Training And Placement Management System

Project By :

ANUSHKA SIDDHU : RA2111026010214

BISHAL DE : RA2111026010231

Subject :

Object Oriented Design And Programming

18CSC202J

Description:

This project **"Training and Placement Management System"** is developed to maintain the training and placement details of all the students, It also stores the students personal and academic details. It has a trace over the training details of student and most important It has details of all the companies coming for recruitments.

Using this project **"Student"** can check status of his application form and all the necessary details which are provided by the recruiters. Students can also view the list of companies available and can apply according to their preferences.

"Companies" coming for recruitments can also make their profile and add their respective criteria. Companies has a right to access all the list of students who are eligible for their criteria.
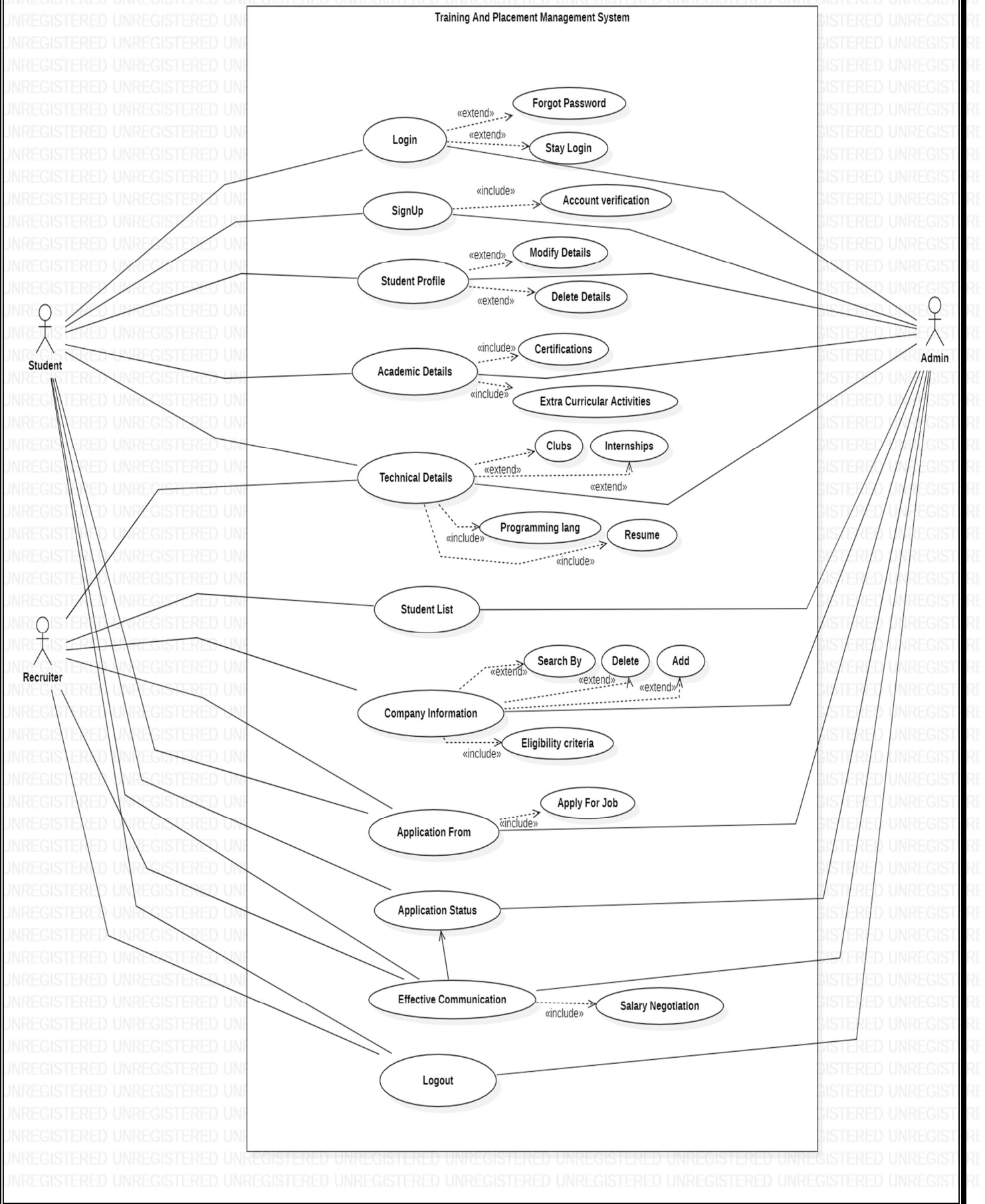
In the same way in this project one more user is involved he is most important person who controlls all the information and also makes the necessary changes when they are raised. He is the **"Administrator"**. He can do the updates and set the job information regarding the company and change the users name and passwords such type of all these things are made by the administrator level.
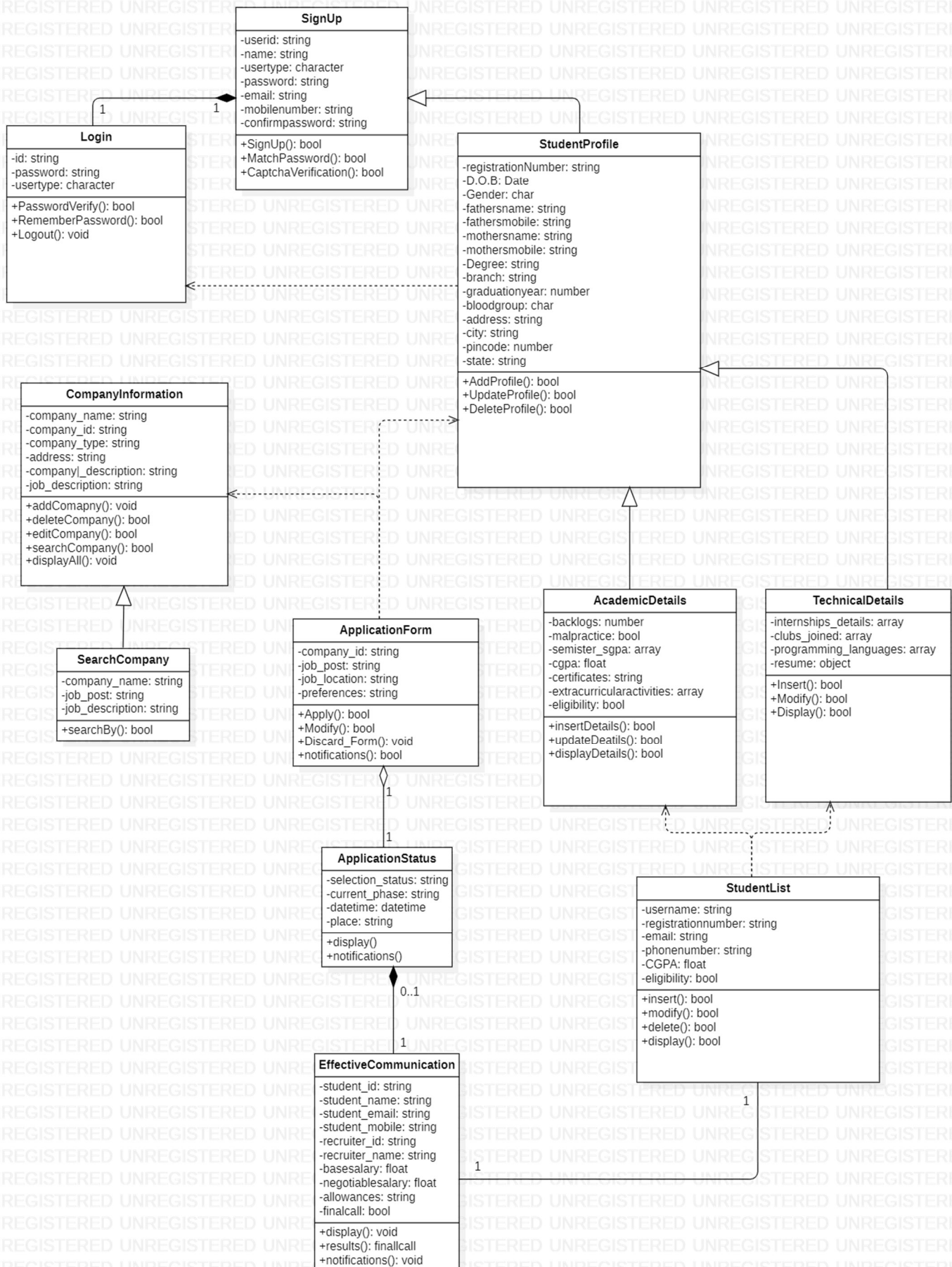
Use Case Diagram:

Project By :-
1) ANUSHKA SIDDHU : RA2111026010214
2) BISHAL DE : RA2111026010231

Training And Placement Management System

Class Diagram:

SignUp
- -userid: string
- -name: string
- -usertype: character
- -password: string
- -email: string
- -mobilenumber: string
- -confirmpassword: string

- +SignUp(): bool
- +MatchPassword(): bool
- +CaptchaVerification(): bool

Login
- -id: string
- -password: string
- -usertype: character

- +PasswordVerify(): bool
- +RememberPassword(): bool
- +Logout(): void

StudentProfile
- -registrationNumber: string
- -D.O.B: Date
- -Gender: char
- -fathersname: string
- -fathersmobile: string
- -mothersname: string
- -mothersmobile: string
- -Degree: string
- -branch: string
- -graduationyear: number
- -bloodgroup: char
- -address: string
- -city: string
- -pincode: number
- -state: string

- +AddProfile(): bool
- +UpdateProfile(): bool
- +DeleteProfile(): bool

CompanyInformation
- -company_name: string
- -company_id: string
- -company_type: string
- -address: string
- -company|_description: string
- -job_description: string

- +addComapny(): void
- +deleteCompany(): bool
- +editCompany(): bool
- +searchCompany(): bool
- +displayAll(): void

SearchCompany
- -company_name: string
- -job_post: string
- -job_description: string

- +searchBy(): bool

ApplicationForm
- -company_id: string
- -job_post: string
- -job_location: string
- -preferences: string

- +Apply(): bool
- +Modify(): bool
- +Discard_Form(): void
- +notifications(): bool

AcademicDetails
- -backlogs: number
- -malpractice: bool
- -semister_sgpa: array
- -cgpa: float
- -certificates: string
- -extracurricularactivities: array
- -eligibility: bool

- +insertDetails(): bool
- +updateDeatils(): bool
- +displayDetails(): bool

TechnicalDetails
- -internships_details: array
- -clubs_joined: array
- -programming_languages: array
- -resume: object

- +Insert(): bool
- +Modify(): bool
- +Display(): bool

ApplicationStatus
- -selection_status: string
- -current_phase: string
- -datetime: datetime
- -place: string

- +display()
- +notifications()

StudentList
- -username: string
- -registrationnumber: string
- -email: string
- -phonenumber: string
- -CGPA: float
- -eligibility: bool

- +insert(): bool
- +modify(): bool
- +delete(): bool
- +display(): bool

EffectiveCommunication
- -student_id: string
- -student_name: string
- -student_email: string
- -student_mobile: string
- -recruiter_id: string
- -recruiter_name: string
- -basesalary: float
- -negotiablesalary: float
- -allowances: string
- -finalcall: bool

- +display(): void
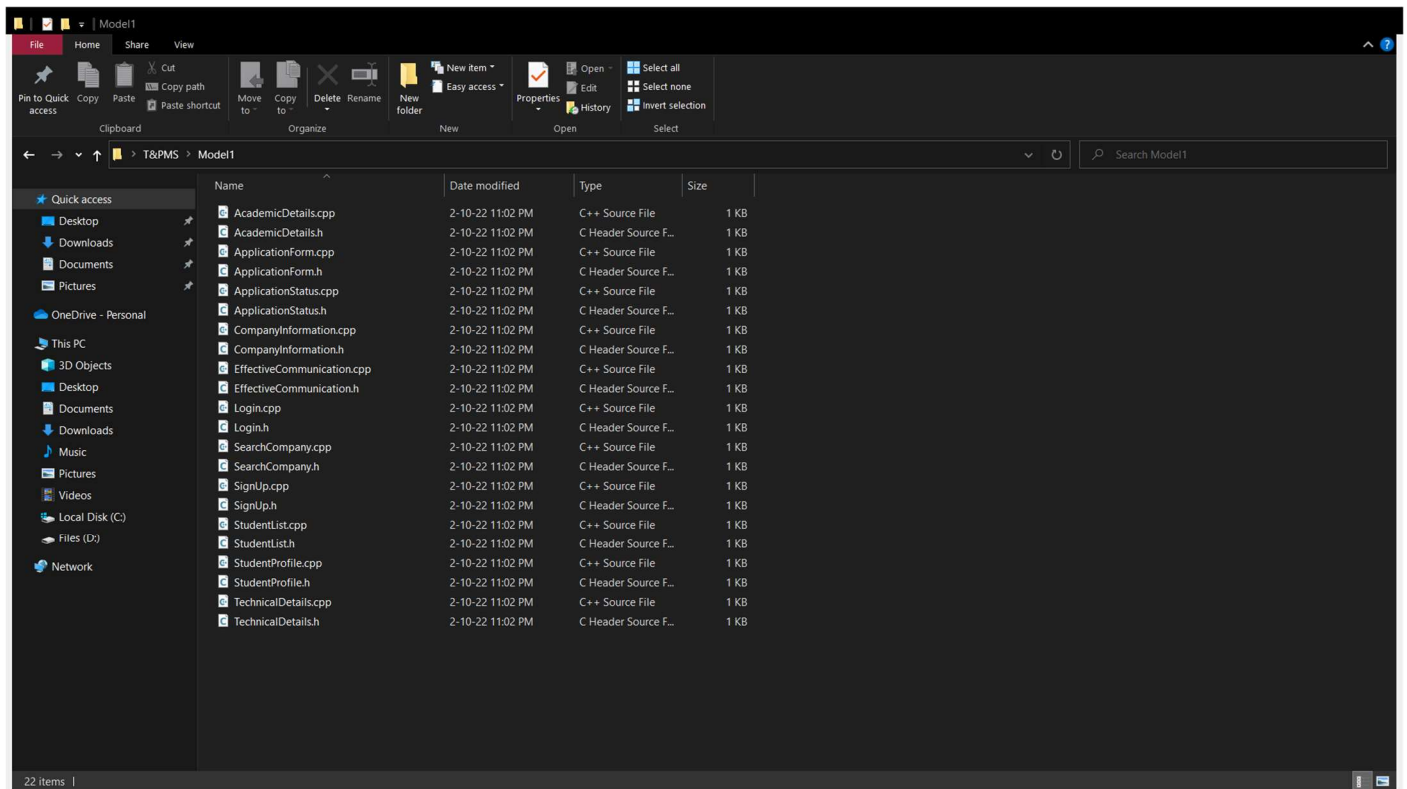- +results(): finallcall
- +notifications(): void

Relationships:

- **Signup & Login :** Composition is used because without creating an account a user can't login in the application. So, Login is not possible without Signup. And the multiplicity is 1 .. 1.

- **Signup & StudentProfile :** Generalisation{inheritance} is used as the signup details are inherited to the studentprofile. StudentProfile is a child class of SignUp that contains the details of SignUp as well as it own properties.

- **Login & StudentProfile :** Dependency is used as Studentprofile is dependent on Login. After login only a user can have access to the StudentProfile.

- **CompanyInformation & SearchCompany :** Generalisation{inheritance} is prefer here as CompanyInformation are inherited to the sub class SearchCompany, where the student can search for a company on the basis of his preference. SearchCompany has the attributes of CompanyInformation & has some of its own properties.

- **AcademicDetails & TechnicalDetails & StudentProfile :** Inheritance {Hierarchical inheritance} is used as both the AcademicDetails & TechnicalDetails are sub class for StudentProfile. Both these classes contains the basic personal information of all the students and plus some of the characteristic of the respective classes.

- **ApplicationForm & CompanyInformation & StudentProfile :** Dependency is used as ApplicationFrom takes the information from StudentProfile & CompanyInformaton, without both of them it is not possible for this class to work.

- **ApplicationForm & ApplicationStatus :** Aggregation is used as even without applying for ApplicationForm ,the ApplicationStatus portal can show "No application filled". So, both the classes are independent classes, but if the ApplicationForm is filled, Application status will show whether the form is approved or not. And the Multiplicity is 1, 0..1 as there can be filled form or else there will be no form filled by the user.

- **Effective Communication and Application Status :** Composition is used because if the application form is approved then only the Effective Communication will take place otherwise it won't take. And the multiplicity is 1, 0..1.

⊞ **Effective Communication and Student List : Association** is used because if two classes that is Student List and Effective Communication need to communicate with each other then there must be a link between them and an association denotes that link.The multiplicity is 1 .. 1

Code Folder Screenshot:

Github Link ⬤ :

https://github.com/BishalDe/Training-And-Placement-Management