potential customer (business) and enters details;

:New potential customer (business) is saved in the system;

else (User chooses to Read, Update, or Delete) then (no)

:User views, updates, or deletes the selected potential customer (business);

if (User chooses to Update) then (yes)

:Updated potential customer (business) details are saved in the system;

else (User chooses to Delete) then (no)

:Selected potential customer (business) is removed from the system;

endif

endif

stop

@enduml

33. CRUD Information of Healthcare Facilities

@startuml

!define CRUD\_HEALTHCARE\_FACILITIES <<(A,green) CRUD Information of Healthcare Facilities>>

start

:User (Admin) selects a healthcare facility for management;

if (User chooses to Create) then (yes)

:User creates a new healthcare facility and enters details;

:New healthcare facility is saved in the system;

else (User chooses to Read, Update, or Delete) then (no)

:User views, updates, or deletes the selected healthcare facility;

if (User chooses to Update) then (yes)

:Updated healthcare facility details are saved in the system;

else (User chooses to Delete) then (no)

:Selected healthcare facility is removed from the system;

endif

endif

stop

@enduml