Chosen Architectural Pattern: Model-View-Controller (MVC) Architecture

# Why MVC Architecture?

## 1. Clear Separation of Concerns for Healthcare Domains

MVC architecture divides your application into three distinct components:  
  
- Model: Manages the core healthcare data and logic, such as:  
 \* Patient Records  
 \* Appointments  
 \* Laboratory Results  
 \* Billing & Insurance  
 \* User Permissions and Access Control  
  
- View: Represents various user interfaces for:  
 \* Doctors and nurses (EHR dashboard, lab results)  
 \* Patients (appointments, notifications, reports)  
 \* Admin staff (billing, insurance, scheduling)  
  
- Controller: Manages user inputs and connects models with the correct view.

## 2. Multiple Tailored Views for Different Healthcare Roles

Healthcare systems often involve multiple roles (doctors, patients, lab techs, administrators). MVC enables customized UIs without affecting the underlying data logic.

## 3. Improved Maintainability and Testability

- Easier Testing: Each component (model, view, controller) can be tested independently.  
- Better Maintainability: UI changes don’t affect logic; logic changes don’t affect data structure.

## 4. Scalable and Extensible Design

While not as inherently scalable as microservices, MVC scales well within web application environments using:  
- Load balancing  
- Service-layer abstraction  
- REST APIs for mobile and 3rd-party system integration

## 5. Enhanced Security and Compliance

- Centralized control in controllers allows secure user input handling.  
- Views can be permission-filtered to match user roles (e.g., using HIPAA-compliant filters).  
- Logging and access tracking can be implemented at controller and model layers.

# Conclusion

The MVC Architecture is well-suited for a healthcare system where:  
- Multiple distinct user interfaces are needed  
- Rapid UI updates and role-specific views are essential  
- Business logic and data management must be kept consistent and secure  
  
This architecture provides a clean structure, robust user experience, and a solid foundation for future enhancements.