**DBSCHEMA:**

**Database Schema Design**

**#### Tables**

1.Appointments

2.Doctors

3.Notifications

4.UsersModel

####**Fields Used In Tables**

1. **Appointments**

userId

doctorId

date

time

status

timeStamps

1. **Doctors**

userId

specialization

experience

fees

isDoctor

timeStamps

1. **Notifications**

userId

isRead

content

timeStamps

1. **UserModel**

firstname

lastname

email

password

isAdmin

isDoctor

age

gender

mobile

address

status

pic

**DBSCHEMA Implementation:**

**For Appointments:**

const mongoose = require("mongoose");

const schema = mongoose.Schema(

{

userId: {

type: mongoose.SchemaTypes.ObjectId,

ref: "User",

required: true,

},

doctorId: {

type: mongoose.SchemaTypes.ObjectId,

ref: "User",

required: true,

},

date: {

type: String,

required: true,

},

time: {

type: String,

required: true,

},

status: {

type: String,

default: "Pending",

},

},

{

timestamps: true,

}

);

const Appointment = mongoose.model("Appointment", schema);

module.exports = Appointment;

**For Doctors:**

const mongoose = require("mongoose");

const schema = mongoose.Schema(

{

userId: {

type: mongoose.SchemaTypes.ObjectId,

ref: "User",

required: true,

},

specialization: {

type: String,

required: true,

},

experience: {

type: Number,

required: true,

},

fees: {

type: Number,

required: true,

},

isDoctor: {

type: Boolean,

default: false,

},

},

{

timestamps: true,

}

);

const Doctor = mongoose.model("Doctor", schema);

module.exports = Doctor;

**For Notifications:**

const mongoose = require("mongoose");

const schema = mongoose.Schema(

{

userId: {

type: mongoose.SchemaTypes.ObjectId,

ref: "User",

required: true,

},

isRead: {

type: Boolean,

default: false,

},

content: {

type: String,

default: "",

},

},

{

timestamps: true,

}

);

const Notification = mongoose.model("Notification", schema);

module.exports = Notification;

**For UserModel:**

const mongoose = require("mongoose");

const schema = mongoose.Schema(

{

firstname: {

type: String,

required: true,

minLength: 3,

},

lastname: {

type: String,

required: true,

minLength: 3,

},

email: {

type: String,

required: true,

unique: true,

},

password: {

type: String,

required: true,

minLength: 5,

},

isAdmin: {

type: Boolean,

default: false,

},

isDoctor: {

type: Boolean,

default: false,

},

age: {

type: Number,

default: "",

},

gender: {

type: String,

default: "neither",

},

mobile: {

type: Number,

default: "",

},

address: {

type: String,

default: "",

},

status: {

type: String,

default: "pending",

},

pic: {

type: String,

default:

"https://icon-library.com/images/anonymous-avatar-icon/anonymous-avatar-icon-25.jpg",

},

},

{

timestamps: true,

}

);

const User = mongoose.model("User", schema);

module.exports = User;

**Troubleshooting and Optimization:**

1. **Network Troubleshooting**: Know common network issues and troubleshooting methodologies (e.g., ping, traceroute).
2. **Performance Optimization**: Understand techniques for optimizing network performance (e.g., congestion control, QoS).