

# FRONT- DESK SYSTEM AT A CLINIC



## SUMMARY OF THE PROJECT

The Front Desk System is a web-based application designed to streamline patient queue management and appointment scheduling at a clinic. It enables front desk staff to efficiently handle walk-in patients, manage appointments, and track patient progress. Key functionalities include:

- **Queue Management:** Assigning queue numbers to patients, updating their statuses (e.g., waiting, with doctor, completed), and managing the queue efficiently.
- **Appointment Management:** Booking, rescheduling, and canceling appointments, viewing available doctors and their time slots, and updating appointment statuses.
- **Doctor Profile Management:** Adding, editing, and deleting doctor profiles, including details like specialization, location, gender, and availability.
- **Authentication:** Secure login functionality using JWT for system access.

# TECH STACK

NestJS: Nodejs framework for building scalable server-side applications.

TypeORM: ORM for managing MySQL database interactions.

Next.js: React framework for server-side rendering and static web pages.

Tailwind CSS: For responsive and efficient UI styling.

JWT Authentication: For secure login and authorization.

MySQL: Database to store user, doctor, and appointment data

#### MODEL:-

- Represents the database structure and handles data logic.
- Entities like User, Doctor, Appointment, and Queue are managed using TypeORM.

## MVC ARCHITECTURE

#### CONTROLLER:-

- Handles user requests and orchestrates actions between the Model and the View.
- Examples:
  - QueueController: Manages requests for queue operations.
  - AppointmentController: Handles appointment-related requests (e.g., booking, rescheduling).
  - AuthController: Manages authentication tasks.

#### VIEW:-

The user interface, built using Next.js and styled with Tailwind CSS, includes:

- Queue Management Page: Displays the patient queue and their statuses.
- Appointment Management View: Lists available doctors and facilitates appointment actions.
- Login Page: Secure authentication interface.

# SAMPLE SCREENSHOTS OF OUR RESULTS

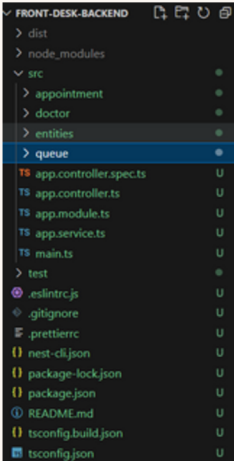
```
C:\Users\xolom\front-desk-backend>npm install
up to date, audited 752 packages in 1s
123 packages are looking for funding
  run 'npm fund' for details
found 0 vulnerabilities

C:\Users\xolom\front-desk-backend>npm run start
> front-desk-backend@0.0.1 start
> nest start

[Nest] 29276 - 01/02/2025, 4:35:59 PM LOG [NestFactory] Starting Nest application...
[Nest] 29276 - 01/02/2025, 4:36:00 PM LOG [InstanceLoader] TypeOrmModule dependencies initialized +61ms
[Nest] 29276 - 01/02/2025, 4:36:00 PM LOG [InstanceLoader] TypeOrmCoreModule dependencies initialized +489ms
[Nest] 29276 - 01/02/2025, 4:36:00 PM LOG [InstanceLoader] TypeOrmModule dependencies initialized +1ms
[Nest] 29276 - 01/02/2025, 4:36:00 PM LOG [InstanceLoader] AppModule dependencies initialized +1ms
[Nest] 29276 - 01/02/2025, 4:36:00 PM LOG [RoutesResolver] AppointmentController (/appointments): +6ms
[Nest] 29276 - 01/02/2025, 4:36:00 PM LOG [RouterExplorer] Mapped {/appointments, GET} route +3ms
[Nest] 29276 - 01/02/2025, 4:36:00 PM LOG [RouterExplorer] Mapped {/appointments, POST} route +1ms
[Nest] 29276 - 01/02/2025, 4:36:00 PM LOG [RouterExplorer] Mapped {/appointments/:id, PUT} route +0ms
[Nest] 29276 - 01/02/2025, 4:36:00 PM LOG [RouterExplorer] Mapped {/appointments/:id, DELETE} route +1ms
[Nest] 29276 - 01/02/2025, 4:36:00 PM LOG [RoutesResolver] DoctorController (/doctors): +0ms
[Nest] 29276 - 01/02/2025, 4:36:00 PM LOG [RouterExplorer] Mapped {/doctors, GET} route +1ms
[Nest] 29276 - 01/02/2025, 4:36:00 PM LOG [RouterExplorer] Mapped {/doctors, POST} route +0ms
[Nest] 29276 - 01/02/2025, 4:36:00 PM LOG [RouterExplorer] Mapped {/doctors/:id, PUT} route +1ms
[Nest] 29276 - 01/02/2025, 4:36:00 PM LOG [RouterExplorer] Mapped {/doctors/:id, DELETE} route +0ms
[Nest] 29276 - 01/02/2025, 4:36:00 PM LOG [RoutesResolver] QueueController (/queue): +0ms
[Nest] 29276 - 01/02/2025, 4:36:00 PM LOG [RouterExplorer] Mapped {/queue, GET} route +1ms
[Nest] 29276 - 01/02/2025, 4:36:00 PM LOG [RouterExplorer] Mapped {/queue, POST} route +1ms
[Nest] 29276 - 01/02/2025, 4:36:00 PM LOG [RouterExplorer] Mapped {/queue/:id, PUT} route +0ms
[Nest] 29276 - 01/02/2025, 4:36:00 PM LOG [NestApplication] Nest application successfully started +2ms
```

Screenshot of backend running successfully

# SAMPLE SCREENSHOTS OF OUR RESULTS



Screenshot of the folders in backend

# SAMPLE SCREENSHOTS OF OUR RESULTS

```
C:\Users\xolom\front-desk-frontend>npm run dev

> front-desk-frontend@0.1.0 dev
> next dev --turbo

⚠ Port 3000 is in use, trying 3001 instead.
  ▲ Next.js 15.1.3 (Turbopack)
  - Local:      http://localhost:3001
  - Network:    http://192.168.29.68:3001

✓ Starting...
✓ Ready in 1003ms
```

Screenshot of frontend running successfully

# SAMPLE SCREENSHOTS OF OUR RESULTS

Login

Email

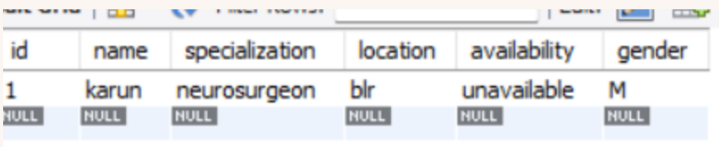
Password

Login

Login page (test image)



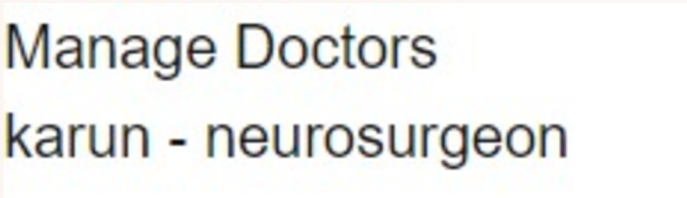
# SAMPLE SCREENSHOTS OF OUR RESULTS



A screenshot of a web browser displaying a table with doctor information. The table has six columns: id, name, specialization, location, availability, and gender. The first row contains the data for a doctor named karun, who is a neurosurgeon located inblr, with an availability of unavailable and gender M. Below the first row, there are several rows with NULL values in the id, name, specialization, location, availability, and gender columns.

id	name	specialization	location	availability	gender
1	karun	neurosurgeon	blr	unavailable	M
NULL	NULL	NULL	NULL	NULL	NULL

Backend (SQL)



A screenshot of a web browser displaying the text "Manage Doctors" and "karun - neurosurgeon".

Frontend  
display

Thank  
you very  
much!

