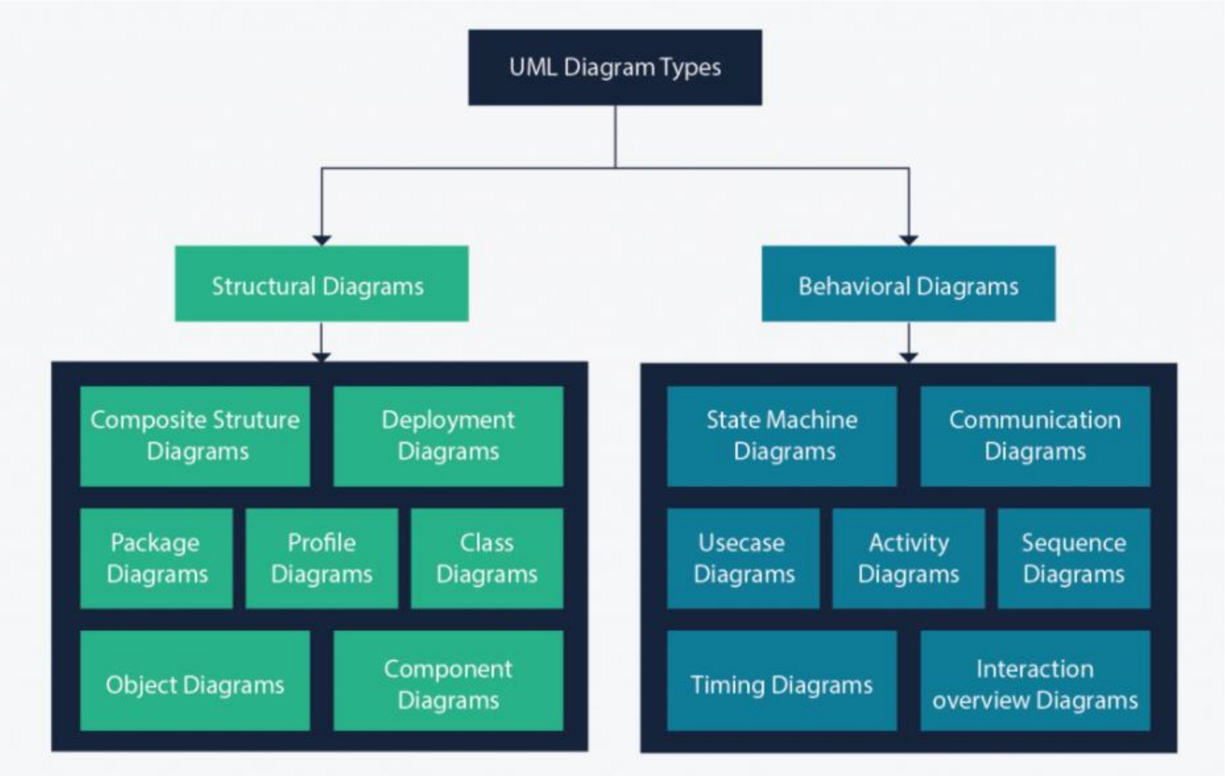
GROUP 10

INVENTION MANAGEMENT SYSTEM

The Invention Management System Database is made to store

- Invention details
- Inventor details
- Awards Received etc.

UML DIAGRAM:



Structural diagrams:

- Structural diagrams show the things in the modeled system.
- In a more technical term, they show different objects in a system.

Behavioral diagrams:

- Behavioral Diagrams show what should happen in a system.
- They describe how the objects interact with each other to create a functioning system.

Use Case Diagram:

- Use case diagrams give a graphic overview of the actors involved in a system, different functions needed by those actors and how these different functions interact.

- It's a great starting point for any project discussion because you can easily identify the main actors involved and the main processes of the system.

- This Use Case Diagram depicts the High-level view of the Invention Management system.

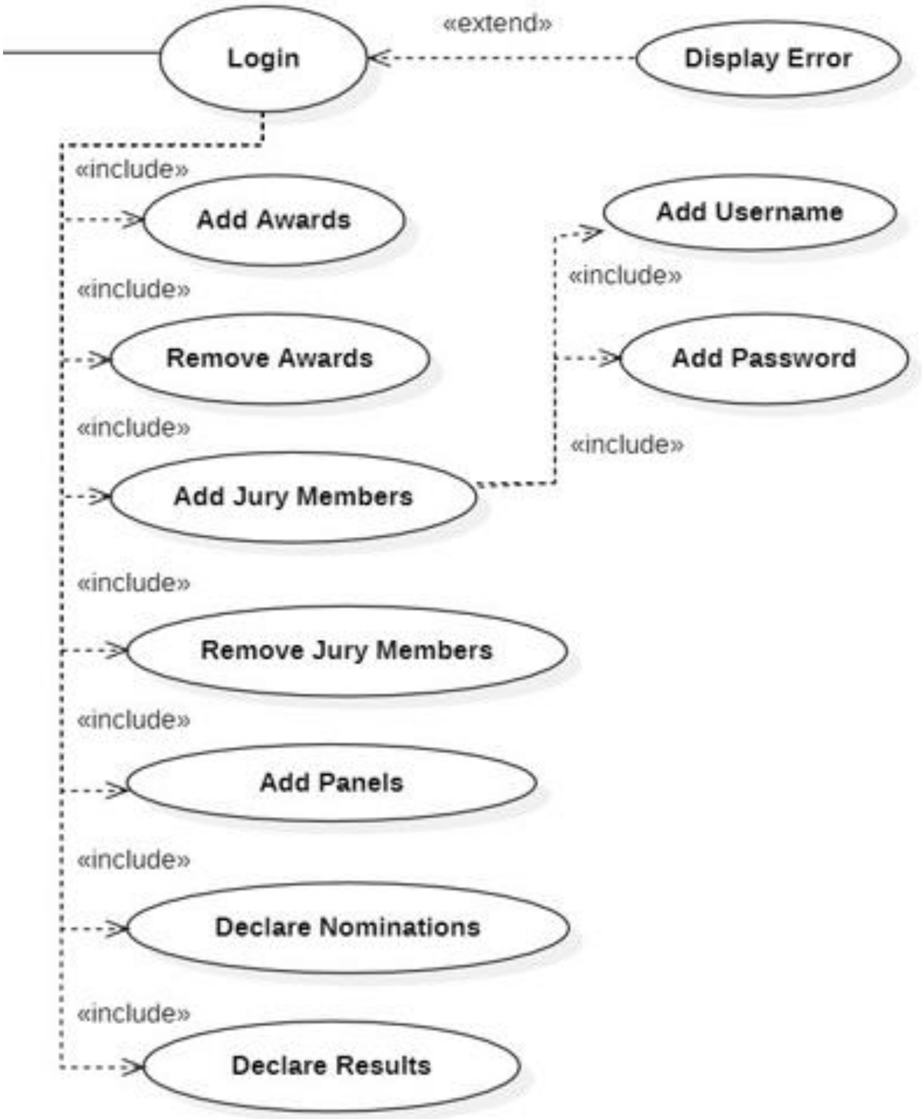
It also provides the scenarios in which the application interacts with,

- Inventor
- Jury
- Admin

Actor Category	Actor
Primary Actor	Jury, Admin
Secondary Actor	Inventor

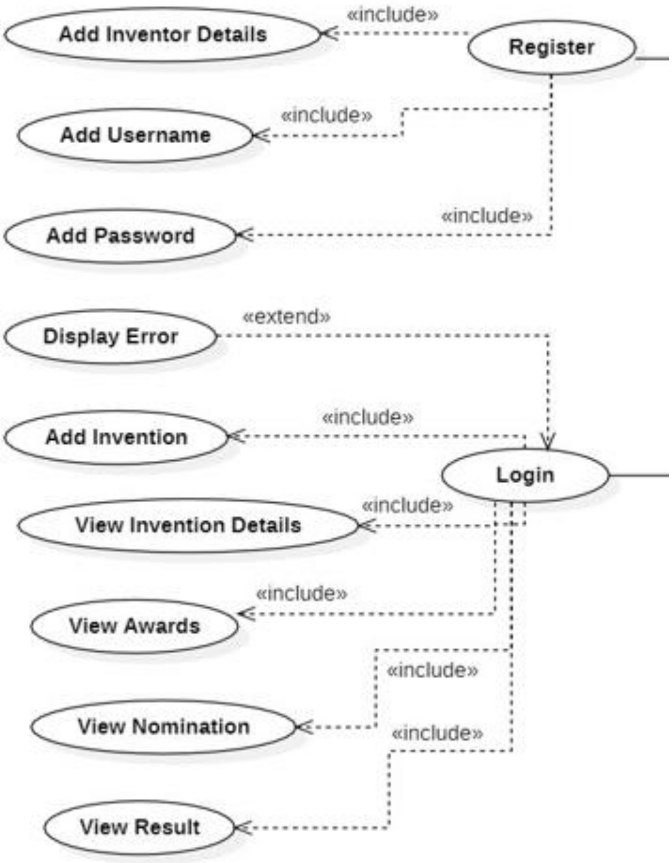
There are total of Twenty-Six use cases that represent the specific functionality of Invention Management System.

Each actor interacts with a particular use case.



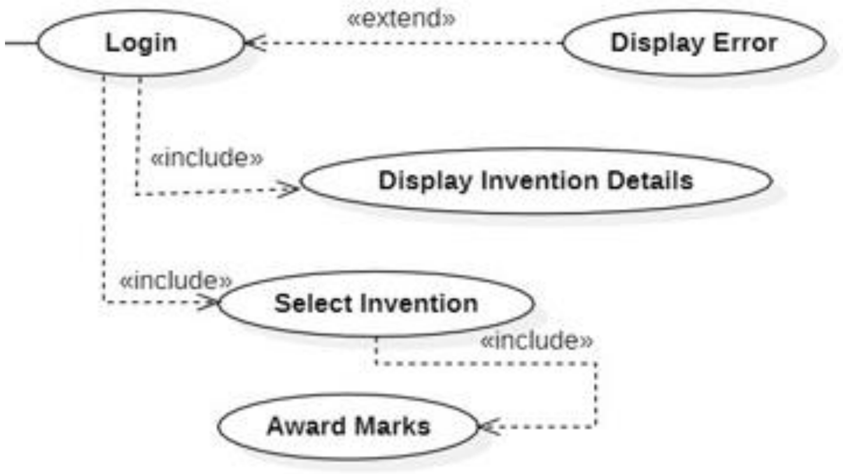
Functionality of Admin:

- Login to take the overall Control of the Data Base
- Add or Remove Awards
- Add or Remove Panel
- Add or Remove Jury
- Declare Nominations
- Declare Results
- Admin takes the overall control of the database or in other words say one of the primary Actors.
- Admin have to just login inside the database and gets the overall control.



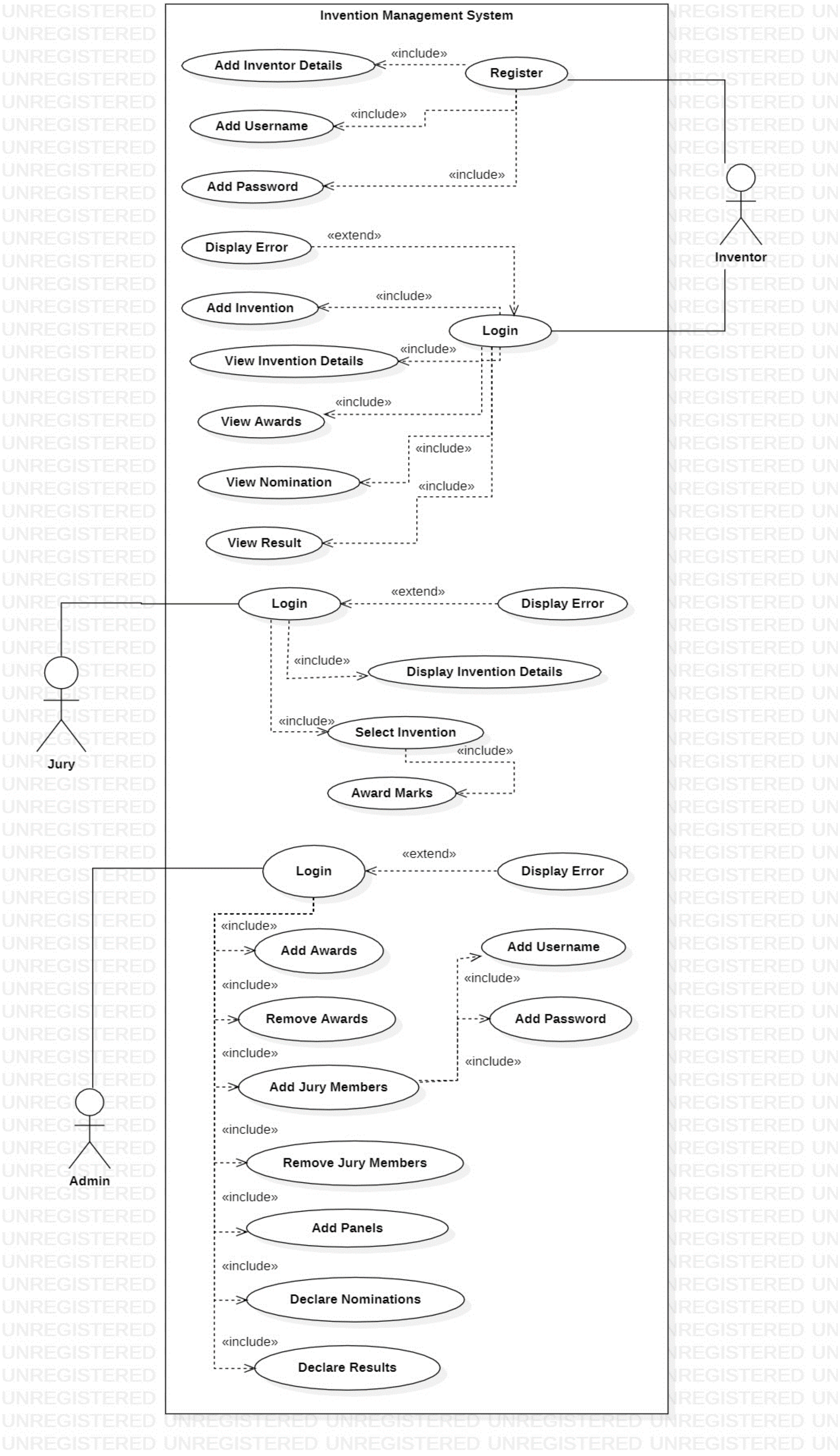
Functionality of Inventor Actor:

- Register in the database to create Account
- Register for New Username and Password
- Login using his/her credentials
- Add the details of his/her invention
- View the Invention details
- View Awards
- View Nominations
- View Results
- Inventor can create the account in the database to register his/her invention.
- Once registered Inventor needs to set the username and password to login into the database next time.
- Once Inventor Actor successfully logins, gets option to add the invention to the database.
- If Inventor is not able to login successfully then, they will get the Error message.
- Inventor can also view the invention details that he/she have enclosed while registering in the database.
- Inventor can view the results of the invention.



Functionality of Jury Actor:

- Login to Invention Management System
- View Invention Details that are Displayed when logged In.
- Select the Invention
- Award Marks for the Selected Invention.
- Jury can login to the Invention management system using the credentials provided by the Admin.
- Once they successfully login inside the Portal they get access to all invention.
- If they are not able to login successfully then they will get the Error message.
- From the list of inventions given they can select the allotted invention.
- They can look into the inventions and award marks to the selected Invention.


Invention Management System
Register
Add Inventor Details
Add Username
Add Password
«include»
«include»
«include»
Inventor
Display Error
«extend»
Login
Add Invention
«include»
View Invention Details
«include»
View Awards
«include»
View Nomination
«include»
View Result
«include»
Jury
Login
Display Error
«extend»
Display Invention Details
«include»
Select Invention
«include»
Award Marks
«include»
Admin
Login
Display Error
«extend»
Add Awards
«include»
Remove Awards
«include»
Add Jury Members
«include»
Add Username
Add Password
«include»
«include»
Remove Jury Members
«include»
Add Panels
«include»
Declare Nominations
«include»
Declare Results
«include»

Class Diagram:

- Class diagrams are the main building block of any object-oriented solution.
- It shows the classes in a system, attributes, and operations of each class and the relationship between each class.
- In most modeling tools, a class has three parts.
- Name at the top, attributes in the middle and operations or methods at the bottom.
- In a large system with many related classes, classes are grouped together to create class diagrams.
- Different relationships between classes are shown by different types of arrows.

Public (+):

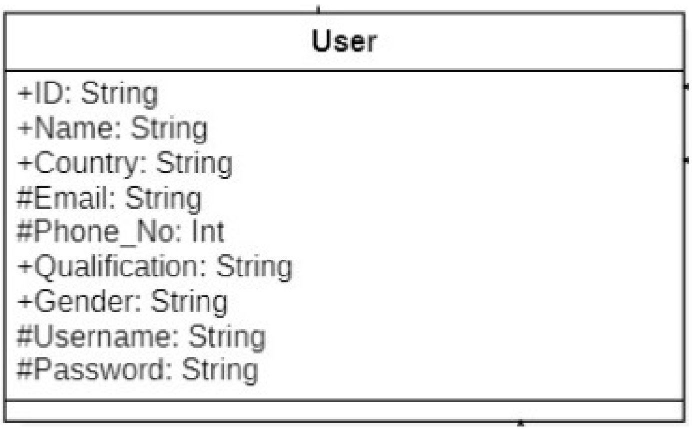
- Public members are visible to all other classes.
- This means that any other class can access a public field or method.
- Further, other classes can modify public fields unless the field is declared as final.

Protected (#):

- The **protected** keyword is an access modifier used for attributes, methods and constructors, making them accessible in the same package and subclasses.

Private (-):

- The methods or data members declared as private are accessible only within the class in which they are declared.
- The access level of a private modifier is only within the class. It cannot be accessed from outside the class.
- Any other class of the same package will not be able to access these members.



User

+ID: String
+Name: String
+Country: String
#Email: String
#Phone_No: Int
+Qualification: String
+Gender: String
#Username: String
#Password: String

- **User** is Implemented from the Interface **Login Details.**

+ ID:

- Unique ID to identify the User.
- It is of type String.

+ Name:

- Name of the User.
- It is of type String.

+ Country:

- Country in which the user resides.
- It is of type String.

+ Qualification:

- An experience that makes the user suitable for a particular job or activity.
- It is of type String.

+ Gender:

- To specify the Gender of the User.
- It is of type String.

# Email:

- Email of the User.
- It is of type String.

# Phone_No:

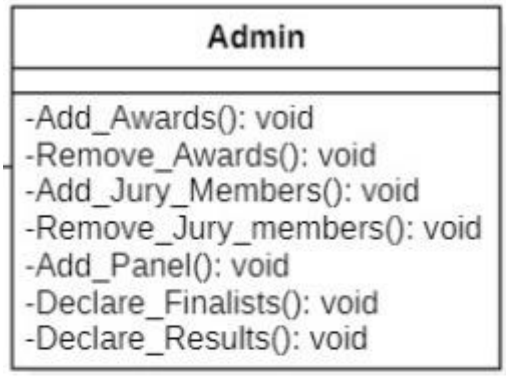
- Phone Number of the User.
- It is of type Integer.

# Username:

- Unique Username that each user has to login to his/her account.
- It is of type String.

# Password:

- Password that each user has to login to his/her account.
- It is of type String.



Admin class is Inherited from the Parent Class User.

It Inherits all properties of the User and it has its own Methods.

- Add_Awards():

- This Method is used to add the available Awards to the DataBase.
- These Awards will be given to the Inventor based on their Invention.
- It doesn’t take any parameter and doesn’t return anything.
- It just asks for the name of the Award that has to added when it is called.

- Remove_Awards():

- This method is used to remove the Awards from the DataBase.
- It doesn't take any parameter and doesn't return anything.

- Add_Jury_Members():

- This Method is used to add the details of the Jury Members to the DataBase.
- It doesn't take any parameter and doesn't return anything.
- It just asks for the details of the Jury that has to added when it is called.

- Remove_Jury_Members():

- This method is used to remove the details of the Jury Members from the DataBase.
- It doesn't take any parameter and doesn't return anything.

- Add_Panel():

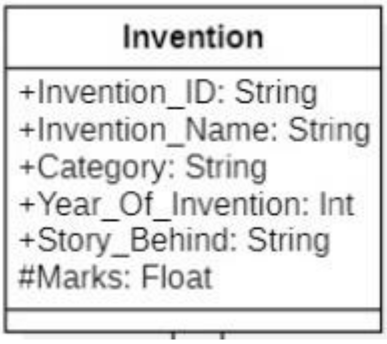
- This Method is used to add the Panel to the DataBase.
- It doesn't take any parameter and doesn't return anything.
- It just asks for the details of the Panel that has to added when it is called.

- Declare_Finalists():

- This method is used to declare the names of the Inventors who have been selected to the Finals.

- Declare_Results():

- This method is used to declare Finale results.



+ Invention_ID:

- Unique ID to identify the Invention.
- It is of type String.

+ Invention_Name:

- Name of the Invention.
- It is of type String.

+ Category:

- This attribute defines the category In which the Invention belongs to.
- It is of type String.

+ Year_Of_Invention:

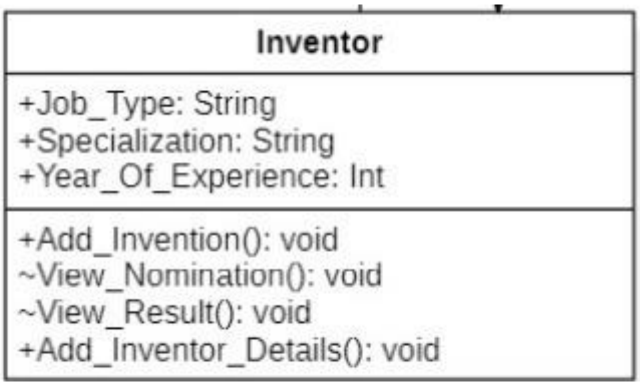
- Year in which the invention has been invented.
- It is of type Integer.

+ Story_Behind:

- This defines the motive and the reason behind Inventing the particular Invention.
- It is of type String.

# Marks:

- Marks that has been awarded for the particular invention by the Jury.



Inventor

+Job_Type: String
+Specialization: String
+Year_Of_Experience: Int

+Add_Invention(): void
~View_Nomination(): void
~View_Result(): void
+Add_Inventor_Details(): void

+ Job_Type:

- This attribute stores the current job of the Inventor
- This is of type String.

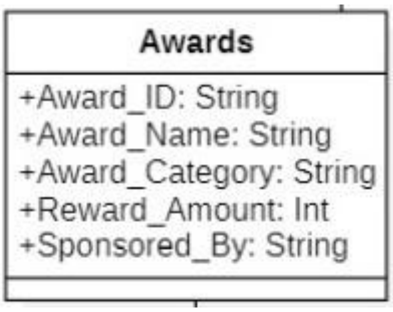
+ Specialization:

- This stores the academic specialization of the inventor.
- This is of type String.

+ Year_Of_Experience

- This attribute stores the years of experience of the inventor.
- It is of type integer(int).

- **Add_Invention()**
  - Using this method, we add the inventions of the inventor which has been nominated to the database.
- **View_Nomination()**
  - This method displays the inventions of the inventor which have been nominated.
- **View_Result()**
  - This method displays the inventions which has been shortlisted as winners.
- **Add_Inventor_Details()**
  - This method is used to add the details of the invention into the database.



Awards

+Award_ID: String
+Award_Name: String
+Award_Category: String
+Reward_Amount: Int
+Sponsored_By: String

+ Award_ID

- This attribute is used to store the id of the award.
- This is of type String.

+ Award_Name

- This attribute is used to store the name of the award.
- This is of type String.

+ Award_Category

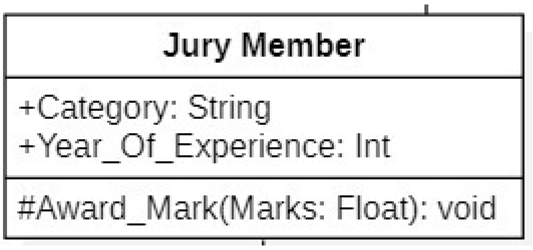
- The invention category is stored in this attribute.
- This is of type String.

+ Reward_Amount

- The prize money given to the winner in this category is stored in this attribute.
- This is of type integer(int).

+ Sponsored_By

- The sponsor for the prize money is stored in this attribute.
- This is of type String.



+ Category

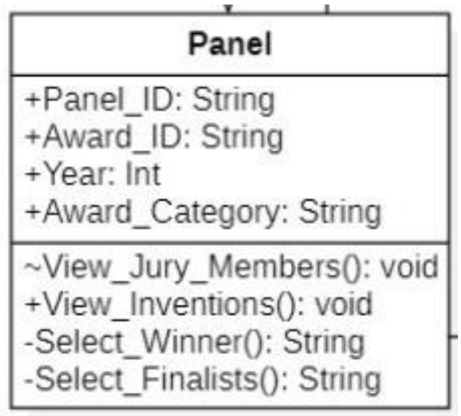
- The category in the award ceremony where the particular person is part of the jury is stored in this attribute.
- This is of type String.

+ Year_Of_Experience

- The years of experience of the Jury Member is stored in this attribute.
- It is of type Integer(int).

# Award_Marks

- It stores the marks that has been awarded for the invention.
- This takes marks as the parameter which is of type Float.
- It doesn’t return anything.



Panel

+Panel_ID: String
+Award_ID: String
+Year: Int
+Award_Category: String

~View_Jury_Members(): void
+View_Inventions(): void
-Select_Winner(): String
-Select_Finalists(): String

+ Panel_ID

- This attribute stores the ID of the particular panel.
- It is of type String.

+ Award_ID

- This attribute stores the ID of the award for which the particular panel is judging.
- It is of type String.

+ Year:

- It stores the year of the award ceremony.
- It is of type Integer(int).

+ Award_Category:

- This attribute stores the category of the award for which the panel is judging for.
- It is of type String.

- View_Jury_Members()

- This method displays the jury members present in the particular panel.

- View_Inventions()

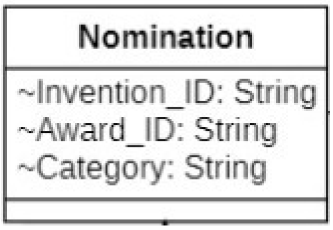
- This method displays the inventions which are part of the category the panel is judging over.

- Select_Winner()

- This method is used to determine the winner in the particular category.
- The winner is returned as a String.

- Select_Finalists()

- This method is used to find out the qualifying inventions in the preliminary round.
- The qualifying inventions are returned as a String.



+ Invention_ID

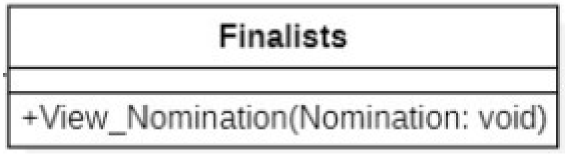
- This attribute stores the invention ID of the particular invention which has been nominated.
- It is of type String.

+ Award_ID

- This attribute stores the award ID of the invention which has been nominated.
- It is of type String.

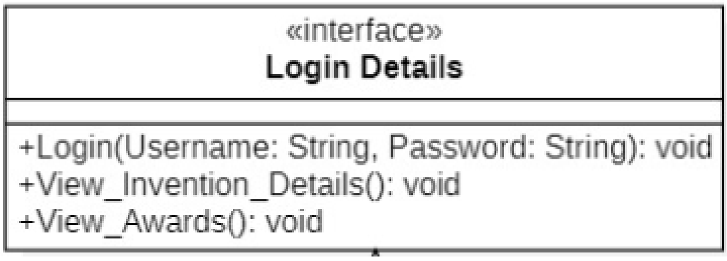
+ Category

- This attribute stores the category of the invention,
- It is of type String.



+ View_Nominations:

- It shows all the nominations of the finalists.



«interface»
Login Details

+Login(Username: String, Password: String): void
+View_Invention_Details(): void
+View_Awards(): void

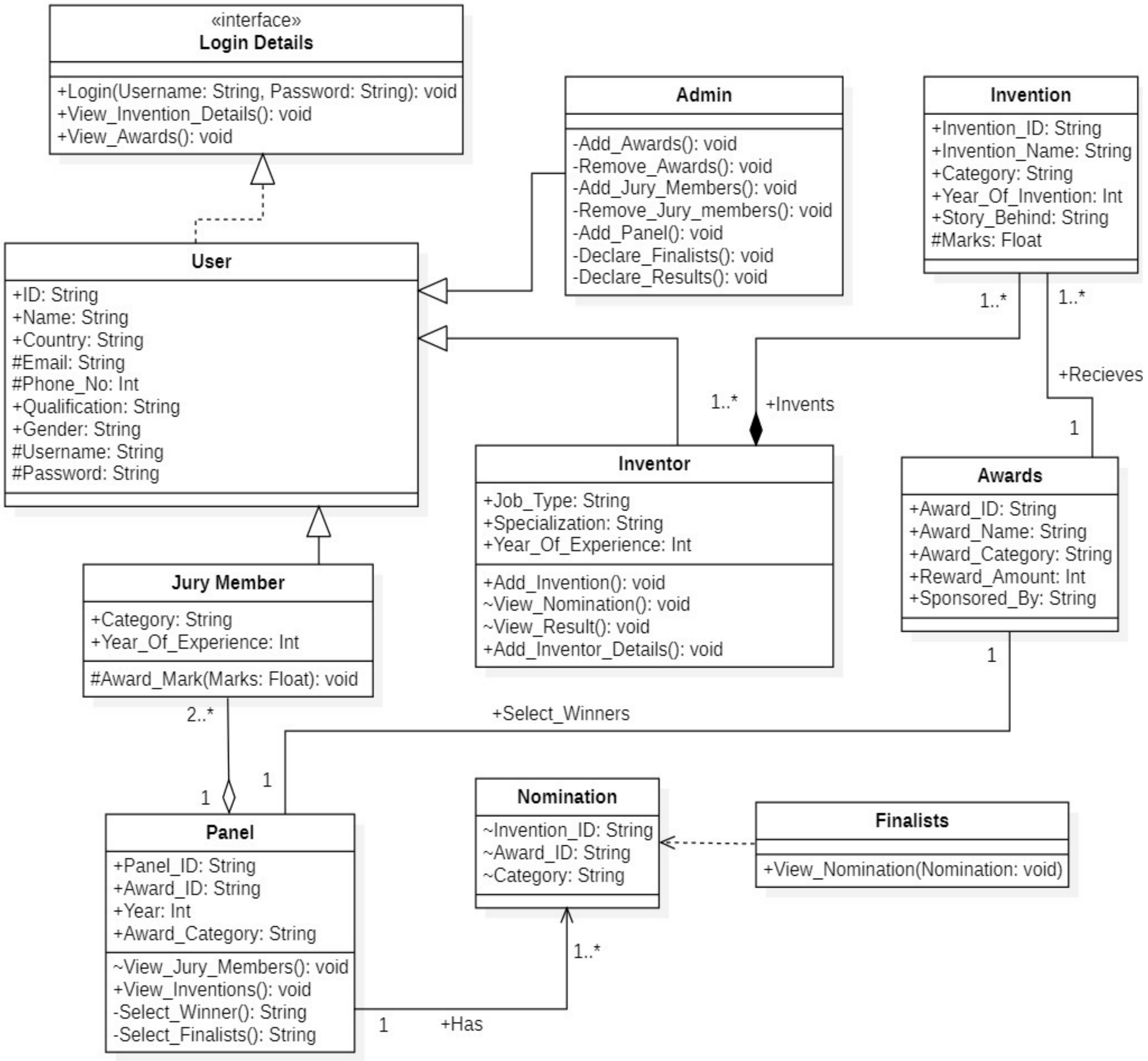
Login:

- This takes the Username and Password as arguments which is of type String.
- It doesn't return anything.
- Using this credential, the user can login to the Portal/System.

ASSOCIATION	Invention ⟷ Award Awards ⟷ Panel
	• One Award can be given to one or more Invention. • Panel selects only one winner for a particular Award. • One Award can be selected by only One Panel.

REALIZATION	Login Details ⬅ User
	• **User** implements an interface **Login Details**, thereby inheriting the abstract methods of the **Login details**.
GENERALIZATION	USER ⬅ Admin ⬅ Inventor ⬅ Jury Member
	• Admin Inherits the properties of User. • Inventor Inherits the properties of User. • Jury Member Inherits the properties of User.
DEPENDENCY	Nominations ⬅ Finalists
	• Finalists Dependent on the Nominations since an object of Nominations is being used by the Finalists.
AGGREGATION	Jury Member ➡ Panel
	• The Jury Member can exist independently of the Panel.

COMPOSITION	Invention → Inventor
	- Invention cannot exist without the Inventor.



Done By,

- ✓ S. Abhishek - AM.EN.U4CSE19147
- ✓ Rahan Manoj – AM.EN.U4CSE19144
- ✓ Harsha Sathish – AM.EN.U4CSE19123
- ✓ Arvind Kumar K – AM.EN.U4CSE19109

THANKYOU!!!