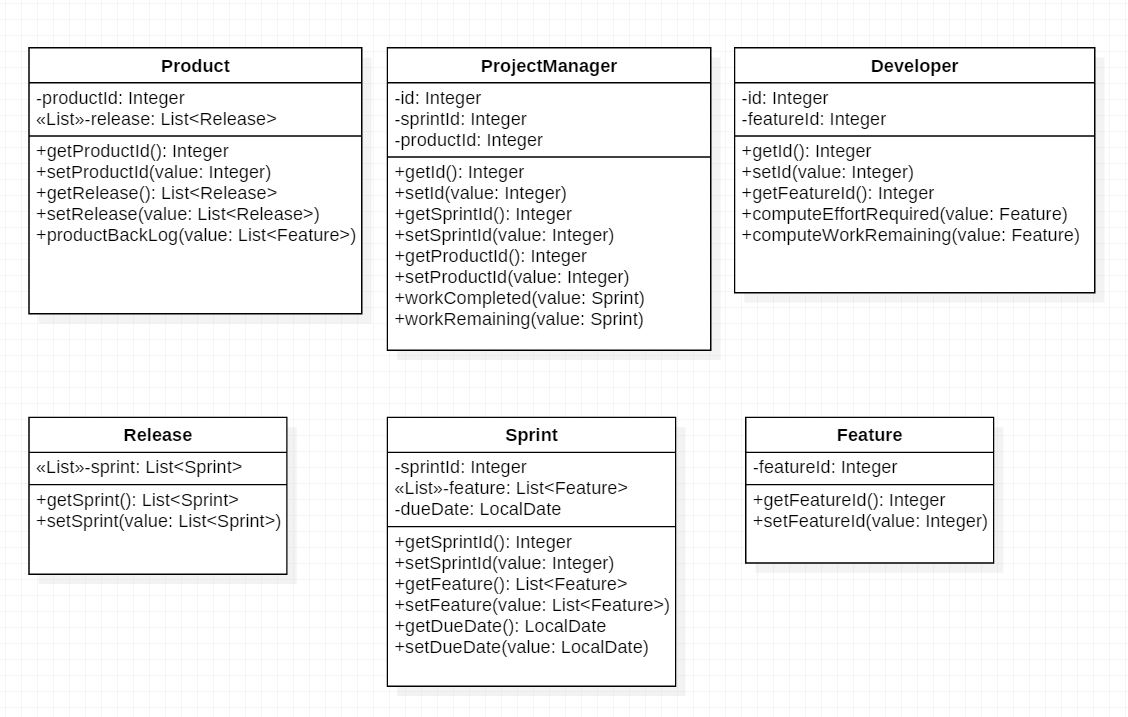
1. UML class diagram of Project Management Tracking System.



1. Java codes
2. Product.java

public class Product {

private Integer productId;

private List<Release> release;

public Integer getProductId() {

return null;

}

public void setProductId(Integer value) {

}

public List<Release> getRelease() {

return null;

}

public void setRelease(List<Release> value) {

}

public void productBackLog(List<Feature> value) {

}

}

1. Release.java

public class Release {

private List<Sprint> sprint;

public List<Sprint> getSprint() {

return null;

}

public void setSprint(List<Sprint> value) {

}

}

1. Sprint.java

public class Sprint {

private Integer sprintId;

private List<Feature> feature;

private LocalDate dueDate;

public Integer getSprintId() {

return null;

}

public void setSprintId(Integer value) {

}

public List<Feature> getFeature() {

return null;

}

public void setFeature(List<Feature> value) {

}

public LocalDate getDueDate() {

return null;

}

public void setDueDate(LocalDate value) {

}

}

1. Feature

public class Feature {

private Integer featureId;

public Integer getFeatureId() {

return null;

}

public void setFeatureId(Integer value) {

}

}

1. Developer.java

public class Developer {

private Integer id;

private Integer featureId;

public Integer getId() {

return null;

}

public void setId(Integer value) {

}

public Integer getFeatureId() {

return null;

}

public void computeEffortRequired(Feature value) {

}

public void computeWorkRemaining(Feature value) {

}

}

1. ProjectManager.java

public class ProjectManager {

private Integer id;

private Integer sprintId;

private Integer productId;

public Integer getId() {

return null;

}

public void setId(Integer value) {

}

public Integer getSprintId() {

return null;

}

public void setSprintId(Integer value) {

}

public Integer getProductId() {

return null;

}

public void setProductId(Integer value) {

}

public void workCompleted(Sprint value) {

}

public void workRemaining(Sprint value) {

}

1. Explaining terminologies: -
2. Objects have state, behavior and identity

An object has attributes and values associated with it, which defines its state. An object has definite task to perform to achieve its goal which is defined in its behavior or operation. The unique value possessed by an individual object irrespective to that of its attribute or operation is known as identity.

1. Encapsulation and Data Hiding

Wrapping the data and the methods that operates on data in a single entity which helps to keep the data safe from unwanted alteration is Encapsulation.

Hiding irrelevant and unnecessary detain from client is known as Data Hiding.

1. Inheritance (generalization)

The feature of OOP that shows a relationship between two classes where one class, called a subclass inherits the attributes and operations of another class, called its superclass.

1. Polymorphism and Late binding

Static polymorphism is achieved by using Operator overloading and Function overloading where we get object of the class at compile time. Late binding is achieved by using virtual function where we get the object at run time.

1. Delegation and Propagation

Delegation is the relationship between two objects where one object calls the method of another object to achieve the task. Such process forms a chain where one objects calls other and so on and forth until the task is achieves which gives rise to the propagation pattern amongst the objects.