SYSTEM ARCHITECTURE

Package Diagram

1. **Views Package**
    - Contains UI components: Admin, Trainer, and Login
    - This package handles the user interface that clients interact with
2. **Application Package**
    - Contains Main and Controller components
    - This is likely the core application logic that coordinates between views and data
3. **Models Package**
    - Contains data models: Customer, Trainer, Receptionist, Shopkeeper, Complaint, Lockers, and Machines
    - These represent the business entities in your fitness tracking system
4. **DB Package**
    - Contains DBHandler component
    - Handles database operations and persistence

The relationships between these packages are:

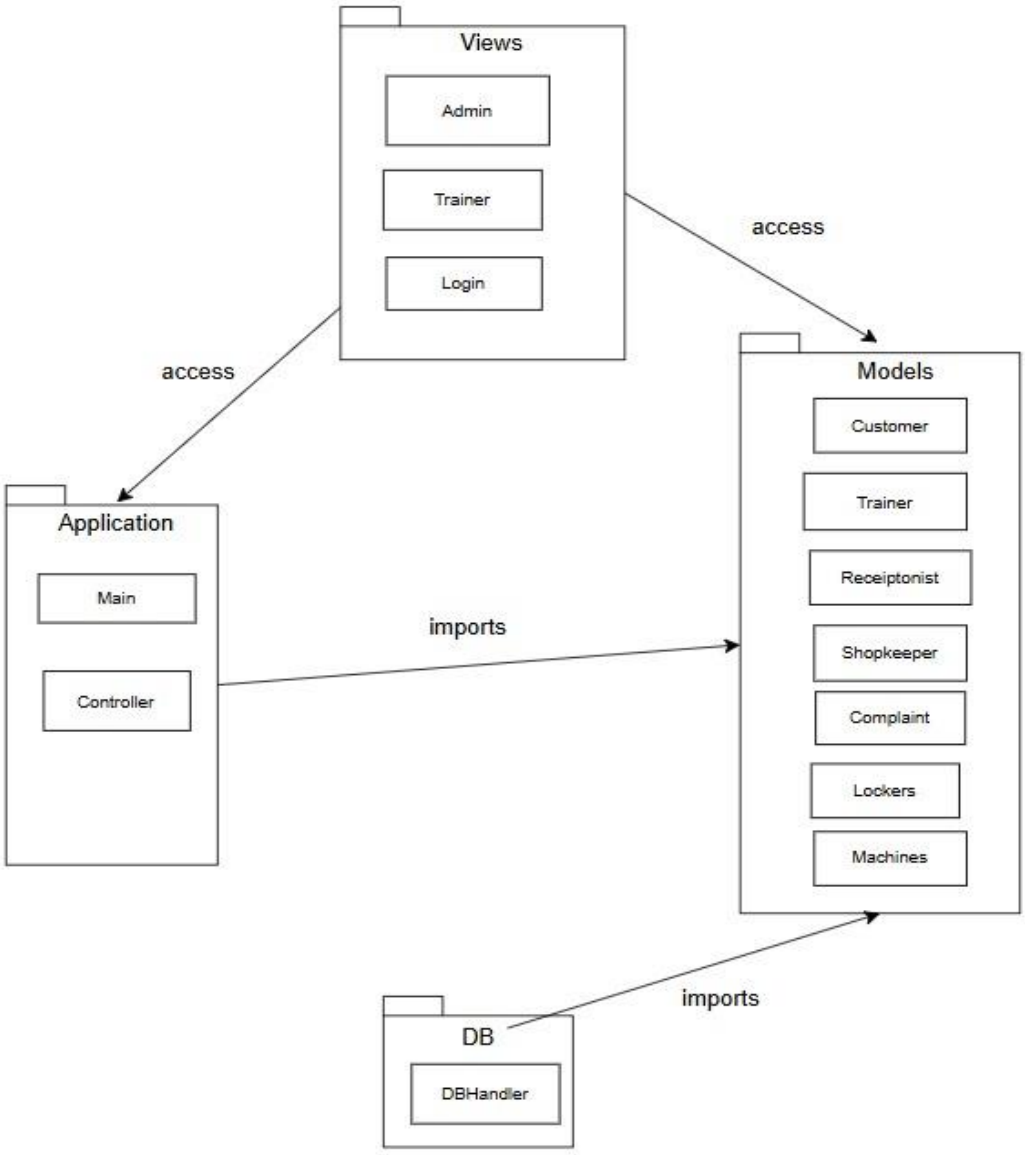
- Views package has "access" to the Application package
- Views package has "access" to the Models package
- Application package "imports" the Models package
- DB package "imports" the Models package

This structure aligns well with your Prime Fitness application requirements. The separation of concerns is appropriate for a web-based fitness tracking system:

- The Views package supports your user interfaces for different user types (admin, trainer, and general login)
- The Models package contains the entities needed for your workout management, nutrition tracking, and user accounts

- The Application package provides the controller logic to implement your product functions
- The DB package handles data persistence for tracking progress, workout history, and user information

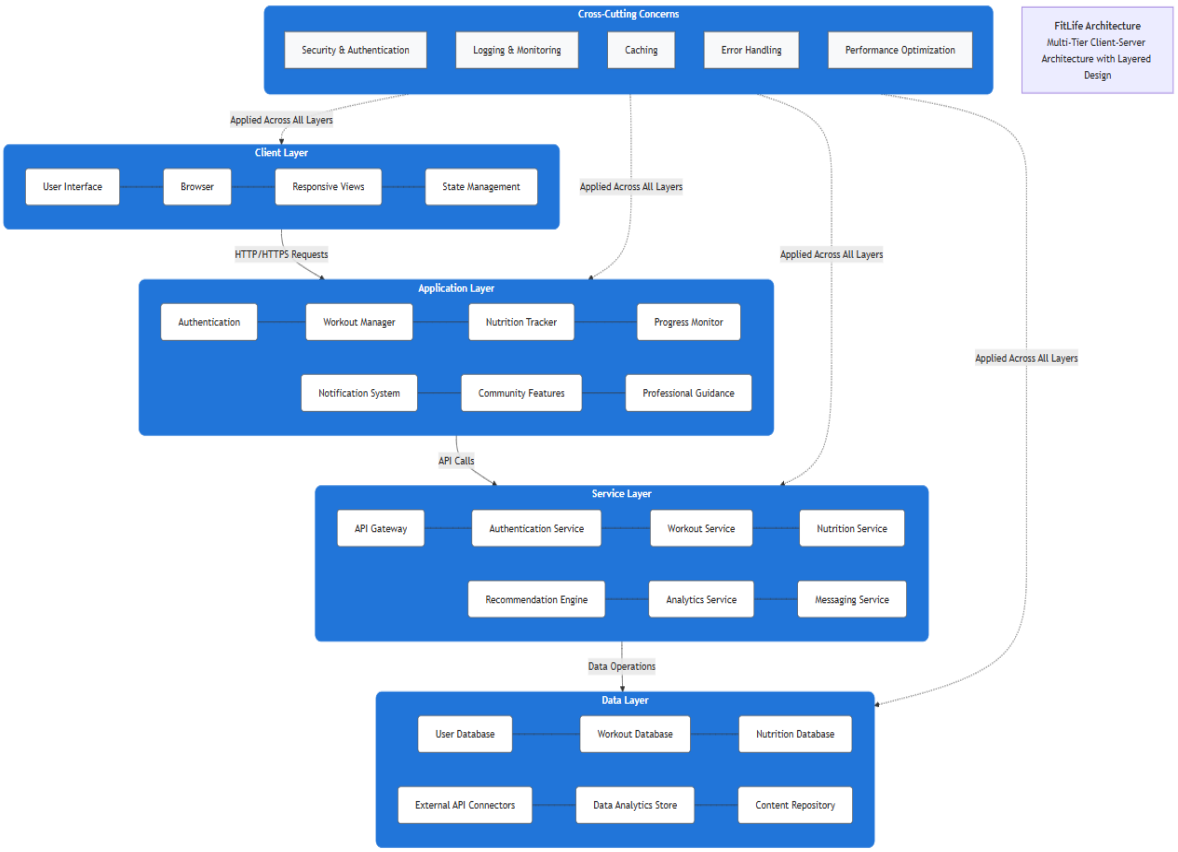
This architecture should facilitate the implementation of key features like user account management, workout tracking, nutrition logging, progress monitoring, professional guidance, community engagement, and notifications as described in your requirements.



Architecture Styles

1. **Client Layer**: Handles the user-facing interface, including browser-based UI components, responsive views, and state management.
2. **Application Layer**: Contains the core business logic components like authentication, workout management, nutrition tracking, progress monitoring, notification systems, professional guidance, and community features.
3. **Service Layer**: Provides specialized services that support the application components, including API gateway, authentication, workout services, nutrition services, recommendation engine, analytics, and messaging.
4. **Data Layer**: Manages data storage and access through various databases (user, workout, nutrition) along with external API connectors, analytics store, and content repository.

This architecture separates concerns while allowing components to communicate across layers, enabling scalability, maintainability, and flexibility. The design allows FitLife to handle diverse fitness tracking functions while maintaining a clean separation between presentation, business logic, services, and data management.


FitLife Architecture
Multi-Tier Client-Server Architecture with Layered Design
Cross-Cutting Concerns
Security & Authentication
Logging & Monitoring
Caching
Error Handling
Performance Optimization
Applied Across All Layers
Client Layer
User Interface
Browser
Responsive Views
State Management
Applied Across All Layers
HTTP/HTTPS Requests
Applied Across All Layers
Application Layer
Authentication
Workout Manager
Nutrition Tracker
Progress Monitor
Notification System
Community Features
Professional Guidance
Applied Across All Layers
API Calls
Service Layer
API Gateway
Authentication Service
Workout Service
Nutrition Service
Recommendation Engine
Analytics Service
Messaging Service
Data Operations
Data Layer
User Database
Workout Database
Nutrition Database
External API Connectors
Data Analytics Store
Content Repository