🏥 Healthcare Appointment Booking System

A **mini project** that demonstrates a complete **Frontend–Backend architecture** using:

- **Django + Django REST Framework (Backend)**
- **PyQt5 (Desktop Frontend)**
- **SQLite (Database)**

The system allows patients to:

- Login
- View available doctors
- Book appointments
- View booked appointments

🧩 Project Architecture

```
Frontend (PyQt5)
   │
   │ HTTP Requests (REST API)
   ▼
Backend (Django + DRF)
   │
   ▼
Database (SQLite)
```

🛠️ Technologies Used

Backend

- Python 3
- Django
- Django REST Framework
- SQLite (default Django DB)

Frontend

- PyQt5
- Requests (HTTP client)

📂 Project Structure

```
mini_project/
├── backend/
│   ├── healthcare_backend/
│   ├── hospital_app/
│   ├── manage.py
│   └── db.sqlite3
│
├── frontend/
│   ├── main.py
│   └── ui/
│       └── main.ui
│
├── requirements.txt
└── README.md
```

⚙️ Installation & Setup

1️⃣ Clone the Repository

```
git clone <your-repo-url>
cd mini_project
```

2️⃣ Create Virtual Environment

```
python -m venv venv
source venv/bin/activate   # Linux / macOS
# venv\\Scripts\\activate  # Windows
```

3️⃣ Install Dependencie

```
pip install -r requirements.txt
```

🚀 How to Run

▶ Backend (Django)

```
cd backend
python manage.py migrate
python manage.py runserver
```

Backend will run at:

http://127.0.0.1:8000/

▶ Frontend (PyQt)

Open **a new terminal** (keep backend running):

```
cd frontend
python main.py
```

The PyQt desktop application will open.

🔌 API Endpoints (Backend)

Method	Endpoint	Description
GET	`/api/doctors/`	List all doctors
POST	`/api/login/`	Login user
POST	`/api/book-appointment/`	Book appointment
GET	`/api/appointments/?patient=<username>`	View appointments

🖥️ Frontend Features (PyQt)

- Login screen
- Patient dashboard
- Doctor selection dropdown
- Date & time picker
- Appointment booking
- Appointment list (QTableWidget)
- Logout & exit

🧠 Key Concepts Demonstrated

- Django REST API design
- Separation of frontend and backend
- HTTP communication using JSON
- Desktop GUI using PyQt
- Real-world CRUD operations

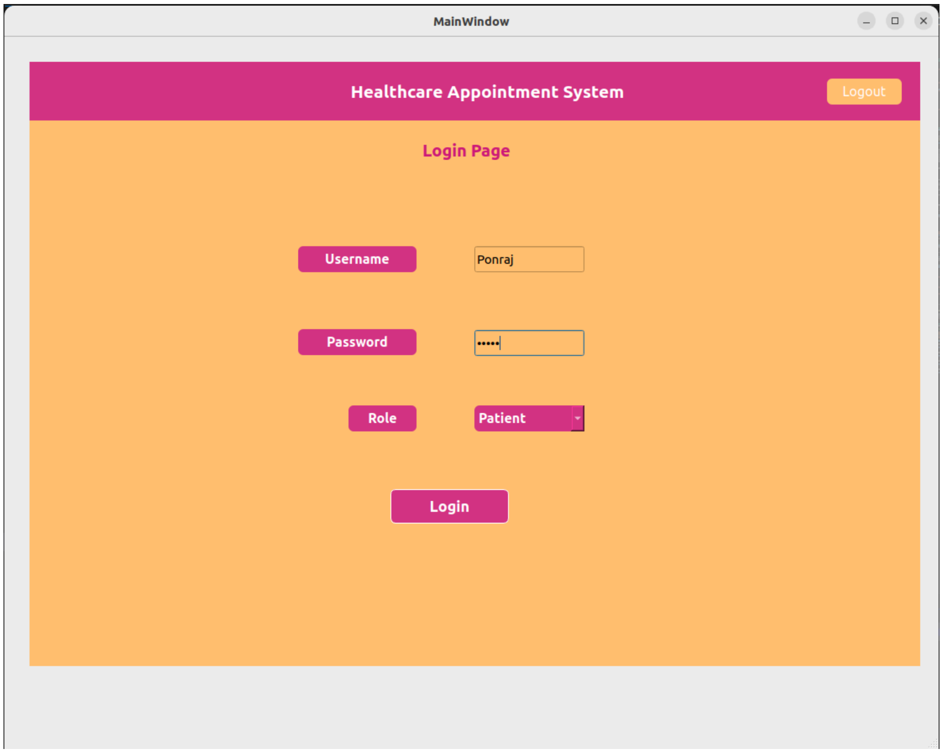
📌 Why Django REST Framework?

- Converts Django models into JSON
- Makes backend reusable for:
    - Web apps
    - Mobile apps
    - Desktop apps (PyQt)
- Clean separation of concerns

🧪 Sample Login Data

Username: Ponraj
Role: Patient

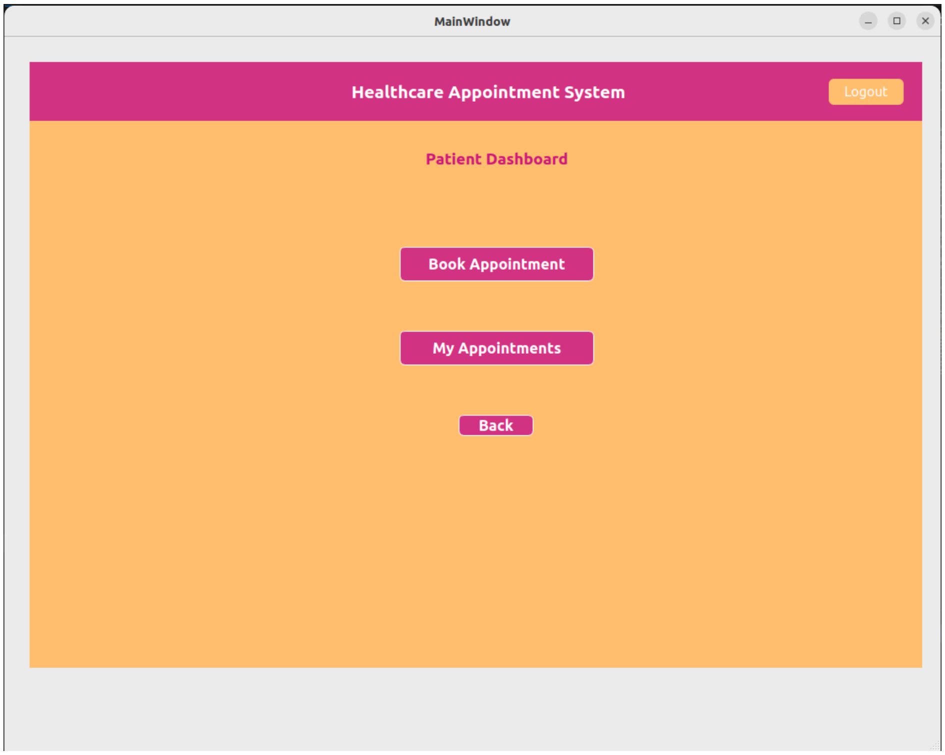
📷 Screenshots

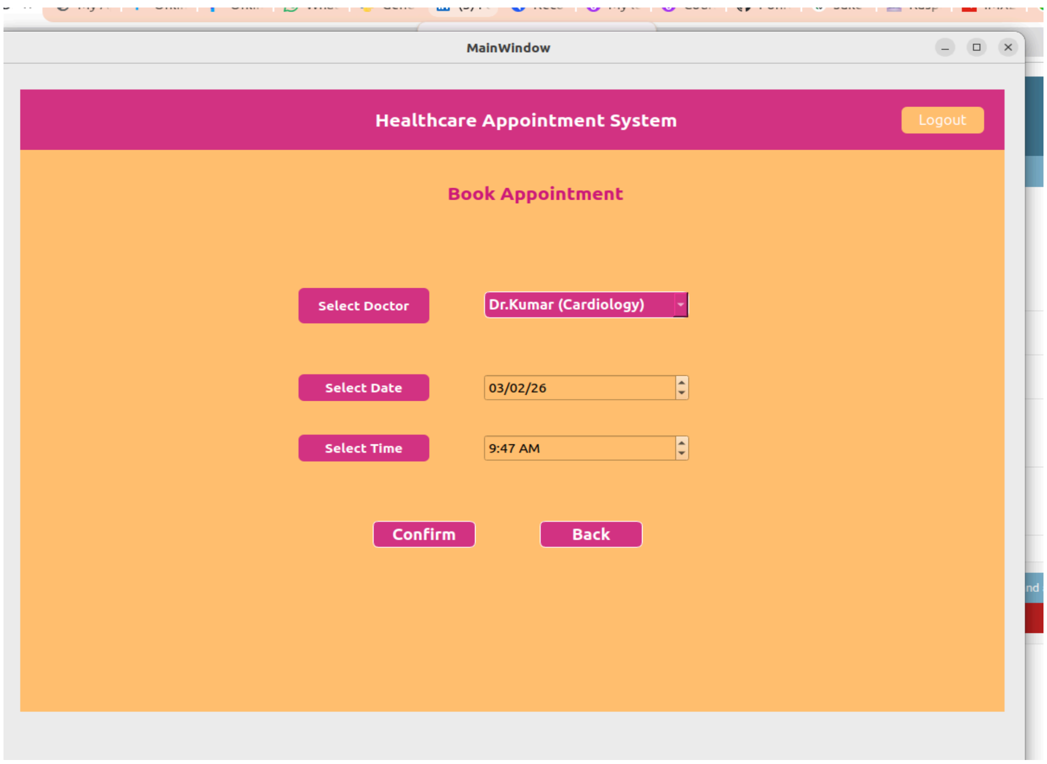


Username: Ponraj

Password: •••••

Role: Patient

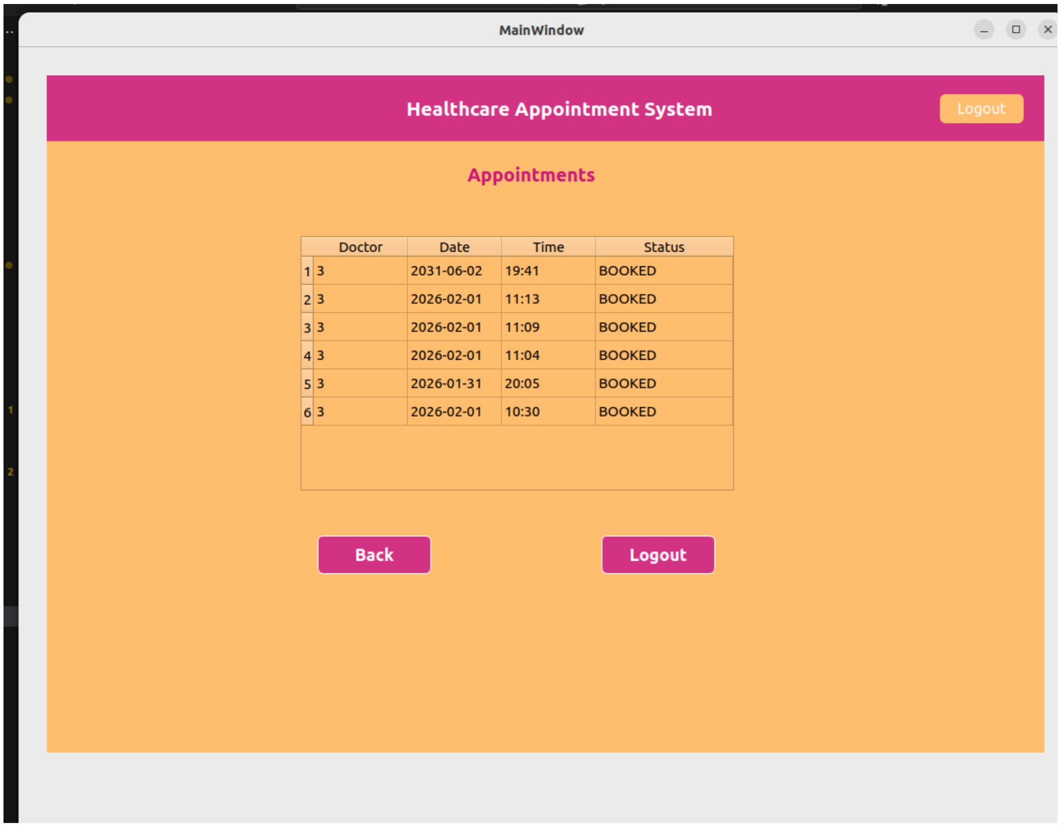




Select Doctor: Dr.Kumar (Cardiology)

Select Date: 03/02/26

Select Time: 9:47 AM



	Doctor	Date	Time	Status
1	3	2031-06-02	19:41	BOOKED
2	3	2026-02-01	11:13	BOOKED
3	3	2026-02-01	11:09	BOOKED
4	3	2026-02-01	11:04	BOOKED
5	3	2026-01-31	20:05	BOOKED
6	3	2026-02-01	10:30	BOOKED

📈 Future Improvements

- Password authentication
- Role-based access (Doctor / Admin)
- Appointment cancellation
- Token-based authentication (JWT)
- Deployment (Docker / Cloud)

👨‍💻 Author

Ponraj P
Python Developer | Django | PyQt | MySQL | OpenCV |

⭐ GitHub

If you like this project, give it a ⭐ and feel free to fork!