Architecture Model for MediLink System

Chosen Pattern:

Model–View–Controller (MVC) architecture — it matches perfectly with the diagram and your web-based system structure.

1. Layered Overview

Layer	**Main Components**	**Responsibilities**	**Example from Diagram / SRS**
Presentation Layer (View)	HTML, CSS, JavaScript, Django Templates	User Interface – patients, doctors, admins, and staff interact here	Login Page, Appointment Form, Dashboard, Reports Page
Application Layer (Controller)	Django Views / Controllers	Handles system logic, processes requests, validates data, coordinates between model and view	Validate Input Data, Check Slot Availability, Send Appointment Confirmation
Data Layer (Model)	Django Models / MySQL Database	Stores and manages persistent data	Tables: Patient, Doctor, Appointment, MedicalRecord, Billing, LabReport

2. Component Breakdown (Detailed Model)

A. Model Layer

Handles **data entities** and database operations.

Model	Attributes (Example)	Functions / Actions (from Diagram)
Patient	ID, Name, Age, Gender, Email, Password	Register(), Login(), BookAppointment(), ViewReports()
Doctor	ID, Name, Department, Email, Password	ViewAppointments(), UpdateMedicalRecord(), ConsultPatient()
Admin	ID, Username, Password	ManageDoctorProfiles(), ManagePatientRecords(), GenerateReports()
Appointment	ID, PatientID, DoctorID, Date, Time, Status	CheckSlotAvailability(), CancelAppointment(), SendConfirmation()
MedicalRecord	RecordID, PatientID, DoctorID, Diagnosis, Prescription	AddRecord(), UpdateRecord(), RemoveRecord()
Billing	BillID, PatientID, Amount, Status	ManageBilling(), AssistInBilling()
LabReport	ReportID, PatientID, TestType, Result	ProvideLabReports(), SaveReport()

B. Controller Layer

Implements all **logical actions** shown in your use case diagram.

Controller Module	Handled Use Cases	Functions
AuthenticationController	Log In, Validate Input, Show Invalid Credentials	verifyCredentials(), forgotPassword()

Controller Module	Handled Use Cases	Functions
AppointmentController	Book Appointment, Cancel Appointment, Check Slot	bookAppointment(), suggestAlternativeSlot()
MedicalRecordController	Add / Update / Remove Medical Record	validateRequiredFields(), saveRecord()
ReportController	Generate Reports, View Reports	generateReport(), saveReport()
BillingController	Manage Billing, Assist in Billing	calculateBill(), processPayment()

C. View Layer

Represents all **user-facing pages and forms**.

Interface / Page	User	Displayed Features
Login Page	Admin, Doctor, Patient	Login form, error messages
Dashboard Page	Admin	Manage Doctors, Patients, Billing
Appointment Page	Patient	Book, Cancel, View Appointments
Doctor Dashboard	Doctor	View Appointments, Update Records
Support Staff Page	Support Staff	Manage Lab Reports, Assist Billing
Reports Page	Admin/Patient	Generate or View Reports

3. System Flow Example

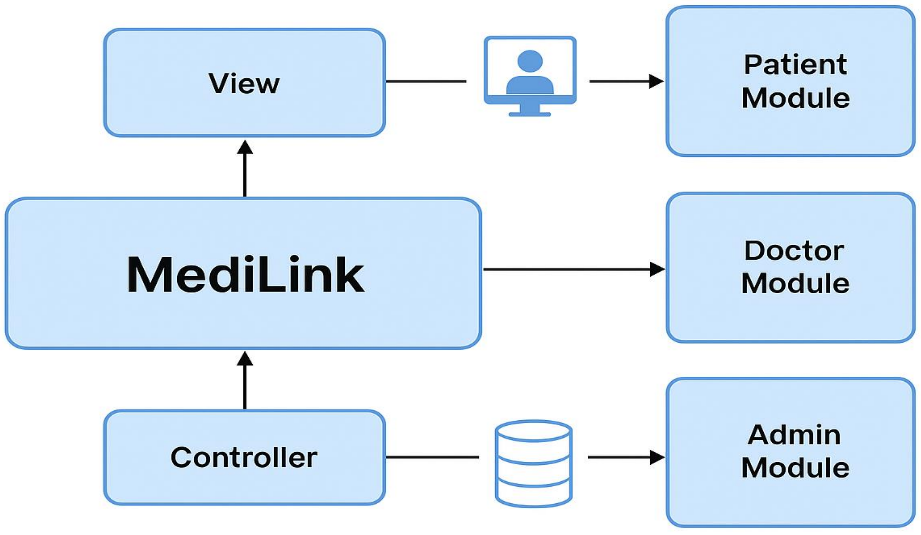
Scenario: Patient Books an Appointment

1. **Patient (View)** → Fills form on “Book Appointment” page.

2. **AppointmentController (Controller)** → Validates input and checks slot availability.
3. **Appointment Model (Model)** → Stores the appointment data in MySQL.
4. **Controller** → Sends confirmation to the patient view.
5. **View** → Displays "Appointment Booked Successfully" message.

4. Interactions Between Actors (as per Use Case Diagram)

Actor	Interacts With	Through Controller
Patient	Appointment, MedicalRecord, Report	AppointmentController, ReportController
Doctor	MedicalRecord, Appointment	MedicalRecordController
Admin	Doctor, Patient, Billing	AdminController, BillingController
Support Staff	LabReport, Billing	SupportController



Model–View-Controller (MVC)