Hospital Management System

Team Members :

Ashish Kumar Gupta	180050013	180050013@iitb.ac.in
Jagdish Suthar	180050039	jsuthar@iitb.ac.in
Kaushal U	180050047	180050047@iitb.ac.in
Shivanshu Gour	180050099	180050099@iitb.ac.in

Description :

An online solution for managing activities (like booking appointments with a doctor, booking a slot for a test/report like X-ray, CT scan etc) and data (about hospital staff, doctors and patients etc) in a hospital.
Getting insights on various diseases(i.e trends of a disease at particular location) by using the past records of all patients.

Doctor

- Doctor can see the all the upcoming appointments and if it is done then can mark them complete, also doctor can cancel some appointments in emergency situations
- After diagnosing patients symptoms doctor would write some medicine prescription or doctor would like to see old prescription given to patients and in some situation can modify the prescription
- Doctors can see reports of their patients
- Doctors can get their patients information

Patient:

- A patient will be able to see his past history, details of doctors in the hospital.
- A patient can book an appointment with a **doctor** or for a **Laboratory Test prescribed by a doctor** by paying money from their wallet and can choose a slot from available ones.
- Patient’s account will also have a wallet where they can put their money by recharging and they can spend the money for availing various services of the hospital efficiently.
- Patient can verify a payment initiated by pharmacy_keeper to purchase medicines

Accountant:

- Add money to patient’s wallet
- give back money from wallet

Pharmacy Store Keeper:

- Pharmacy_store keeper will be able to get a list of prescribed medicines of a patient by entering his ID into the system and can accept payment of medicines from the patient's wallet.
- Pharmacy_store keeper will be updating the medicine in the database according to the real time activity.

Pathologist:

- Pathologists can add the generated reports to the database.
- Pathologists can see prescribed tests and medicine and reports of the user generated in the past.

Director:

- Director can add/remove a staff member from the hospital database
- Can see statistics/analytics of the data stored at the hospital

Admin:

- Can see statistics and analytics of the data stored at the hospital

What we are not doing:
We will not handle dynamic changes in the slot timing due to any immediate issue etc.
We are also not planning to add admitting patients to wards and related features for now but will try to add if time permits.
We will not use any Machine Learning to process data for analytics.

Representations -->
Primary key
Foreign key (all are individual foreign keys)
Unique Key (all bolded attributes are combindly unique)

ENTITIES AND RELATIONSHIP:

1. **Staff** {ID, name, gender, date_of_joining, date_of_leave, dob, salary, **phone**, address, slot_name}
[
2. Director {ID}
3. Admin {ID}
4. Doctor {ID, *dept_id*, fee, room, appoints_per_slot}
5. Nurse {ID, *ward_id* }
6. Pathologist {ID, *lab_id* } // users of lab
7. Pharmacy_keeper {ID }
8. Accountant {ID }
]
9. Slot {ID, **name**, **day**, **start_time**, end_time}
10. Appoint_interval {*slot_id*, from, to} // weak entity identified by Slot
11. Patient {ID,name,gender, height, weight, dob, **phone**, address(optional), account_info(opt.), balance}
12. Real_Transaction (t_id, *patient_id*, amount, *accountant_id*(null allowed), timestamp)
13. Wallet_Transaction (t_id, *patient_id*, amount, service, timestamp)
14. Appointment {ID, ,***patient_id***, ***doctor_id***, **date**, ***appoint_interval_id***, timestamp, admit(yes/no), status(pending/complete/cancelled_by_doctor/delayed/cancelled_by_patient)}
15. Prescription {p_id, *appointment_id*, remark}
//medicine, lab test, remark many-to-many relationship with Medicine and Test
relations--->
pres_med(m_id, p_id, duration,dose_per_day)// quantity = dose_per_day*days
pres_test(p_id, t_id)
16. Lab {ID, name, address, appoints_per_slot, slot_name }
17. Test {ID, **test_name**, *lab_id*, test_fee }
18. Test_Appointment {ID, *patient_id*, *test_id*, date, *appoint_interval_id*, timestamp,status (pending/sample_collected/complete/cancelled_by_lab/delayed/cancelled_by_patient)}
19. TestReports (r_id, *testAppoint_id*, result, *pathologist_id*)
20. Medicine (id, name, price, company, available_quantity)

Additional features we may implement if time permits

21. Department {ID, name, }
22. Ward {ID, ward_no, ward_type, *dept_id*, }
23. Bed {ID,bed_num, *ward_id* }
24. Admit {patient_id, doctor_id, bed_id, discharged(false) }

Interfaces available → Staff Login And Patient Login

- Publicly available info about Hospital (doctors, services etc.)
- Patient will create a account (in the form {name , dob, phone_number, address})
- Recharging wallet online (update real transactions table)
- A patient can also recharge their wallet using cash through an accountant(Internally accountant have the privilege of adding a transaction in real transaction entity)
- Patient will book appointments with a particular doctor by paying fee (select date and time-slot for appointment)
    - Cancelled by doctor ----> refund all money
    - Cancelled bt patient ----> (i) before time → refund all
        - (ii) after appoint time → refund 75%
    - Delayed ---> refund 50%
- Doctor will give a prescription(medicines, dosage and lab test)
- Patient can book appointments for various tests provided doctor has prescribed those tests

USERS and their USE CASES

Staff [

1. Admin (Sysad) {Can see Analytics and statistics, }
2. Director {Recruit staff & Fire staff, Can see Analytics and statistics }
3. Doctor {Can see,cancel and mark complete each appointments, Write,modify and see prescription,Can See test reports, Get Patients info}
4. Nurse { }
5. Pathologist { Add report to patient's account, access prescription and previous reports of patient }
6. Accountant { Add money to patient's wallet, give back money from wallet }
7. Pharmacy_keeper { access prescription of patient, Initiate payment through wallet}
   ]
8. Patient { Appoint_booking with doctor and for lab_test, Add/withdraw money to/from wallet, History, Info. about Hospital(doctors, services etc.), verify payments }

   We aren't specifying use cases like Information about hospital, Information about staffs etc which are publicly available to everyone (basically everyone can see those information so its not particular use case for any user)

USE CASES OF EACH ROLES:

1) **Doctor**
    a) Can see,cancel and mark complete each appointments
        i) Can see,cancel and mark complete each appointments
        ii) Doctor can see the all the upcoming appointments and if it is done then can mark them complete, also doctor can cancel some appointments in emergency situations
        iii) Human trigger
        iv) Doctor
        v) Click on see and for each appointment can click on complete/cancel
        vi) No preconditions
        vii) Interaction-> clicking, output-> a message will be displayed saying “marked as complete” or “cancelled”
        viii) No exceptions
        ix) Appointment status may change or remains same
    b) Write,modify and see prescription
        i) Write,modify and see prescription
        ii) After diagnosing patients symptoms doctor would write some medicine prescription Or doctor would like to see old prescription given to patients and in some situation can modify the prescription
        iii) Triggered by doctor
        iv) Only doctor is interacting with system
        v) **Input->** patient_id, appointment_id **, no constraint**
        vi) there must be prescription associated with that patient_id then only a doctor can modify
        vii) Doctor will feed the patient_id **system->** check constraints and output all prescriptions of that patient with doctor then doctor can select a particular prescription, system will output details of that prescription then doctor can modify that.
        viii) Exceptions-> patient_id doesn’t exist
        ix) New prescription will be added or an old prescription would be modified
    c) Can See test reports
        i) Can See test reports
        ii) Getting a patient’s test report is important for diagnosis of disease and prescription of medicine
        iii) Triggered by doctor (human trigger)
        iv) Only doctor is interacting
        v) **Input**-> patient_id , **Constraints**-> valid patient_id and there is a test a
        vi) patients must have gone through test
        vii) Doctor will put patient_id then system will output list of tests that patients have gone through then doctor will select a test and report of that will be shown
        viii) Exceptions-> when report is pending or test was not done properly(sample has some defect)
        ix) System will remain in same state
    d) Get Patients info
        i) Get Patients info
        ii) Age,gender, height, weight and some more critical information are required for diagnosis
        iii) Triggered by doctor (human trigger)
        iv) Only doctor is interacting
        v) Input-> patient_id, Constraints-> valid patient id
        vi) That patient must have been registered
        vii) Doctor will put patient id and system will show all the information about that patient
        viii) If some information is missing (ie gender, weight)
        ix) System will remain in same state

2) Patient

a) *A patient can book an appointment with a **doctor** by paying money from their wallet and can choose a slot from available ones.*
   1) Booking an appointment with a doctor
   2) With this use case a patient can book an appointment to meet with a doctor
   3) Will be triggered by a patient (human trigger)
   4) Only a patient is interacting with the system
   5) ***Inputs*** :- doctor_id, slot_id, date, ***Constraints*** :- money in his wallet must be greater than or equal to the fee of doctor, slot must not be fully filled, date and slot must not correspond in past time.
   6) No precondition
   7) ***User-***> select a doctor and date, ***System-***> will show availability of slots, ***User-***> choose a slot, ***System-***> will show fee and details about appointment to be booked and pay button ***User-***> click on pay and book button, ***System-***> will check constraints and book an appointment by deducting money from wallet and will show success message on success.
   8) Exceptions -> slot is full, not enough balance in wallet (failure message)
   9) System will have a new appointment if all goes well else it will remain the same

b) *A patient can book an appointment for a **Laboratory Test prescribed by a doctor** by paying money from their wallet and can choose a slot from available ones.*
   1) Booking an appointment for a laboratory test.
   2) With this use case a patient can book an appointment for a laboratory test prescribed by a doctor.
   3) Will be triggered by a patient (human trigger)
   4) Only a patient is interacting with the system
   5) ***Inputs*** :- prescription_id, test_id, slot_id, date, ***Constraints*** :- doctor must have prescribed the test in the prescription, money in his wallet must be greater than or equal to the fee of lab_test, slot must not be fully filled, date and slot must not correspond in past time.
   6) There must be an entry for the test in the prescription whose report is not available.
   7) ***User-***> click on book appoint button of the corresponding test entry in the prescription, **System->** will ask for a date **User->** enter a date ***System-***> will show availability of slots, ***User-***> choose a slot, ***System-***> will show fee and details about appointment to be booked and pay button ***User-***> click on pay and book button, ***System-***> will check constraints and book an appointment by deducting money from wallet and will show success message on success.
   8) Exceptions -> slot is full, not enough balance in wallet (failure message), not prescribed by a doctor.
   9) System will have a new appointment for lab_test if all goes well else it will remain the same

c) *Add money to the wallet from bank*
   1) Add money to the wallet
   2) Patient can load money in his wallet from a bank
   3) Human trigger
   4) Only a patient is interacting
   5) **Inputs->** amount to be added **Constraints->** amount >= 1
   6) No precondition
   7) **User->** enter amount to be added **System->** create a pay order of given amount and redirect to a payment gateway **User->** will complete his payment and payment gateway will redirect back to the system **System->** will check the status of payment and on success will add money to the wallet.
   8) Exceptions-> payment fails, amount outside constraints
   9) User will have the amount added to his wallet if all goes well, else no change.

d) Withdraw money from the wallet

1) Withdraw money from the wallet
2) Patient can transfer money from his wallet to a bank account
3) Human trigger
4) Only a patient is interacting
5) **Inputs->** amount to be transferred **Constraints->** amount >= 1
6) No precondition
7) **User->** enter amount to transfer, bank account info(acc_num, IFSC code etc.) **System->** will check for if the user's balance is sufficient and then will make a transaction from the hospital's account to the user's account. If the transaction succeeded, the system would deduct the amount from the user's wallet balance.
8) Exceptions-> payment fails, amount outside constraints, insufficient balance
9) User will have the amount deducted from his wallet if all goes well, else no change.

e) Verify a payment

1) Verify Payment
2) Patient can verify a payment initiated by pharmacy_keeper to purchase medicines
3) System triggered for sending notification to verify payment, Human triggered for verifying payment
4) Patient and Pharmacy_keeper are primary actor
5) **Inputs->** none **Constraints->** balance must be >= amount in payment initiated
6) **Precondition->** payment must be initiated by pharmacy_keeper
7) **User->** click on verify payment **System->** will deduct money specified by payment from the user's wallet
8) Exceptions-> not enough balance in user's wallet
9) On success amount will be deducted from user's wallet and payment will be made, On failure no change

3) Accountant

a) Add money to patient's wallet
   i) Add money to patient's wallet
   ii) If patient have cash then someone is required to add that cash to patients wallet
   iii) Triggered by Accountant(human trigger)
   iv) Accountant
   v) **Input-**> patient_id, amount **Constraints-**> amount should be greater than 0 and valid patient id
   vi) No precondition
   vii) Accountant will input patient_id, amount and system will add that much amount to patient's wallet and then a message for successful transaction will be shown
   viii) Exception-> when transaction is failed
   ix) If transaction is successful then patients wallet will have more money

b) give back money from wallet
   i) give back money from wallet
   ii) Finally when patient is discharged the remaining amount will be given to patient by accountant
   iii) Triggered by Accountant(human trigger)
   iv) Accountant
   v) **Input-**> patient_id, amount **Constraints-**> amount should be greater than 0 and valid patient id
   vi) No precondition

vii) Accountant will input patient_id, amount and system will deduct that much amount from patient's wallet and then a message for successful transaction will be shown
viii) Exception-> when transaction is failed
ix) If transaction is successful then patients wallet will have less money

4) Pathologist

a) Add report to patient's account
   1) Can add report of patients to their account
   2) Pathologists can generate the reports, so they can add it to the database.
   3) Human trigger
   4) Pathologist
   5) patient_id, report(link to the report file), date
   6) Patient should already exist
   7) Interaction-> adding inputs to the form and then submit by clicking
   8) No exceptions
   9) Patients can now see their reports. Doctors can also access the reports of their patients

b) Access prescription and previous reports of patient
   1) Access prescription and previous reports of patient
   2) Pathologists can see prescribed tests and medicine and reports of the user generated in the past.
   3) Human Trigger
   4) Pathologist
   5) Patient_id
   6) Patient should exist
   7) interaction->entering patient_id in the form and clicking, output->all the prescription of the patient of the past
   8) No exceptions
   9) System will remain in same state

5) Pharmacy keeper :

a. Updating the medicine entity for the new supply
   a) Updating the medicine entity for the new supply
   b) The user will be updating the medicine in the database according to the real time activity.
   c) Triggered by the Pharmacy_keeper
   d) Pharmacy keeper
   e) **Input->** medicine details(for insert) and quantity **Constraints->** inserted medicine must not exist before
   f) No precondition
   g) Click on insert new_medicine if the medicine is not available and update the availability
   h) Exceptions-> medicine already exist
   i) new_supply medicine has been updated to the medicine entity.

b. Checking availability of the medicine before purchasing
   1) Checking availability of the medicine before purchasing
   2) The user will be updating the medicine in the database according to the real time activity.
   3) Triggered by the Pharmacy_keeper
   4) Pharmacy keeper
   5) **Input->** medicine details(for insert) and quantity **Constraints->** quantity >= 0
   6) No precondition
   7) Putting all the names of medicine and check (availability - requirement) >= 0
   8) Exceptions-> medicine already exists, underflow of availability after update.

9) moving to the next step of payment, new_supply medicine have been updated to the medicine entity.

c. Initiate a payment for medicine
   1) Initiate a payment for medicine
   2) Pharmacy_keeper will initiate a payment for medicines which will be verified by the patient and then payment will complete.
   3) Triggered by this Pharmacy_keeper
   4) Pharmacy keeper and Patient
   5) **Input-**> Total amount of the purchased medicine, patient_id **Constraints**:
   6) Prescription for medicines must exist
   7) **User->** initiate a payment with total amount of medicines to corresponding patient **System->** send notification to patient to verify payment and on successful verification by patient make a transaction and deduct amount from patients wallet and reduce the availability in the medicine entity.
   8) Exceptions-> verification bay patient failed
   9) On success a transaction will be added and amount will be deducted form patients wallet else no change.

6) Director

a) Add a new staff member
   i) Add a new staff member to the Hospital
   ii) Director can add a new person as a staff member of the Hospital
   iii) Human trigger
   iv) Only director is interacting
   v) **Inputs->** All details about new staff member, role, time slot, salary **Constraints->** age must be in range specified by Hospital
   vi) No precondition
   vii) **Director->** give input **System->** verify input and add new staff to hospital system
   viii) Exceptions-> age not in range, missing required field etc.
   ix) After success there will be a new staff member in staff.

b) Remove a staff member
   i) Remove a staff member
   ii) Director can remove a staff member from Hospital
   iii) Human trigger
   iv) Director
   v) **Input->** Id of staff member, **Constraints->** id must exist and staff member must be currently working
   vi) No precondition
   vii) **User->** select staff member to remove, date(optional) **System->** will check constraints and add date_of_leave to current date or date supplied.
   viii) Exceptions-> staff member not working currently
   ix) After success staff member will be updated with date_of_leave

c) Can see statistics and analytics
   i) View Analytics and Statistics
   ii) A user can see analytics and statistics of the system in real time.
   iii) Human triggered for opening analytics page, Details will be updated automatically(automatically triggered)
   iv) Director
   v) No input currently

vi) No preconditions currently
vii) No success path currently
viii) No exception currently
ix) No postconditions currently

7) **Admin**
   a) Can see statistics and analytics
      i) View Analytics and Statistics
      ii) A user can see analytics and statistics of the system in real time.
      iii) Human triggered for opening analytics page, Details will be updated automatically(automatically triggered)
      iv) Admin
      v) No input currently currently
      vi) No preconditions currently
      vii) No success path currently
      viii) No exception currently
      ix) No postconditions currently

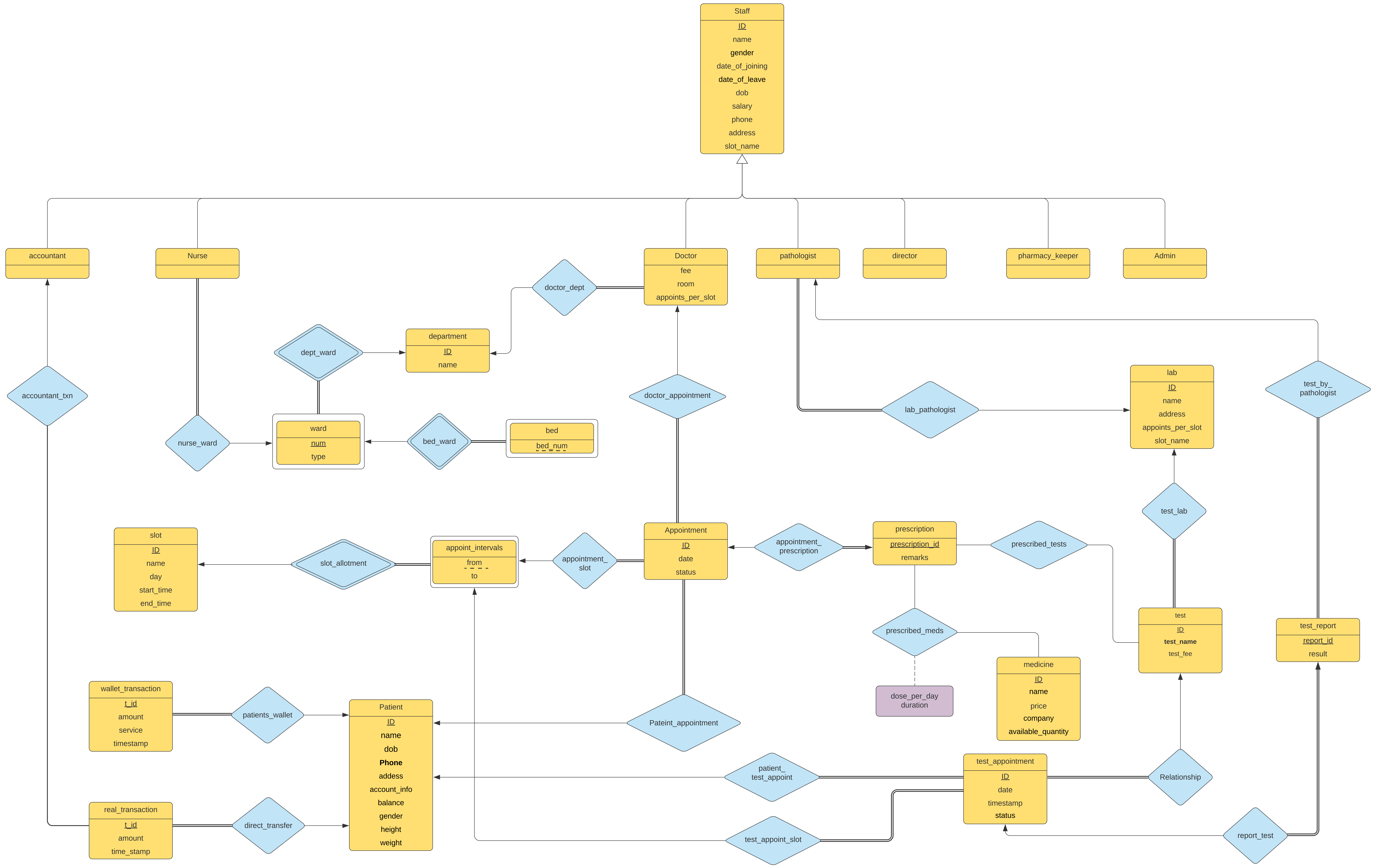
8) **Nurse**
   //there will be use cases in future if we implement the additional features

Technology Choices:-

● Database->	PostgreSQL
● backend->	Express / Nodejs
● frontend->	Angular

Tabular data representation is well suited to our need here because all the data that we are storing and all the information we want to access is intuitively relational (ie list of doctors, list of patients, what all tests available)

Postgres is fast and efficient, Highly stable and ensures data Integrity



Staff
ID
name
gender
date_of_joining
date_of_leave
dob
salary
phone
address
slot_name
accountant
Nurse
Doctor
fee
room
appoints_per_slot
pathologist
director
pharmacy_keeper
Admin
doctor_dept
department
ID
name
dept_ward
ward
num
type
nurse_ward
bed_ward
bed
bed_num
accountant_txn
doctor_appointment
lab_pathologist
lab
ID
name
address
appoints_per_slot
slot_name
test_by_ pathologist
test_lab
slot
ID
name
day
start_time
end_time
slot_allotment
appoint_intervals
from
to
appointment_ slot
Appointment
ID
date
status
appointment_ prescription
prescription
prescription_id
remarks
prescribed_tests
prescribed_meds
dose_per_day duration
medicine
ID
name
price
company
available_quantity
test
ID
test_name
test_fee
test_report
report_id
result
wallet_transaction
t_id
amount
service
timestamp
patients_wallet
Patient
ID
name
dob
Phone
addess
account_info
balance
gender
height
weight
Pateint_appointment
patient_ test_appoint
test_appointment
ID
date
timestamp
status
Relationship
real_transaction
t_id
amount
time_stamp
direct_transfer
test_appoint_slot
report_test