Topic : Construction Management System

Group no : MLB_WE_01.02_03

Campus : Malabe

Submission Date: 17th May 2022

We declare that this is our own work, and this Assignment does not incorporate without acknowledgment any material previously submitted by anyone else in SLIIT or any other university/Institute. And we declare that each one of us equally contributed to the completion of this Assignment.

Registration No	Name	Contact Number
IT21237072	Subasinghe S.S.Y.H.	077 920 0393
IT21467448	Nalinka G.K.	076 484 6325

1 System Requirements for Construction Management System

1. Guests can register into the system
2. Registered clients can login, after providing the username and the password
3. Registered clients can arrange a meeting with a representative agent
4. For arrange a meeting client needs to enter available dates, time slots and a place
5. Representative manager looks up on those details and sends a confirmation email with details
6. Clients can see their projects, after they enter the project number and the contract number
7. Project page shows project details such as project progress, project engineer's details, materials, and quality of materials
8. Project progress manager frequently update system about project progress
9. Material manager order suitable materials
10. Material quality manager measure quality of materials and accept suitable quality materials

Identified classes:

- Client
- Representative agent manager
- Project progress manager
- Material manager
- Material quality manager

3 CRC cards

Representative agent class	
responsibilities	collaborations

client class	
responsibilities	collaborations
login	
Request a meeting and enter meeting details	Representative agent manager
Check project	file

Representative agent manager class	
responsibilities	collaborations
Check meeting details	client
Send meeting details	Representative agent
send confirmation email	client '

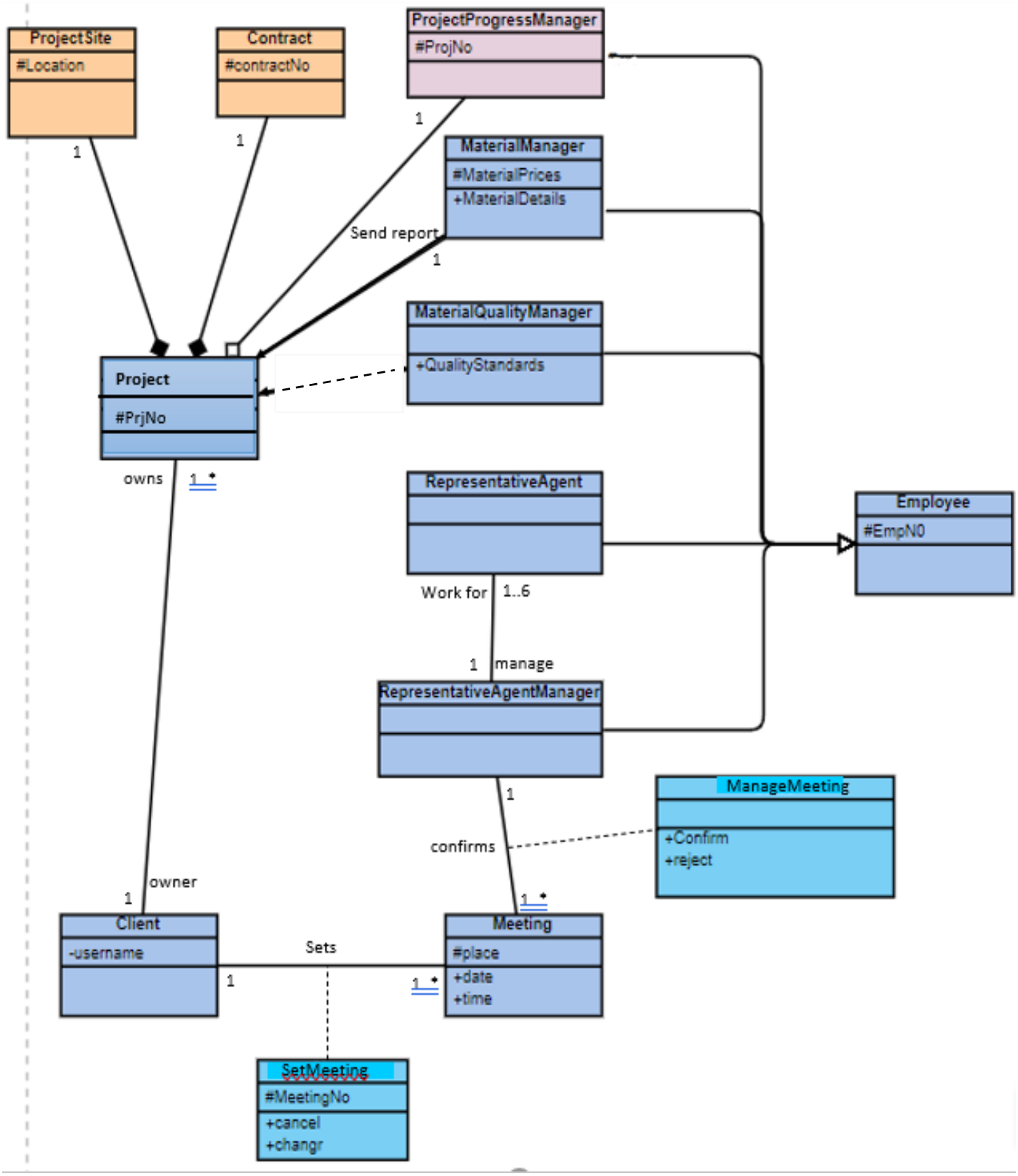
File class	
responsibilities	collaborations
Generate project report	Project progress manager, Material quality manager, material manager

Material quality manager class	
responsibilities	collaborations
Send material quality details	file

Project progress manager class	
responsibilities	collaborations
Send project progress details	file

material manager class	
responsibilities	collaborations
Send material details	file

Exercise 1- class diagram



Exercise 2- codes

```
#include<iostream>

using namespace std;


class employee{

protected:

int Eno[8];

};


class ProjectProgressManager:public employee

{

protected:

int ProjNo[5];

};


class MaterialManager:public employee

{

protected:

int MaterialPrices[];

public:

char MaterialDetails[];

};


class MaterialQualityManager:public employee

{

public:

char QualityStandards[];

};


class RepresentativeAgent:public employee
```

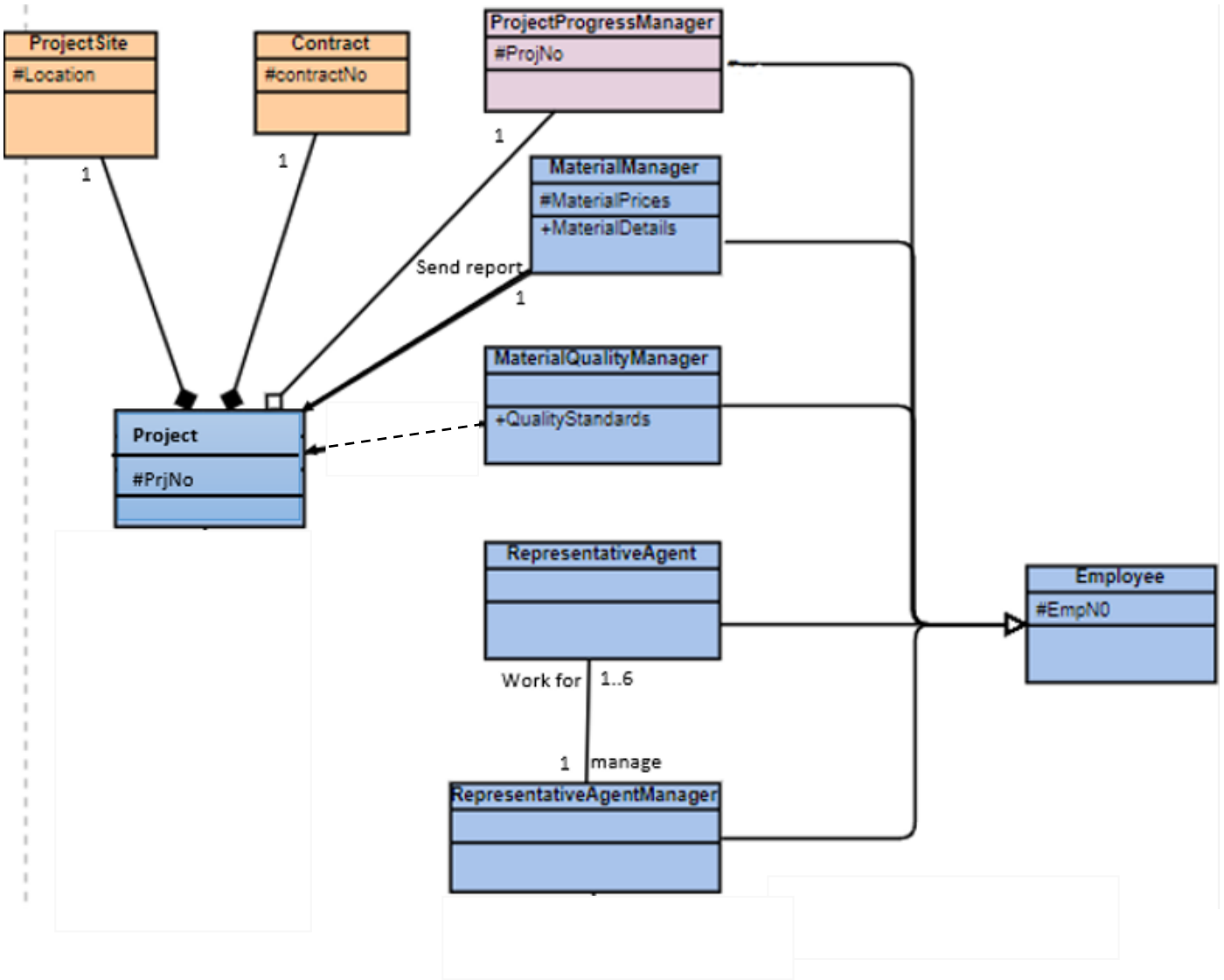
```
{
private:
// bi-directional association
RepresentativeAgentManager*RepresentativeAgentManager[1];
};

class RepresentativeAgentManager:public employee
{
private:.
// bi-directional association
Meeting*Meeting[];
RepresentativeAgen*RepresentativeAgen[];
// dependancy
void addManageMeeting(ManageMeeting*ManageMeeting);
};

class Meeting{
protected:
char place[];
// bi-directional association
Client*Client[];
public:
char date[];
char time[];
};

class Client{
private:
// bi-directional association
Project*Project;
Meeting*Meeting[];
// dependancy
void addSetMeeting(SetMeeting*SetMeeting);
```

```
protected:
char UserName[];
};

class Project{
protected:
int PrjNo[5];

//composition
ProjectSite*ProjectSite[1];
Contract*Contract[1];

// aggregation
ProjectProgressManager*ProjectProgressmanager[1];

// uni-directional association
MaterialManager*MaterialManager;
// bi-directional association
Client*Client[];

public:
// composition
void addProjectSite()
void addContract
{
        ProjectSite[0]=Projectsite();
        Contract[0]=Contract();
//Dependency
MaterialQualityManager*MaterialQualityManager;

}
// aggregation
void addProjectProgressManager(ProjectProgressManager*ProjectProgressManager)
```

```
{
        ProjectProgressManager[1]=ProjectProgressManager;
}
};

class ProjectSite{
protected:
char Location[];
};

class Contract{
protected:
int ContractNo[5];
};

class SetMeeting{
        protected:
        int MeetingNo[5];
};

class ManageMeeting{
};

return 0;
```

Individual contribution

IT21237072 – Subasinghe S.S.Y.H.



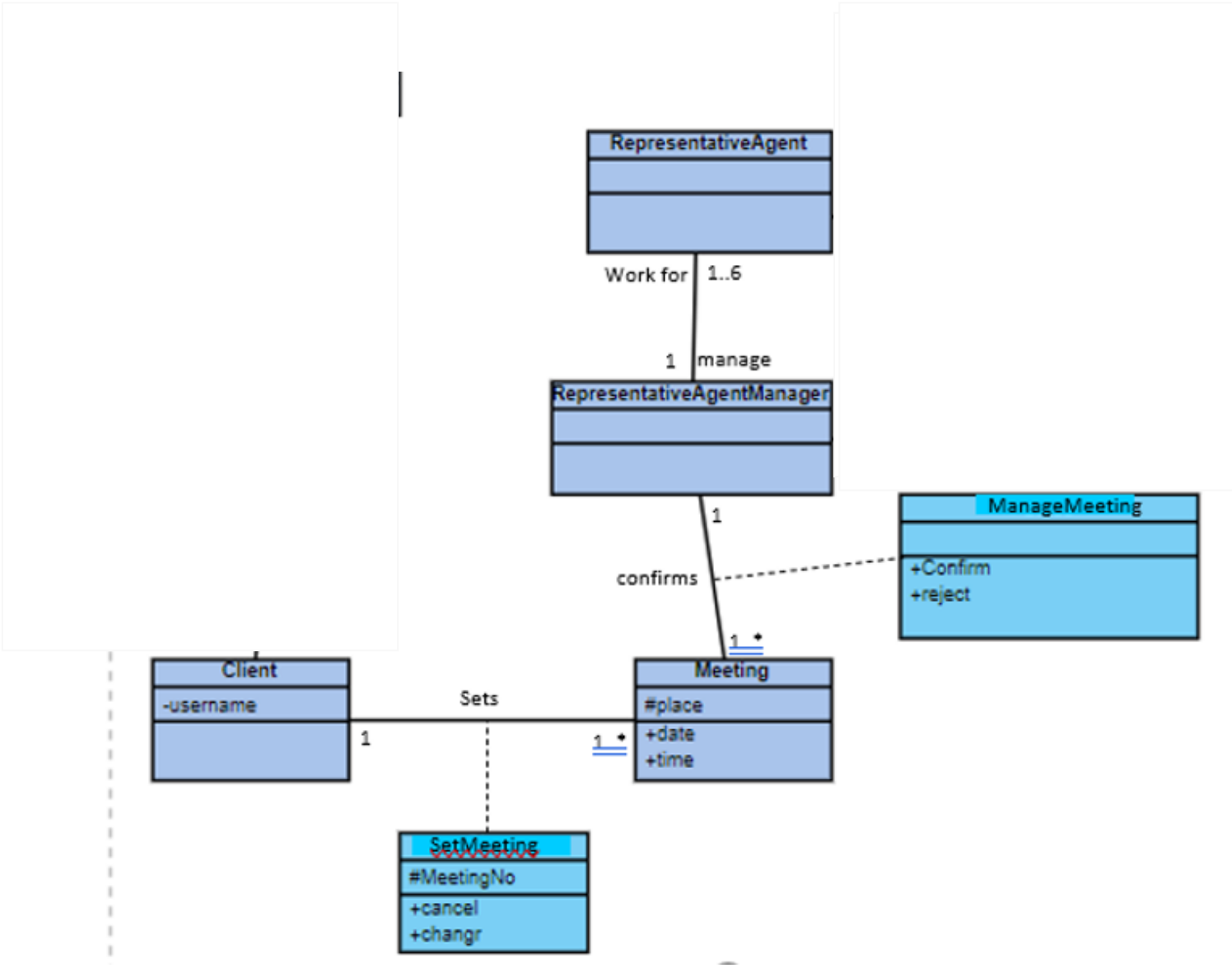

```
#include<iostream>

using namespace std;


class employee{

protected:

int Eno[8];
```

```
};

class ProjectProgressManager:public employee
{
protected:
int ProjNo[5];
};

class MaterialManager:public employee
{
protected:
int MaterialPrices[];
public:
char MaterialDetails[];
};

class MaterialQualityManager:public employee
{
public:
char QualityStandards[];
};

class RepresentativeAgent:public employee
{
private:
// bi-directional association
RepresentativeAgentManager*RepresentativeAgentManager[1];
};

class RepresentativeAgentManager:public employee
{
private:.
// bi-directional association
```

```
Meeting*Meeting[];
RepresentativeAgen*RepresentativeAgen[];

};

class Project{
protected:
int PrjNo[5];

//composition
ProjectSite*ProjectSite[1];
Contract*Contract[1];

// aggregation
ProjectProgressManager*ProjectProgressmanager[1];

// uni-directional association
MaterialManager*MaterialManager;
public:
// composition
void addProjectSite()
void addContract
{
        ProjectSite[0]=Projectsite();
        Contract[0]=Contract();
}
// aggregation
void addProjectProgressManager(ProjectProgressManager*ProjectProgressManager)
{
        ProjectProgressManager[1]=ProjectProgressManager;
// Dependency
MaterialQualityManager*MaterialQualityManager;
```

```
}
};

class ProjectSite{
protected:
char Location[];
};

class Contract{
protected:
int ContractNo[5];
};

return 0;
```

IT21467448 – Nalinka G.K.




```
#include<iostream>

using namespace std;


class RepresentativeAgent:public employee

{

private:

// bi-directional association

RepresentativeAgentManager*RepresentativeAgentManager[1];
```

```
};

class RepresentativeAgentManager:public employee
{
private:.
// bi-directional association
Meeting*Meeting[];
RepresentativeAgen*RepresentativeAgen[];
// dependancy
void addManageMeeting(ManageMeeting*ManageMeeting);
};

class Meeting{
protected:
char place[];
// bi-directional association
Client*Client[];
public:
char date[];
char time[];
};

class Client{
private:
// bi-directional association
Project*Project;
Meeting*Meeting[];
// dependancy
void addSetMeeting(SetMeeting*SetMeeting);
protected:
char UserName[];
};
```

```
class SetMeeting{
        protected:
        int MeetingNo[5];
};

class ManageMeeting{
};

return 0;
```