Mô hình Facade là một mẫu thiết kế phần mềm thuộc nhóm cấu trúc (Structural Pattern) trong đó một đối tượng được tạo ra để cung cấp một giao diện đơn giản và thân thiện cho việc tương tác với một tập hợp các đối tượng phức tạp hơn. Facade Pattern cho phép người lập trình giảm độ phức tạp của hệ thống bằng cách che giấu các chi tiết phức tạp và cung cấp một giao diện đơn giản hơn để sử dụng.

Một số ưu điểm của Facade Pattern là:

* Giảm độ phức tạp của hệ thống: Facade Pattern giúp giảm độ phức tạp của hệ thống bằng cách che giấu các chi tiết phức tạp và cung cấp một giao diện đơn giản hơn để sử dụng.
* Tăng tính linh hoạt của hệ thống: Với Facade Pattern, người lập trình có thể dễ dàng thay đổi các phần của hệ thống mà không cần thay đổi các phần khác của hệ thống.
* Dễ dàng sử dụng: Facade Pattern cung cấp một giao diện đơn giản để sử dụng và hiểu, giúp người lập trình dễ dàng sử dụng hệ thống mà không cần hiểu các chi tiết phức tạp của hệ thống.

Ví dụ về sử dụng Facade Pattern trong mô hình bệnh viện có thể là:

Giả sử trong bệnh viện có rất nhiều phòng khám, mỗi phòng khám có một loạt các đối tượng như bác sĩ, y tá, trang thiết bị y tế và vật liệu tiêu hao. Nếu mỗi lần bệnh nhân cần đến phòng khám để chữa bệnh, họ phải tương tác với tất cả các đối tượng này, điều này có thể làm cho hệ thống trở nên rất phức tạp.

Thay vào đó, Facade Pattern có thể được sử dụng để tạo ra một đối tượng Facade đại diện cho một phòng khám, và đối tượng này sẽ cung cấp một giao diện đơn giản cho bệnh nhân để tương tác với hệ thống. Bệnh nhân chỉ cần tương tác với đối tượng Facade, và đối tượng này sẽ đảm nhận việc tương tác với các đối tượng phức tạp hơn như bác sĩ, y tá, trang thiết bị y tế và vật liệu tiêu hao.

Ví dụ :

public class Doctor {  
 public void diagnose() {  
 }  
  
 public void treat() {  
 }  
}

public class Equipment {  
 public void prepareForTreatment() {  
  
 }  
  
 public void finishTreatment() {  
  
 }  
}

public class Medicine {  
 public void prepareForTreatment() {  
  
 }  
  
 public void finishTreatment() {  
  
 }  
}

public class Nurse {  
 public void prepareForTreatment() {  
  
 }  
  
 public void finishTreatment() {  
  
 }  
}

public class HospitalSystem {  
 private Doctor doctor;  
 private Nurse nurse;  
 private Equipment equipment;  
 private Medicine medicine;  
  
 public HospitalSystem() {  
 doctor = new Doctor();  
 nurse = new Nurse();  
 equipment = new Equipment();  
 medicine = new Medicine();  
 }  
  
 public void treatPatient() {  
 doctor.diagnose();  
 nurse.prepareForTreatment();  
 equipment.prepareForTreatment();  
 medicine.prepareForTreatment();  
 doctor.treat();  
 nurse.finishTreatment();  
 equipment.finishTreatment();  
 medicine.finishTreatment();  
 }  
}