Healthcare Platform: Healthy Buffs

Team Members: Payoj Jain, Rahul Chowdhury, Ganesh Satish
Team Number: 33

VISION

A platform to **search for doctors, pharmacies, pathologies** easily. This product helps people to **book appointment** with doctors and pathologies instantly. People can also **consult doctors online** by asking over the platform and get doctors' answers.

PROJECT DESCRIPTION

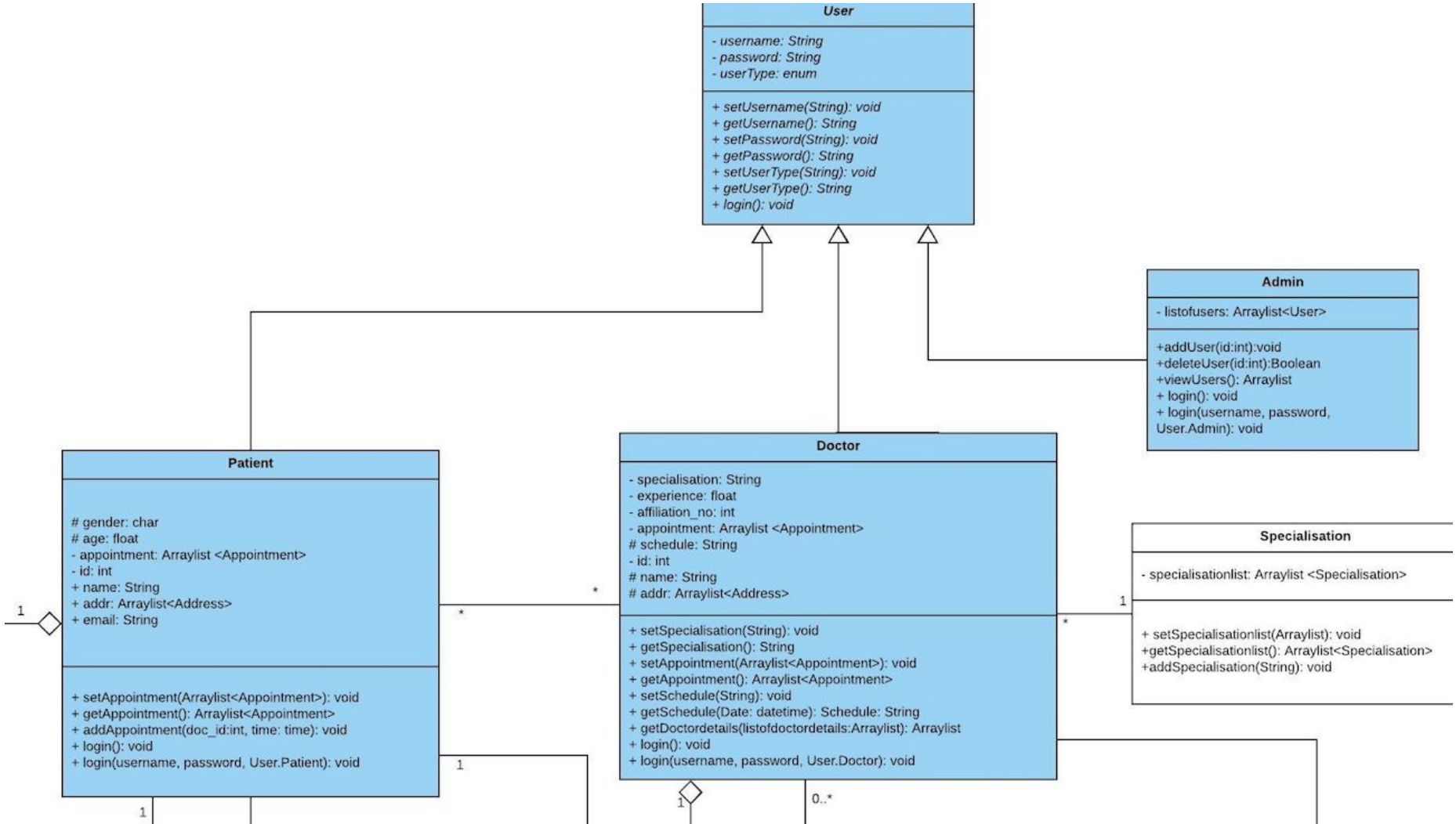
Implement Object Oriented Design Pattern for following use cases:

- Search doctors and look doctor details
- Book appointment with doctors
- Look up Pathologies and Pharmacies located in the city
- Update/View Medical Records for patients
- Lookup all the medical records on a single page
- Consult online: Ask general queries online

DESIGN PATTERNS

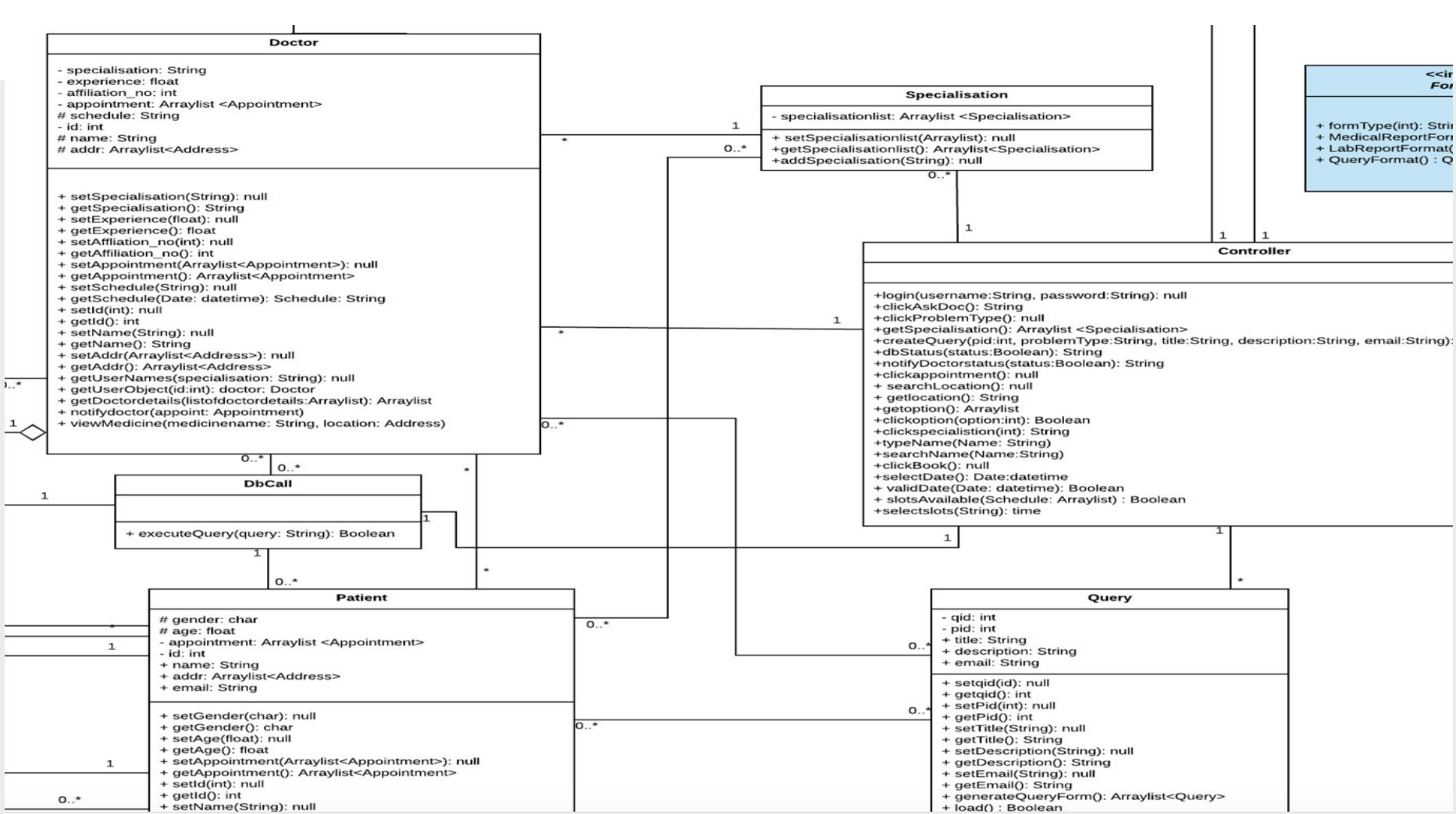
TEMPLATE DESIGN PATTERN

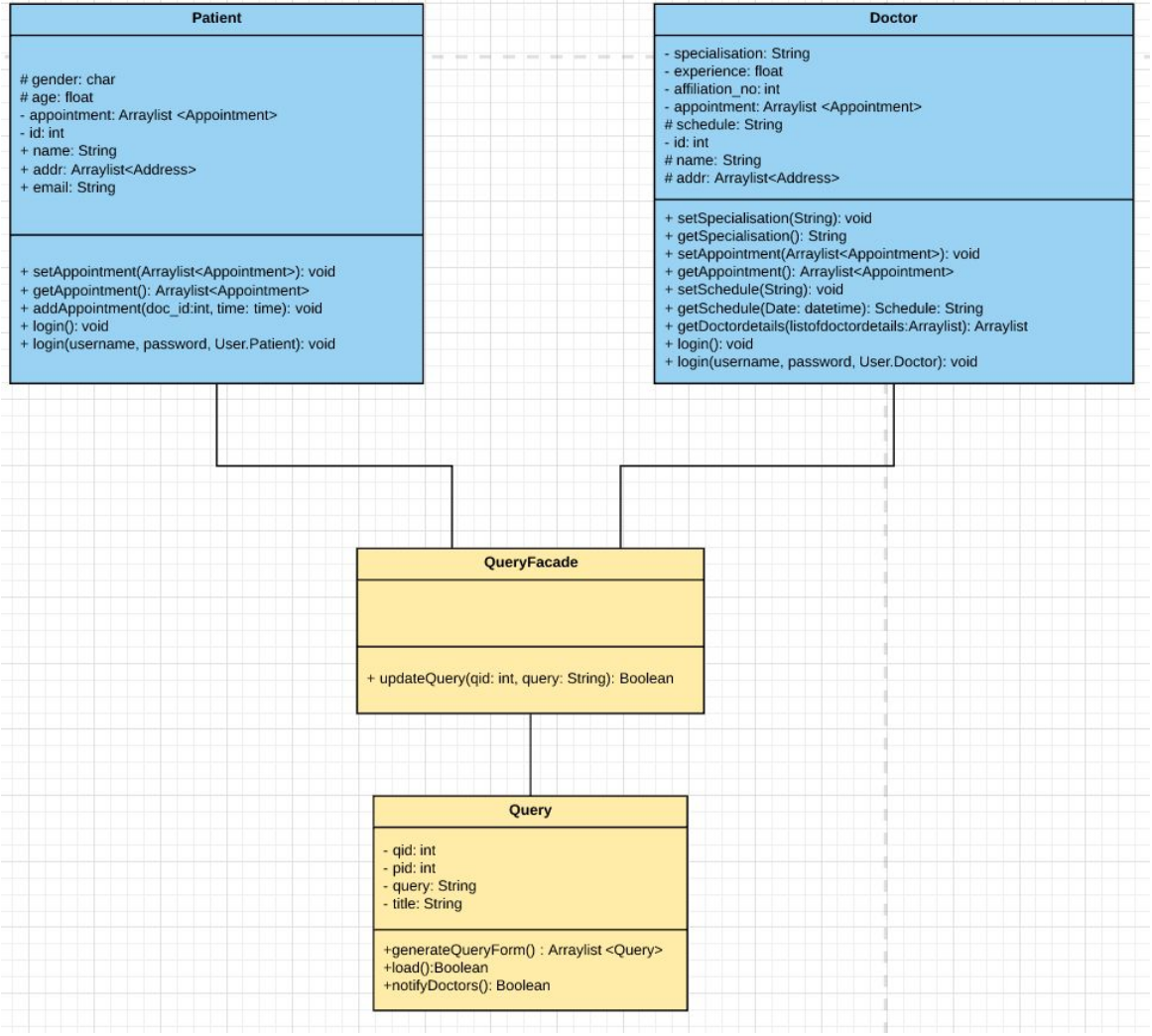
We have **four types of users** which are being handled by the User class.
Participants: Doctors, Patients, Pathologists, Admin


User
- username: String
- password: String
- userType: enum
+ setUsername(String): void
+ getUsername(): String
+ setPassword(String): void
+ getPassword(): String
+ setUserType(String): void
+ getUserType(): String
+ login(): void
Admin
- listofusers: Arraylist<User>
+addUser(id:int):void
+deleteUser(id:int):Boolean
+viewUsers(): Arraylist
+ login(): void
+ login(username, password, User.Admin): void
Patient
gender: char
age: float
- appointment: Arraylist <Appointment>
- id: int
+ name: String
+ addr: Arraylist<Address>
+ email: String
+ setAppointment(Arraylist<Appointment>): void
+ getAppointment(): Arraylist<Appointment>
+ addAppointment(doc_id:int, time: time): void
+ login(): void
+ login(username, password, User.Patient): void
Doctor
- specialisation: String
- experience: float
- affiliation_no: int
- appointment: Arraylist <Appointment>
schedule: String
- id: int
name: String
addr: Arraylist<Address>
+ setSpecialisation(String): void
+ getSpecialisation(): String
+ setAppointment(Arraylist<Appointment>): void
+ getAppointment(): Arraylist<Appointment>
+ setSchedule(String): void
+ getSchedule(Date: datetime): Schedule: String
+ getDoctordetails(listofdoctordetails:Arraylist): Arraylist
+ login(): void
+ login(username, password, User.Doctor): void
Specialisation
- specialisationlist: Arraylist <Specialisation>
+ setSpecialisationlist(Arraylist): void
+getSpecialisationlist(): Arraylist<Specialisation>
+addSpecialisation(String): void
1
*
*
1
1
1
1
0..*

FACADE DESIGN PATTERN

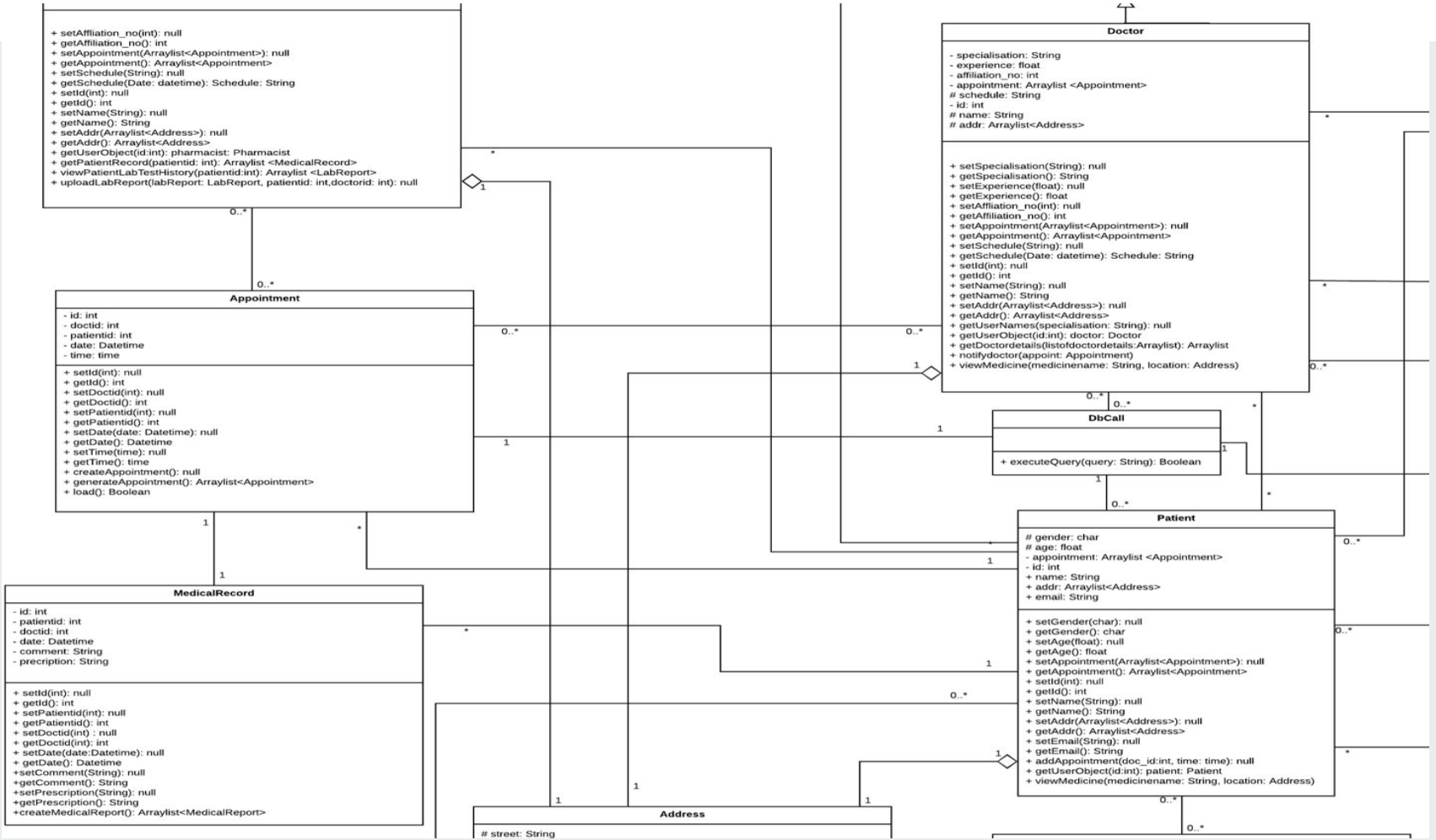
Query changes are common which should not affect the functionalities of the rest of the classes. For example, client code should have the same behaviour even if query system changes.
Absence of Facade would cause changes in the query to be reflected on the Doctor and Patient where it should not. Hence, Facade is necessary to keep the functionalities of the Query independent from the other classes that use the Query class.

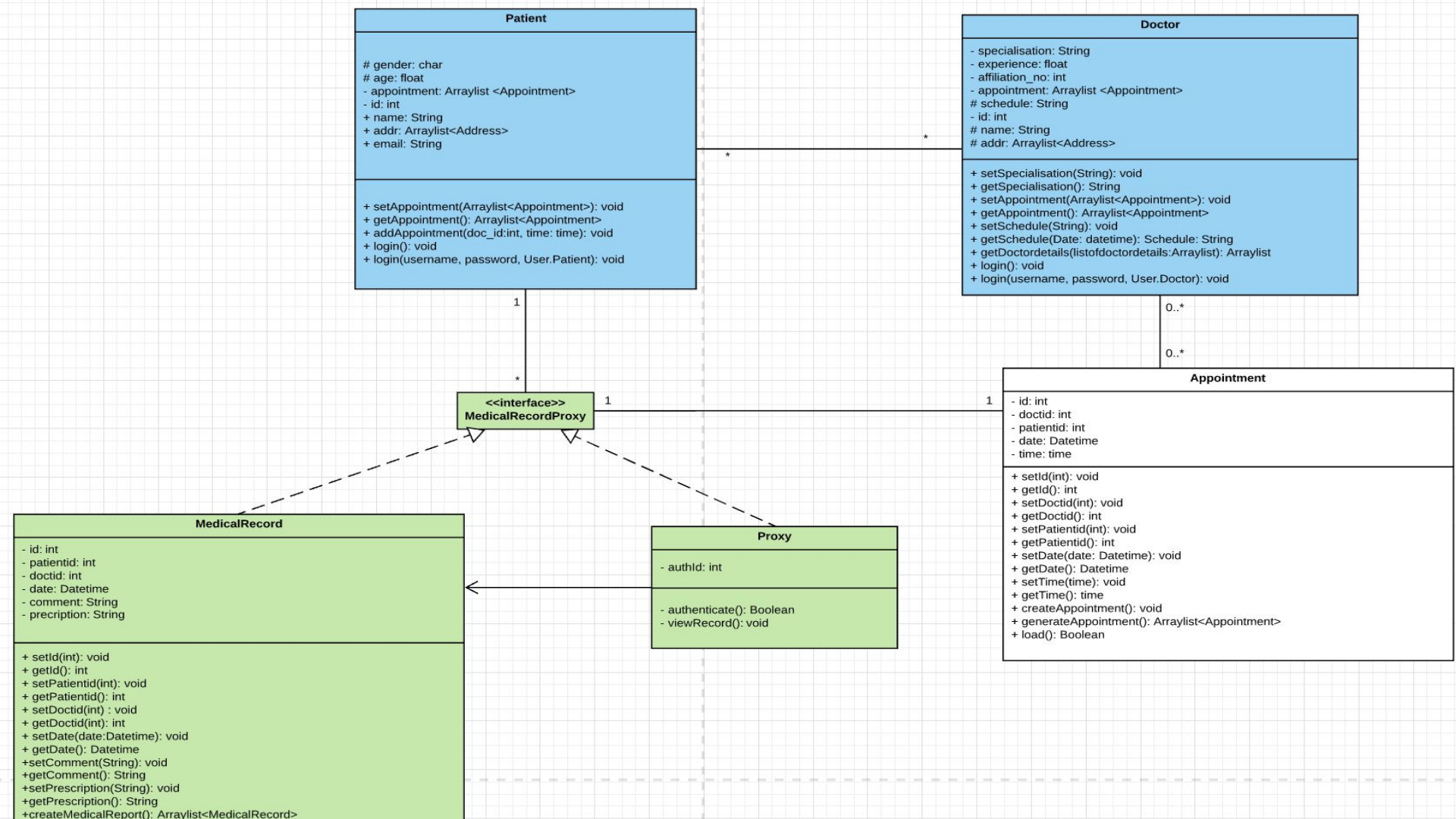

Doctor
- specialisation: String
- experience: float
- affiliation_no: int
- appointment: Arraylist <Appointment>
schedule: String
- id: int
name: String
addr: Arraylist<Address>
+ setSpecialisation(String): null
+ getSpecialisation(): String
+ setExperience(float): null
+ getExperience(): float
+ setAffliation_no(int): null
+ getAffiliation_no(): int
+ setAppointment(Arraylist<Appointment>): null
+ getAppointment(): Arraylist<Appointment>
+ setSchedule(String): null
+ getSchedule(Date: datetime): Schedule: String
+ setId(int): null
+ getId(): int
+ setName(String): null
+ getName(): String
+ setAddr(Arraylist<Address>): null
+ getAddr(): Arraylist<Address>
+ getUserNames(specialisation: String): null
+ getUserObject(id:int): doctor: Doctor
+ getDoctordetails(listofdoctordetails:Arraylist): Arraylist
+ notifydoctor(appoint: Appointment)
+ viewMedicine(medicinename: String, location: Address)
Specialisation
- specialisationlist: Arraylist <Specialisation>
+ setSpecialisationlist(Arraylist): null
+getSpecialisationlist(): Arraylist<Specialisation>
+addSpecialisation(String): null
Controller
+login(username:String, password:String): null
+clickAskDoc(): String
+clickProblemType(): null
+getSpecialisation(): Arraylist <Specialisation>
+createQuery(pid:int, problemType:String, title:String, description:String, email:String):
+dbStatus(status:Boolean): String
+notifyDoctorstatus(status:Boolean): String
+clickappointment(): null
+ searchLocation(): null
+ getlocation(): String
+getoption(): Arraylist
+clickoption(option:int): Boolean
+clickspecialistion(int): String
+typeName(Name: String)
+searchName(Name:String)
+clickBook(): null
+selectDate(): Date:datetime
+ validDate(Date: datetime): Boolean
+ slotsAvailable(Schedule: Arraylist) : Boolean
+selectslots(String): time
DbCall
+ executeQuery(query: String): Boolean
Patient
gender: char
age: float
- appointment: Arraylist <Appointment>
- id: int
+ name: String
+ addr: Arraylist<Address>
+ email: String
+ setGender(char): null
+ getGender(): char
+ setAge(float): null
+ getAge(): float
+ setAppointment(Arraylist<Appointment>): null
+ getAppointment(): Arraylist<Appointment>
+ setId(int): null
+ getId(): int
+ setName(String): null
Query
- qid: int
- pid: int
+ title: String
+ description: String
+ email: String
+ setqid(id): null
+ getqid(): int
+ setPid(int): null
+ getPid(): int
+ setTitle(String): null
+ getTitle(): String
+ setDescription(String): null
+ getDescription(): String
+ setEmail(String): null
+ getEmail(): String
+ generateQueryForm(): Arraylist<Query>
+ load() : Boolean
+ formType(int): Stri
+ MedicalReportFor
+ LabReportFormat(
+ QueryFormat() : Q


Patient
gender: char
age: float
- appointment: Arraylist <Appointment>
- id: int
+ name: String
+ addr: Arraylist<Address>
+ email: String
+ setAppointment(Arraylist<Appointment>): void
+ getAppointment(): Arraylist<Appointment>
+ addAppointment(doc_id:int, time: time): void
+ login(): void
+ login(username, password, User.Patient): void
Doctor
- specialisation: String
- experience: float
- affiliation_no: int
- appointment: Arraylist <Appointment>
schedule: String
- id: int
name: String
addr: Arraylist<Address>
+ setSpecialisation(String): void
+ getSpecialisation(): String
+ setAppointment(Arraylist<Appointment>): void
+ getAppointment(): Arraylist<Appointment>
+ setSchedule(String): void
+ getSchedule(Date: datetime): Schedule: String
+ getDoctordetails(listofdoctordetails:Arraylist): Arraylist
+ login(): void
+ login(username, password, User.Doctor): void
QueryFacade
+ updateQuery(qid: int, query: String): Boolean
Query
- qid: int
- pid: int
- query: String
- title: String
+generateQueryForm() : Arraylist <Query>
+load():Boolean
+notifyDoctors(): Boolean

Proxy Design Pattern

To authenticate the doctors' and patients' access to the medical record. Only particular patients and doctors can access medical records of patients. To protect this access, we implemented "Protection Proxy". The presence of Proxy ensures that only authorized doctors can access the medical record.


+ setAffliation_no(int): null
+ getAffiliation_no(): int
+ setAppointment(Arraylist<Appointment>): null
+ getAppointment(): Arraylist<Appointment>
+ setSchedule(String): null
+ getSchedule(Date: datetime): Schedule: String
+ setId(int): null
+ getId(): int
+ setName(String): null
+ getName(): String
+ setAddr(Arraylist<Address>): null
+ getAddr(): Arraylist<Address>
+ getUserObject(id:int): pharmacist: Pharmacist
+ getPatientRecord(patientid: int): Arraylist <MedicalRecord>
+ viewPatientLabTestHistory(patientid:int): Arraylist <LabReport>
+ uploadLabReport(labReport: LabReport, patientid: int,doctorid: int): null
Doctor
- specialisation: String
- experience: float
- affiliation_no: int
- appointment: Arraylist <Appointment>
schedule: String
- id: int
name: String
addr: Arraylist<Address>
+ setSpecialisation(String): null
+ getSpecialisation(): String
+ setExperience(float): null
+ getExperience(): float
+ setAffliation_no(int): null
+ getAffiliation_no(): int
+ setAppointment(Arraylist<Appointment>): null
+ getAppointment(): Arraylist<Appointment>
+ setSchedule(String): null
+ getSchedule(Date: datetime): Schedule: String
+ setId(int): null
+ getId(): int
+ setName(String): null
+ getName(): String
+ setAddr(Arraylist<Address>): null
+ getAddr(): Arraylist<Address>
+ getUserNames(specialisation: String): null
+ getUserObject(id:int): doctor: Doctor
+ getDoctordetails(listofdoctordetails:Arraylist): Arraylist
+ notifydoctor(appoint: Appointment)
+ viewMedicine(medicinename: String, location: Address)
Appointment
- id: int
- doctid: int
- patientid: int
- date: Datetime
- time: time
+ setId(int): null
+ getId(): int
+ setDoctid(int): null
+ getDoctid(): int
+ setPatientid(int): null
+ getPatientid(): int
+ setDate(date: Datetime): null
+ getDate(): Datetime
+ setTime(time): null
+ getTime(): time
+ createAppointment(): null
+ generateAppointment(): Arraylist<Appointment>
+ load(): Boolean
DbCall
+ executeQuery(query: String): Boolean
Patient
gender: char
age: float
- appointment: Arraylist <Appointment>
- id: int
+ name: String
+ addr: Arraylist<Address>
+ email: String
+ setGender(char): null
+ getGender(): char
+ setAge(float): null
+ getAge(): float
+ setAppointment(Arraylist<Appointment>): null
+ getAppointment(): Arraylist<Appointment>
+ setId(int): null
+ getId(): int
+ setName(String): null
+ getName(): String
+ setAddr(Arraylist<Address>): null
+ getAddr(): Arraylist<Address>
+ setEmail(String): null
+ getEmail(): String
+ addAppointment(doc_id:int, time: time): null
+ getUserObject(id:int): patient: Patient
+ viewMedicine(medicinename: String, location: Address)
MedicalRecord
- id: int
- patientid: int
- doctid: int
- date: Datetime
- comment: String
- precription: String
+ setId(int): null
+ getId(): int
+ setPatientid(int): null
+ getPatientid(): int
+ setDoctid(int) : null
+ getDoctid(int): int
+ setDate(date:Datetime): null
+ getDate(): Datetime
+setComment(String): null
+getComment(): String
+setPrescription(String): null
+getPrescription(): String
+createMedicalReport(): Arraylist<MedicalReport>
Address
street: String
* 1 0..*


Patient
gender: char
age: float
- appointment: Arraylist <Appointment>
- id: int
+ name: String
+ addr: Arraylist<Address>
+ email: String
+ setAppointment(Arraylist<Appointment>): void
+ getAppointment(): Arraylist<Appointment>
+ addAppointment(doc_id:int, time: time): void
+ login(): void
+ login(username, password, User.Patient): void
Doctor
- specialisation: String
- experience: float
- affiliation_no: int
- appointment: Arraylist <Appointment>
schedule: String
- id: int
name: String
addr: Arraylist<Address>
+ setSpecialisation(String): void
+ getSpecialisation(): String
+ setAppointment(Arraylist<Appointment>): void
+ getAppointment(): Arraylist<Appointment>
+ setSchedule(String): void
+ getSchedule(Date: datetime): Schedule: String
+ getDoctordetails(listofdoctordetails:Arraylist): Arraylist
+ login(): void
+ login(username, password, User.Doctor): void
*
*
1
*
0..*
0..*
<<interface>>
MedicalRecordProxy
1
1
Appointment
- id: int
- doctid: int
- patientid: int
- date: Datetime
- time: time
+ setId(int): void
+ getId(): int
+ setDoctid(int): void
+ getDoctid(): int
+ setPatientid(int): void
+ getPatientid(): int
+ setDate(date: Datetime): void
+ getDate(): Datetime
+ setTime(time): void
+ getTime(): time
+ createAppointment(): void
+ generateAppointment(): Arraylist<Appointment>
+ load(): Boolean
MedicalRecord
- id: int
- patientid: int
- doctid: int
- date: Datetime
- comment: String
- precription: String
+ setId(int): void
+ getId(): int
+ setPatientid(int): void
+ getPatientid(): int
+ setDoctid(int) : void
+ getDoctid(int): int
+ setDate(date:Datetime): void
+ getDate(): Datetime
+setComment(String): void
+getComment(): String
+setPrescription(String): void
+getPrescription(): String
+createMedicalReport(): Arraylist<MedicalRecord>
Proxy
- authId: int
- authenticate(): Boolean
- viewRecord(): void

INTERESTING FEATURES

1. People have the flexibility to ask general queries from doctors.
2. Doctors can reply to those queries as soon as they receive it in the platform.

DEMO

Use Cases demo:
https://www.youtube.com/watch?v=EgTiavDfVZk

1. Book appointment
2. Add prescription
3. Consult online

THANK YOU