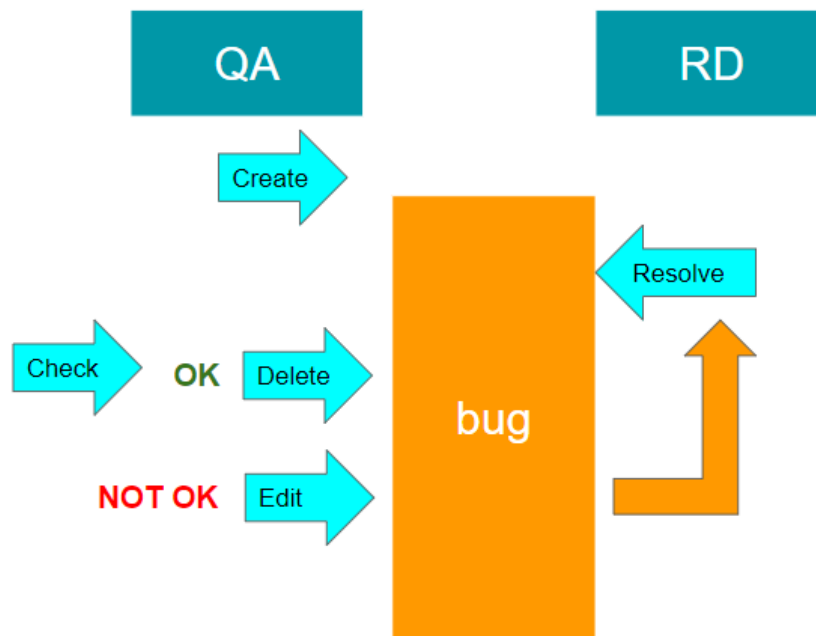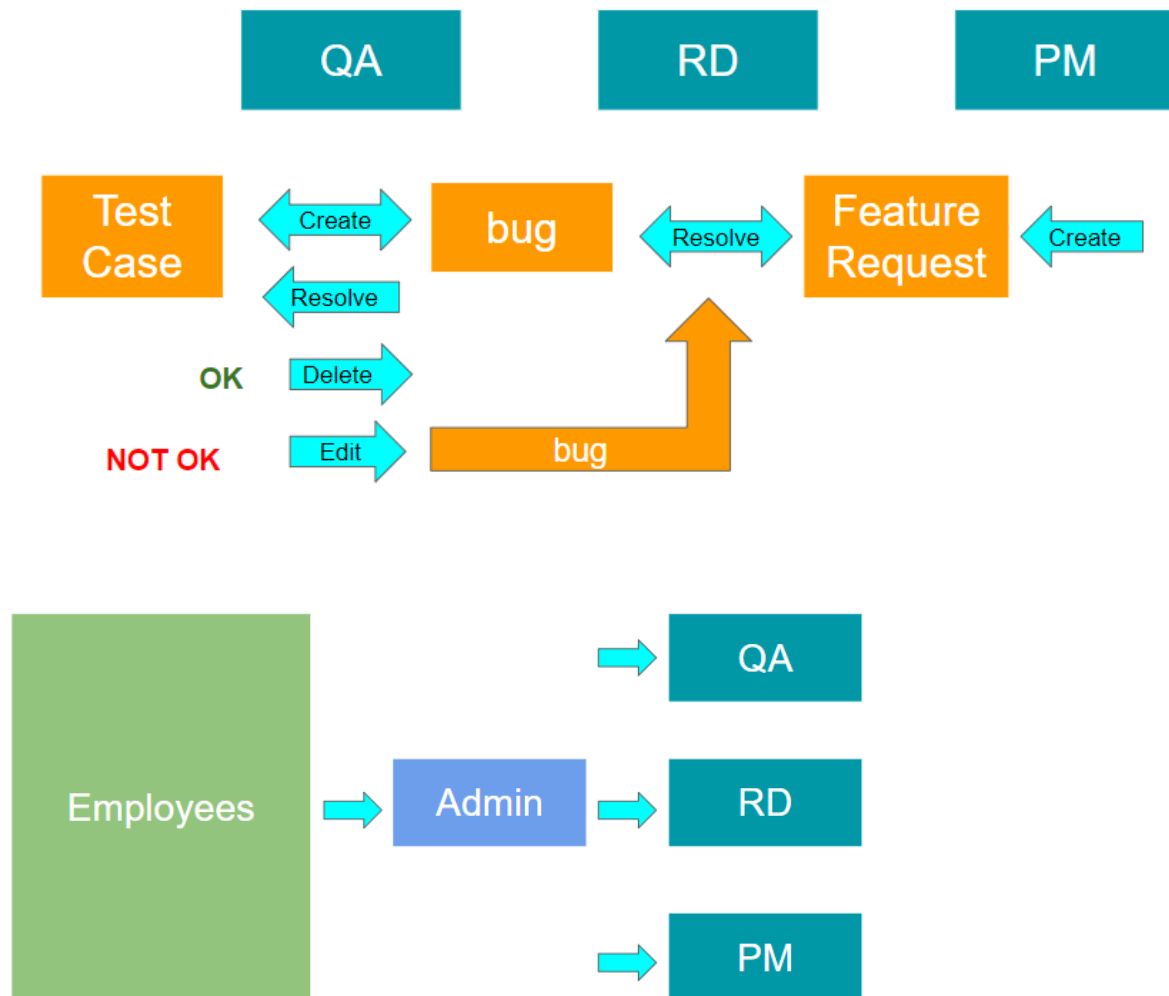# Task 1 -

Please write down all the use cases either in text or diagram you can think of for Phase I and Phase II requirements separately.

Phase I

Phase II

QA	RD	PM

Test Case ← Create → bug ← Resolve → Feature Request ← Create

Test Case ← Resolve

OK → Delete →

NOT OK → Edit → bug

Employees → Admin → QA

Admin → RD

Admin → PM

Task 2 -

Please implement the A.
Phase I Requirement by .NET Core MVC/Java Spring MVC/PHP Laravel 8/ Python Django.
For front-end, you can use any framework you like, but we prefer Vue XDD.

Please See codes in the MVC .NET Project next to this Doc. And you can try it on like :

Screenshot 1:

Please select your Identity:
- ● QA
- ○ RD

Add Bug >

Current Bugs :

Resolve	Delete	Summary	Description
☐	Delete	Summary1	Description1
☐	Delete	Summary2	Description2
☐	Delete	Summary3	Description3

Screenshot 2:

Tool | W

localhost:44346 顯示
Only QA can Delete Bugs

確定

Plea

- ○ QA
- ● RD

Add Bug >

Current Bugs :

Resolve	Delete	Summary	Description
☐	Delete	Summary1	Description1
☐	Delete	Summary2	Description2
☐	Delete	Summary3	Description3

Screenshot 3:

Tool | W

localhost:44346 顯示
Only QA can Edit Bugs

確定

Please se

- ○ QA
- ● RD

Add Bug >

Current Bugs :

Resolve	Delete	Summary	Description
☐	Delete	Sum	Description1
☐	Delete	Summary3	Description3

Screenshot 4:

Tool | W

localhost:44346 顯示
You have to be RD to resolve a BUG.

確定

Please se

- ● QA
- ○ RD

Add Bug >

Current Bugs :

Resolve	Delete	Summary	Description
☐	Delete	Summary1	Description1
☐	Delete	Summary2	Description2

Task 3 -

Think of yourself as an architect. How will you design this system,
 please write down the design document at least to include
data model, class diagram and UI mock up.

I think 'this system' means Phase II,

data model :
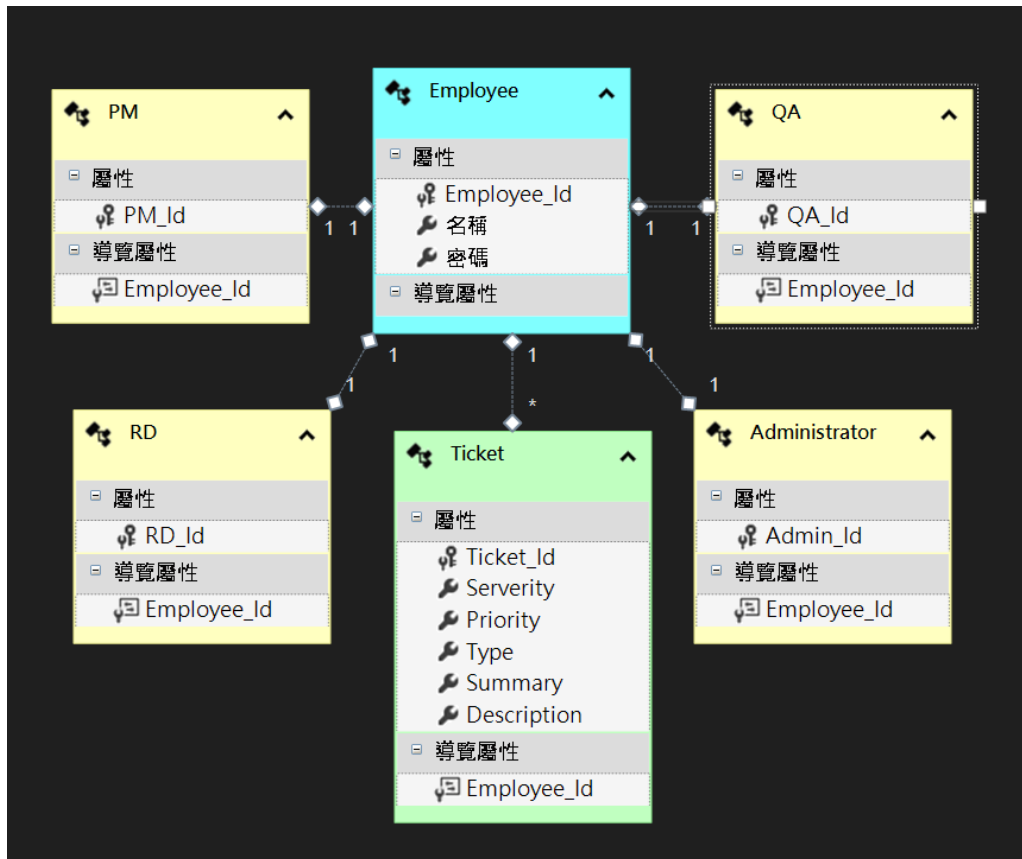
class diagram :

```
class Administrator
{
        int Admin_Id;
        int Employee_Id;
        void  Create_QA(int Employee_Id){};
        void  Create_RD(int Employee_Id){};
        void  Create_PM(int Employee_Id){};
}

class QA
{
        int QA_Id;
        int Employee_Id;
```

```
        void Create_TestCase(string summary, string desc){};
        void Create_Bug(string summary, string desc){};
        void Delete_Bug(int Tickey_Id){};
        void Edite_Bug(int Tickey_Id ,string new_summary, string new_desc){};
}

class RD
{
        int RD_Id;
        int Employee_Id;
        void Resolve_Bug(int Tickey_Id){};
        void Resolve_FeatureRequest(int Tickey_Id){};
}

class PM
{
        int PM_Id;
        int Employee_Id;
        void Create_FeatureRequest(string summary, string desc){};
}
```
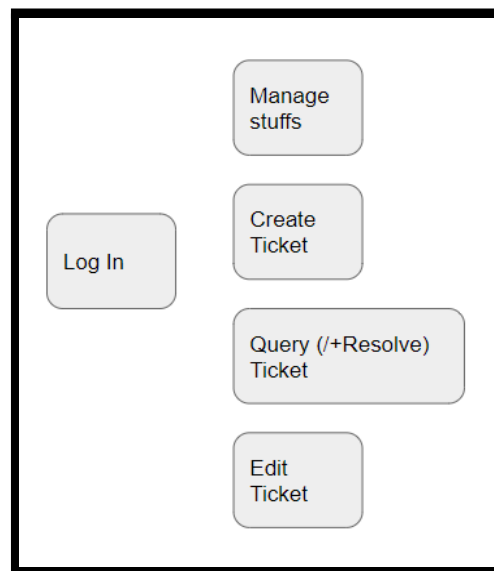
UI mock up :
Planning on having this 5 pages, can only enter from Login page.

Task 4 -

If we are going to open the system for 3rd party to use,
 can you please **design the Web API(Json format) and api document**?

Format_1
```
{
 "Identity":"Admin",
 "Name":"Kate",
 "Password":"hello",

 "Action":"Add_RD",
 "Employee_Id":"001",
}
```
Format_2
```
{
 "Identity":"RD",
 "Name":"Kate",
 "Password":"hello",

 "Action":"Resolve",
 "Ticket_Id":"0002"
}
```
Format_3
```
{
 "Identity":"PM",
 "Name":"Kate",
 "Password":"hello",

 "Action":"Add_FeatureRequest",
 "Ticket_Id":"0002",

 "Summary":"summary1xxxx",
 "Desc":"descriptionxxxxx",
}
```

Document :

接收Format1 (for 人事操作)
- 檢查帳密是否可以登入 Admin 身份
- 檢查Action是否在可選操作的範圍內
- 以Employee_Id為輸入, 進行該Action

接收Format2 (for Ticket Delete 和 Resolve)
- 檢查帳密是否可以登入 (RD/QA) 身份
- 檢查Action是否在該 身份可選的操作範圍內 (如：RD只能Resolve, QA只能Delete)
- 以Ticket_Id為輸入, 進行該Action

接收Format3 (for Ticket Create 和 Edit)
- 檢查帳密是否可以登入 (QA/PM) 身份
- 檢查Action是否在 該 身份的可選操作範圍內

- 如果Action為 Edit，要檢查 Ticket_Id 不能為空。
- 以 Summary, Desc 為輸入，進行該Action