Ryan Bishop
Aamir Ali
Ruchi Gupta
Akhyar Zaman
Pooja Aggarwal
Christian Angel

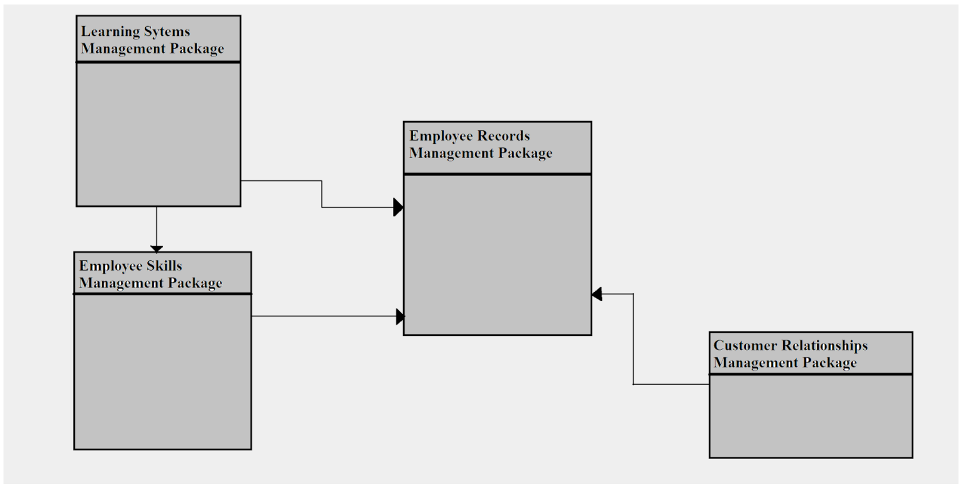
Group #1 - Milestone 3 (Design Phase)

Table of Contents:

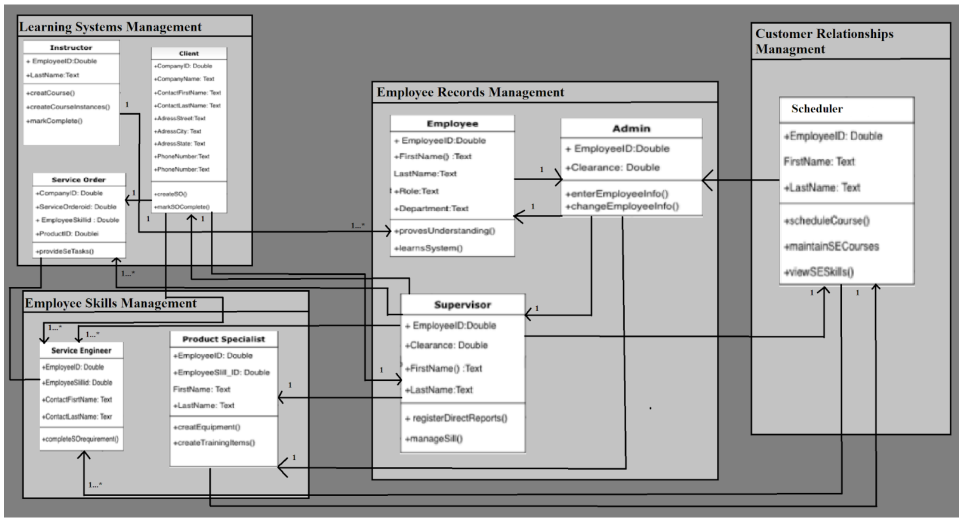
Executive Summary

Initially we outline our package diagram which shows the dependencies between our packages, allowing developers to observe various views of the system. Our ‘Design and Acquisition strategy defines our high-level solutions for our business need, addressing in-house experience, developing project skills, time frame and project management. Our acquisition strategy uses an alternative matrix to determine the optimal application language to utilize for our system. The ‘Design Criteria and Design Pattern’ section details our interrelationships among the classes, methods, and classes in order to handle coupling, cohesion, and consonance in the system. Following this section is the ‘Constraints and Contracts’ portion where the CRC cards that detail the object oriented design can be found. The constraints are reflected in the contracts, which illustrate the messages passing between a given two objects in our system. The ‘Method Specifications’ portion will aid our programmers in developing the software, by outlining methods for each system. We then employ normalization to create an optimized mapping diagram for the system, showing interrelations between methods. The ‘User Interface Design’ portion includes the use real case descriptions, which divulge information on the components of the system as well as the flow of information in the system. This section includes use scenarios that allow the system developers to design the system to operate for all the needed appropriate functions. In the ‘Web Navigation Design’, the interface for our remote online system for each component can be found. Subsequently, the storyboards portion includes the links from our menus, classes, and class info. This portion also includes our User interface with examples of various use case screens, from the users end. Finally, the ‘Physical Architecture layer’ details our selection of a Client-Server model, provides deployment diagrams to outline links between hardware, and includes network diagrams to show high and low level networking between Agilent’s servers. This also includes hardware & software specs, as well as nonfunctional requirements, technical, system integration, portability, and maintainability requirements.

Packages and Package Diagrams


Learning Sytems Management Package
Employee Records Management Package
Employee Skills Management Package
Customer Relationships Management Package

(Overview of the Package Diagram)



(Package Diagram)

<u>**Design and Acquisition Strategy**</u>

Design strategy

In order to decide if our project team should use the custom development, purchased package software, or outsourcing design strategy we must observe and analyze the five different characteristics of each strategy. Doing so would allow us to understand what each provides to the system we intend to develop.

Business Need: In terms of business needs, our system has common solutions when it comes to the scheduling and management of information since there are existing systems that allow us to perform these function that would sufficient for the field supervisors to handle this type of task. On the other hand, a custom system to improve the facilitation of training for service engineers and user interface for clients could be a better alternative because if we program the system to perform these task based on our business needs then it would definitely accomplish our expectations.

In-house Experience: Since our business is presumed to have service engineers that have experience performing all the functional and technical needs of the system it will be easier to build a custom application that can schedule and manage tasks at our business level. We also understand our current user interface so we can manipulate to perform its task more effectively.

Project skills: Project skills refer to the skills that are applied during projects that involve either technical skills like coding in Java or SQL or functional skills like security protocols that are viable based on the company strategy. The business has the technical skills necessary to develop the system application and a custom system would be beneficial in this case because it would allow our engineers skills to develop and improve.

Project Management: If we were to use a package system and outsourcing alternatives the project team would not encounter internal obstacles because the external parties have their own objectives and priorities surrounding their methodologies of the packages. Luckily we have an expert project sponsor, Thomas Tong, that can guide us through the various obstacles we my encounter such as staffing hold ups that may occur or and how to contain the demands of the business users.

Time Frame: Our current system for our business is current and functional but it need to be improved to allow our users to perform their task more efficiently. That being said time is not a constraint for the development of the new system since they could continue to use the old system so there is no reduction in the business process. The downside of building a custom system if some time constraints come up during the process we have to test the system by parts rather than the entire system which may lead to some errors or an entire system failure. There is a way to get around the time constraints if they are an issue for custom system development and that is the use of timboxing which allocates a fixed time period a planned activity is to be completed by.

After a careful inspection of the criteria above our project team has decided to use the custom development design strategy for our system. This is due to the fact that we are trying to create a system that will serve our company specific needs that include a better a scheduling and information management for the field supervisors. Furthermore, we have the personnel with the necessary skills in the technical, functional, and management aspects needed to build a custom system that would also expand and improve their skills from creating a system designed for the company.

Acquisition strategy

Using an alternative matrix the project team will organize the pros and cons of the design alternatives so that we may determine the best solution in the end. We will be comparing a custom system that the company can build using one of the following alternatives Java, SQL, or Visual Basic custom application.

Analyst # 1 Alternative Matrix

Evaluation Criteria	Relative Importance (Weight)	Alternative 1: Custom Application Using Java	Score (1-5)*	Weighted Score (WS)	Alternative 2 : Custom Application Using SQL	Score (1-5)*	Weighted Score (WS)	Alternative 3 : Custom Application Using Visual Basic	Score (1-5)*	WS
Technical Issues:		Look at Java section below for supporting information			Look at the SQL section below for supporting information			Look at the Visual Basic section below for supporting information		
Hardware Installation	15		3	45		3	45		3	45

Software Installation	25		5	125		5	125		5	125
Covert Data	5		2	10		3	15		3	15
Economic Issues:										
Total Cost	10		5	50		5	50		5	50
Total Benefits	10		4	40		3	30		4	40
Organizational Issues:										
System Adoption	20		3	60		3	60		5	100
Training	10		3	30		4	40		5	50
Management Policies	5		3	15		3	15		3	15
Total	100			375			380			440

* This denotes how well the alternative meets the criteria. 1 = poor fit; 5 = perfect fit

Analyst # 2 Alternative Matrix

Evaluation Criteria	Relative Importance (Weight)	Alternative 1: Custom Application Using Java	Score (1-5)*	Weighted Score (WS)	Alternative 2 : Custom Application Using SQL	Score (1-5)*	Weighted Score (WS)	Alternative 3 : Custom Application Using Visual Basic	Score (1-5)*	WS
Technical Issues:		Look at Java section below for supporting			Look at the SQL section below for			Look at the Visual Basic section below		

Hardware Installation	15	information	2	30	supporting information	2	30	for supporting information	3	45
Software Installation	25		5	125		5	125		5	125
Covert Data	5		3	15		2	10		3	15
Economic Issues:										
Total Cost	10		5	50		5	50		5	50
Total Benefits	10		4	40		3	30		4	40
Organizational Issues:										
System Adoption	20		2	40		3	60		5	100
Training	10		3	30		4	40		5	50
Management Policies	5		2	10		3	15		4	20
Total	100			340			360			445

* This denotes how well the alternative meets the criteria. 1 = poor fit; 5 = perfect fit

Custom application using Java: The Java platform is know to be one of the most popular programming languages for about two decades and is popular for being object oriented, concurrent and class based. The language is know to be able to handle large amounts of data that can help organization better manage their tasks at an efficient manner. Some of the benefits of using this language as the custom application platform is that the learning curve is extremely short learning curve since it is easy to write, compile, debug than other major programming languages. In addition, it is a object oriented programming language that allows for the creation of modular programs and allows the reuse of code that keeps the system extensible as well as flexible but bes of all it is cost efficient since it is free. Some of the downsides of using Java for custom applications is that the memory management is excessive with Java since objects that are no longer used are cleared to make space for new objects. Furthermore, the lack of templeants can limit the ability for Java to create high quality data structures.

Custom application using SQL: The SQL languages uses machine learning functions that provide organizations with a shortage of talent to carry our machine learning plans due to its low learning curve. With an SQL database the company will have a more distributed system that would allow the programmers to use more cores and have better parallelism. This would enable more hardware to address a single query all at once with the best performance as possible. The distributed system could allow for scalability in terms of using several servers that increases the performance than using a single node system. Moreover, code generation helps optimize the queries and custom functions in the database by converting the requests into machine code allowing them to run faster. Unfortunately, using the the SQL language makes the system less flexible which can complicate the interface that makes it difficult for some user to access it.

Custom application using Visual Basic: Visual basic is a programming language that was developed by Microsoft that was based on the BASIC computer language which fot expanded on to allow for easy programming of windows applications. The programming language has a very low level learning curve since the syntax is generally more straightforward than other programming language. Some advantages to using VB is that it has been highly optimized to support rapid application development that can easily develop graphical user interfaces and connects them to handler functions provided by the application. The language is built around the .NET environment used by all Microsoft visual languages so there is very little that can be done in VB that can be done in other programming language. On the other hand, one disadvantage is that visual basic is a proprietary programming language that is written by Microsoft, so programs that are written in this language cannot, easily, be transferred to other operating systems.

Using our analysts' alternative matrix, the project team was able to derive the three different application languages that can be used to build the company's custom application. Through this chart we learned the advantages and disadvantages of using a Java, SQL, and Visual Basic as application platforms and how well they satisfy the different technical, economic and organizational criteria. Even though they were scored differently in both matrices, the custom application Visual Basic alternative had the highest weighted score with 440 in the first table and 445 in the second table. Assuming that the company has a windows based application system then this would be the best option for us to build our application because it requires minimal training that would allow our engineers to build a user interface quickly and efficiently.

Design Criteria And Design Pattern

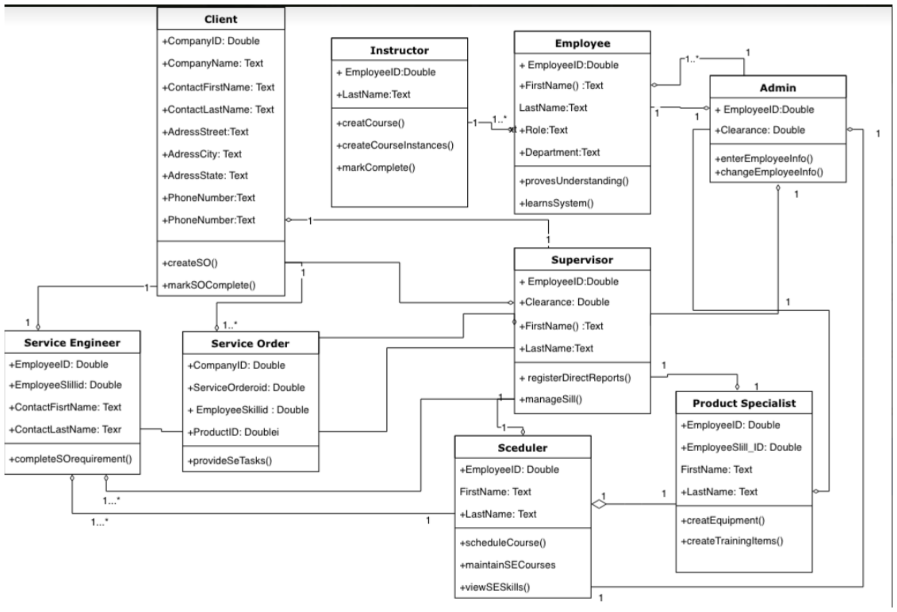
a) Coupling- how interrelated the system will be

i) Type: **Interaction Data Coupling**

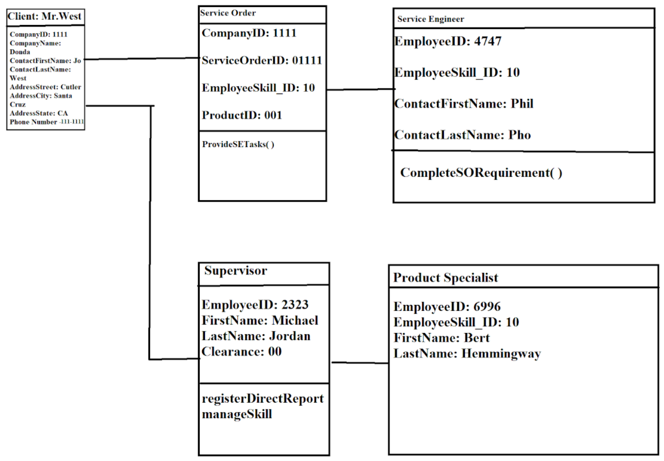
- The calling method passes a variable to the called method. If the variable is composite (i.e., an object), the entire object is used by the called method to perform its function

ii) Relationships for our system

■ Class Diagram-showing interdepencies

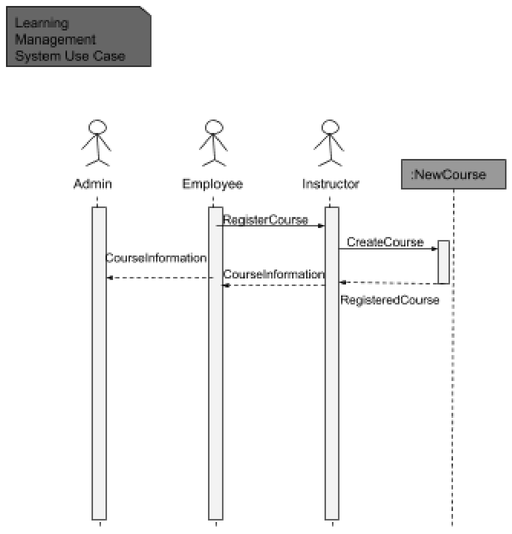


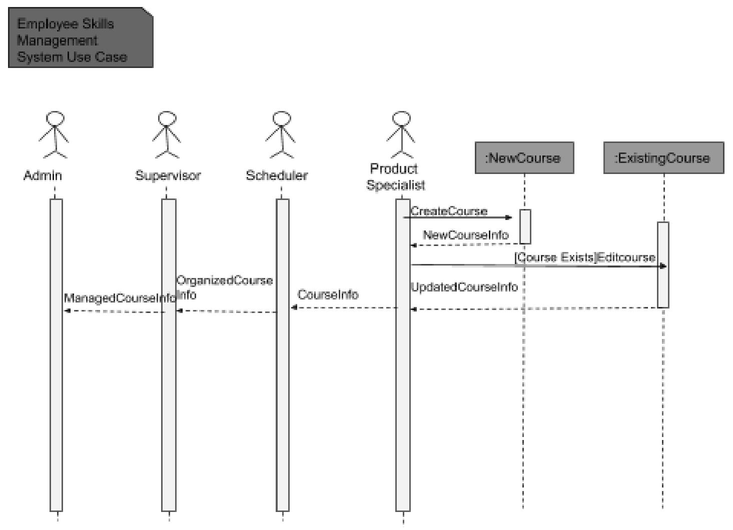
2) Object Diagrams-showing inheritance between objects

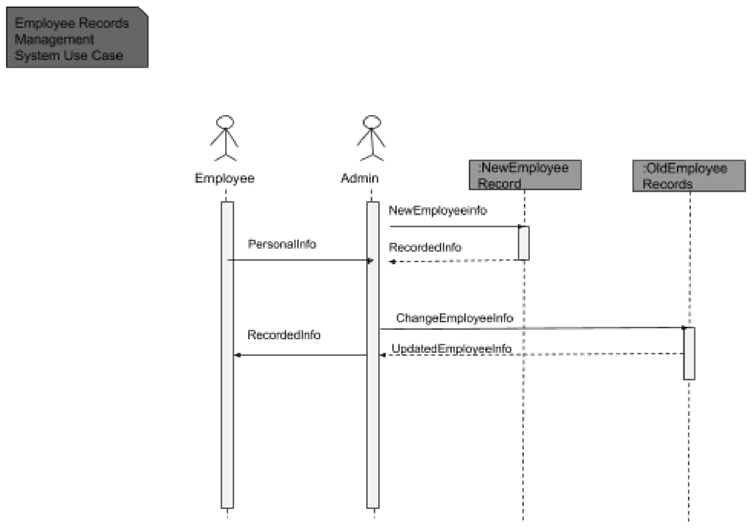


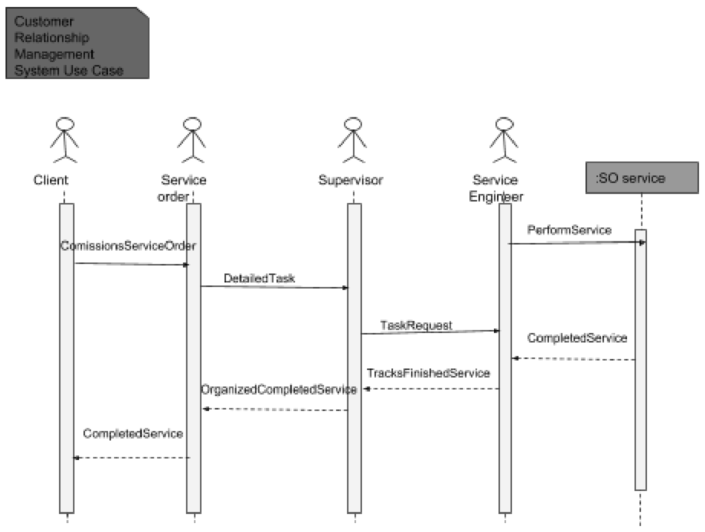
2) Behavioral Models

a.) Sequence Diagrams

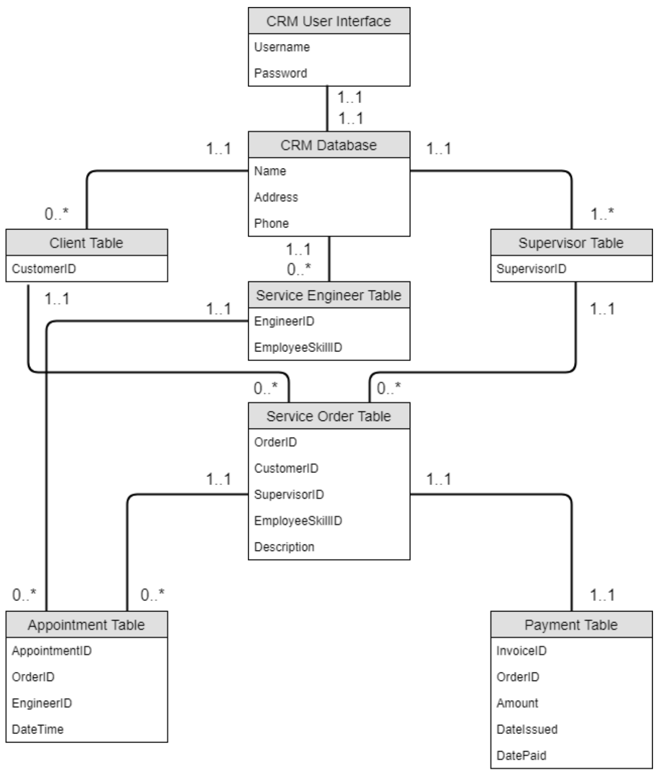



Employee Skills Management System Use Case
Admin
Supervisor
Scheduler
Product Specialist
:NewCourse
:ExistingCourse
CreateCourse
NewCourseInfo
[Course Exists]Editcourse
UpdatedCourseInfo
CourseInfo
OrganizedCourse Info
ManagedCourseInfo


Employee Records Management System Use Case
Employee
Admin
:NewEmployee Record
:OldEmployee Records
NewEmployeeinfo
PersonalInfo
RecordedInfo
ChangeEmployeeInfo
RecordedInfo
UpdatedEmployeeInfo



b) Cohesion- level of cohesion among attributes and methods of a class
   i) Method Cohesion: **Functional**
      - method diagram shows interrelations



ii) Class Cohesion**: Partially Sequential**
    - Please see class diagram in coupling section above

iii) Generalization/Specialization Cohesion: **Mixed-Role Cohesion**

c) Connascence- Proactive checklist for our system in order to minimize consonance of coupling and maximize for cohesion

   i) Consider encapsulation of any elements in the system

   ii) Name
      - If a method refers to an attribute, it is tied to the name of the attribute.
      - If the attribute's name changes, the content of the method will have to change.

   iii) Type or Class
      - If a class has an attribute of type A, it is tied to the type of the attribute.
      - If the type of the attribute changes, the attribute declaration will have to change.

   iv) Convention
      - A class has an attribute in which a range of values has a semantic meaning
      - Confirm range ahead of time else every method that used the attribute will have to be modified.

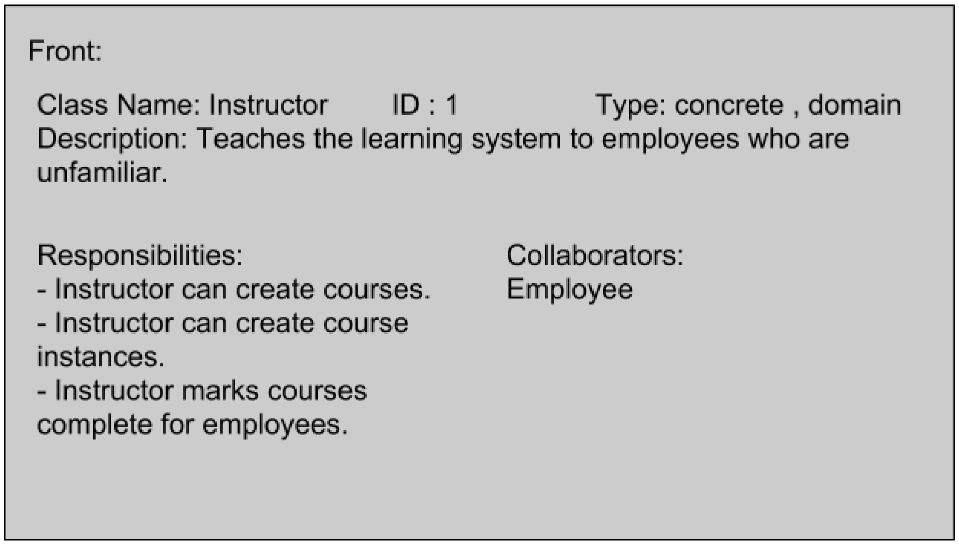
   v) Algorithm
      - Two different methods of a class are dependent on the same algorithm to execute correctly.
      - If the underlying algorithm changes, then the insert and find methods would also have to change.

   vi) Position
      - The order of the code in a method or the order of the arguments to a method is critical for the method to execute correctly.
      - If either is wrong, then the method will, at least, not function correctly.

Constraints and Contracts

The following are the expansion of the CRC cards to include the constraints that capture the object oriented design.

Learning Management System



Front:

Class Name: Instructor ID : 1 Type: concrete , domain

Description: Teaches the learning system to employees who are unfamiliar.

Responsibilities:	Collaborators:
- Instructor can create courses.	Employee
- Instructor can create course instances.	
- Instructor marks courses complete for employees.	

Back

Attributes:
EmployeeID(double): { EmployeeID = Employee.GetEmployeeID()}
FirstName(text): { FirstName = Employee.GetFirstName()}
LastName(text): { LastName = Employee.GetLastName()}
Role(text): { Role = Employee.GetRole)}
Department(text): { Department= Employee.GetDepartment()}
Create Course: (in anCourse: Course)
Create CourseInstance: (in anCourseInstance: CourseInstance)

Relationships:
Generalization:
Aggregation: Instructor
Other Associations:

Employee Skills Management

Front

Class Name:Supervisor Type:concrete,domain
ID: 1

Description: Manage Employees' smills.

Responsibilities:
- Register direct reports for courses.
- Manage skill and skill level.

Collaborators:
Admin
Scheduler
Product Specialist

Back

Attributes:
EmployeeID(double);
Clearance(double):
FirstName(text):
LastName(text):
RegisterReports: (in anRegisterReports: RegisterReports)

Relationships:
Generalizations:
Aggregation: Admin
Scheduler
Product Specialist
Other Association:

Front:

Class Name: Admin Type: concrete, domain
ID: 2

Description: Manage employees' information and skills.

Responsibilities:	Collaborators:
- Enter Employee's Information.	Supervisor
	Scheduler
- Change Employee's Information.	Product Specialist

Back

Attributes:
EmployeeID(double): { EmployeeID = Employee.GetEmployeeID }
Clearance(double):
enterEmployeeInfo(): (in anEmployeeInfo(): EmployeeInfo): void
ChangeEmployeeInfo(): (in anEmployeeInfo: EmployeeInfo)
Relationships:
Generalizations:
Aggregation: Supervisor
Scheduler
Product Specialist
Other Associations:

Front

Class Name:Product Specialist ID: 3
Type: concrete, domain
Description: Creates or edits training resources.

Responsibilities:	Collaborators:
- Create equipment.	Supervisor
- Create training items.	Admin
- Help edit the course.	Scheduler

Back

Attributes:
EmployeeID(double):{ EmployeeID = Employee.GetEmployeeID }
EmployeeSkill_ID(double):
FirstName(text): { FirstName = Employee.GetFirstName }
LastName(text):{ LastName = Employee.GetLastName }
CreateEquipment(): (in anEquipment = Equipment)
CreateTrainingItems(): (in anTrainingItems = TrainingItems)

Relationships:
Generalizations:
Aggregation:Supervisor
Admin
Scheduler
Other Associations:

Front

Class Name: Scheduler ID: 4
Type: concrete, domain
Description: View any SE’s skills and courses.

Responsibilities:	Collaborators:
- Schedule courses for employees.	Supervisor
- Maintain SE’s courses.	Admin
- View SE’s skills.	Product Specialist

Back

Attributes:
EmployeeID(double):{ EmployeeID = Employee.GetEmployeeID }
FirstName(text): { FirstName = Employee.GetFirstName }
LastName(text): { LastName = Employee.Get.LastName }
ScheduleCourse(): (in anScheduleCourse = ScheduleCourse)

Relationships:
Generalizations:
Aggregation: Supervisor
Admin
Product Specialist
Other Associations:

Customer Relationship Management

Client

Front

Class Name: Client **ID:** 1 **Type:** Concrete, domain
Description: commissions engineers from Agilent

Responsibilities	Collaborators
Create SO's	Service Engineer
Mark SO complete	Supervisor
	Service Order

Back

Attributes:

CompanyID(double):	AddressStreet(text):
CompanyName(text):	AddressCity(text):
ContactFirstName(text):	AddressState(text):
ContactLastName(text):	PhoneNumber(text):
CreateSO: (in anSO = SO)	

Relationships

Generalization:	
Aggregation:	Service Engineer
	Supervisor
Other Associations:	Service Order

Service Order

Front

Class Name: Service Order **ID:** 2 **Type:** Concrete, domain
Description: details task required by client

Responsibilities	**Collaborators**
Provide task/skills required of SE	Supervisor
Provide client's info	Service Engineer(SE)
	Client

Back
Attributes:
ServiceOrderid(double): Status(text):
CustomerID(double): { CustomerID = Client.GetClient }
EmployeeSkillid(double):
Productid(double):
CreateTask: (in anTask= Task)
Relationships
Generalization:
Aggregation:
Other Associations: Supervisor
Service Engineer Client

Service Engineer

Front

Class Name: Service Engineer **ID:** 4 **Type:** Concrete, domain
Description: fulfills service requested by SO's

Responsibilities	**Collaborators**
Complete service required by SO when SP notifies	Supervisor Service Order

Back

Attributes:
EmployeeID(double): { EmployeeID = Employee.GetEmployeeID }
EmployeeSkillid(double): { EmployeeSkillid= Service Order.GetEmployeeSkillid}
ContactFirstName(text): { ContractFirstName = Client.GetContractFirstName}
ContactLastName(text):{ ContractLastName = Client.GetContractLastName}
PhoneNumber(text): { PhoneNumber = Client.GetPhoneNumber}
Relationships
Generalization:
Aggregation: Client
Supervisor
Other Associations: Service Order

Contracts

Now that we have added the constraints to the CRC cards that also reflect on the class diagram we can document the different contracts. The contracts illustrate the message passing that takes place between two objects.

Method Name: Create Course	**Class Name:** Instructor	**ID:** 1
Clients (consumers): Employee		
Associates Use Cases: Learning Management System		
Description of Responsibilities: A course is created by an instructor to teach the learning system to the employees who are unfamiliar with it.		
Arguments Received: anCourse:Course		
Type of Value Returned: void		
Pre-Conditions: not course.includes(anCourse)		
Post-Conditions: none		

Method Name: Register Reports	**Class Name:** Supervisor	**ID:** 1
Clients (consumers): Employee		
Associates Use Cases: Employee Skills Management		
Description of Responsibilities: To register direct reports for the courses that manage the skill level of each employee		

Arguments Received: anRegister:Register
Type of Value Returned: void
Pre-Conditions: not course.includes(anCourse)
Post-Conditions: none

Method Name: Enter Employee Info	**Class Name:** Admin	**ID:** 2
Clients (consumers): Employee		
Associates Use Cases: Employee Skills Management		
Description of Responsibilities: To register employee information or update an existing employees information		
Arguments Received: anEmployeeInfo:EmplyeeInfo		
Type of Value Returned: void		
Pre-Conditions: none		
Post-Conditions: none		

Method Name: Change Employee Info	**Class Name:** Admin	**ID:** 2
Clients (consumers): Employee		
Associates Use Cases: Employee Skills Management		
Description of Responsibilities: To register employee information or update an existing employees information		

Arguments Received: anEmployeeInfo:EmplyeeInfo
Type of Value Returned: void
Pre-Conditions: Employee.includes(Employee)
Post-Conditions: none

Method Name: Create Equipment	**Class Name:** Product Specialist	**ID:** 3
Clients (consumers): Employee		
Associates Use Cases: Employee Skills Management		
Description of Responsibilities: To create and edit the training resources that employees		
Arguments Received: anEquipment:Equipment		
Type of Value Returned: void		
Pre-Conditions: not course.includes(anCourse)		
Post-Conditions: none		

Method Name: Create Training Items	**Class Name:** Product Specialist	**ID:** 3
Clients (consumers): Employee		
Associates Use Cases: Employee Skills Management		
Description of Responsibilities: To create and edit the training resources that employees		
Arguments Received: anTrainingItems:TrainingItems		

Type of Value Returned: void
Pre-Conditions: not course.includes(anCourse)
Post-Conditions: none

Method Name: Schedule Courses	**Class Name:** Scheduler	**ID:** 4
Clients (consumers): Employee		
Associates Use Cases: Employee Skills Management		
Description of Responsibilities: To schedule the courses for each employee to that will test their skills		
Arguments Received: anScheduleCourse:ScheduleCourse		
Type of Value Returned: void		
Pre-Conditions: not course.includes(anCourse)		
Post-Conditions: none		

Method Name: Create Service Order	**Class Name:** Client	**ID:** 1
Clients (consumers): Service Engineers		

Associates Use Cases: Customer Relationship Management
Description of Responsibilities: To commission engineers from Agilent
Arguments Received: anOS: OS
Type of Value Returned: void
Pre-Conditions: none
Post-Conditions: none

Method Name: Create Task	**Class Name:** Service Order	**ID:** 2
Clients (consumers): Service Engineers		
Associates Use Cases: Customer Relationship Management		
Description of Responsibilities: To provide the tasks required for the Service Engineer to complete their work		
Arguments Received: anTask: Task		
Type of Value Returned: void		
Pre-Conditions: none		
Post-Conditions: none		

Method Name: Create Service Order	**Class Name:** Client	**ID:** 1
Clients (consumers): Service Engineers		

Associates Use Cases: Customer Relationship Management
Description of Responsibilities: To commission engineers from Agilent
Arguments Received: anOS: OS
Type of Value Returned: void
Pre-Conditions: none
Post-Conditions: none

Method Specifications:

The method specification helps our programmers develop the program with specific instructions to code the method provided in the method form.

For **Learning Management System**

Method Name: Create course	**Class Name:** Instructor	**ID:** 1
Contract ID: 1	**Programmer:** TBD	**Date Due:** TBD
Programming Language: Java		
Triggers/Events: when more courses are needed		

Arguments Received: **Data Types:**	**Notes:**	
Messages Sent & Arguments Passed: ClassName. MethodName:	**Data Type:**	**Notes:**
Instructor.AvailSlot(List)	List	Lists time slots for classes
Instructor.Create		Create the course
Arguments Returned: **Data Type:**	**Notes:**	
Algorithm Specification: see below this form		
Misc. Notes:		

Algorithm Specification for Create Course:

```
Request list of available time slots
    If Instructor picks available time slot
        Execute creation of course
    Else
        Tell the instructor there is a time conflict


End
```

For **Employee Skills Management**

Method Name: Reserve seats	**Class Name:** Supervisor	**ID:** 1
Contract ID: 1	**Programmer:** TBD	**Date Due:** TBD
Programming Language: Java		
Triggers/Events: When direct report lets the supervisor know an SE needs to be enrolled in a class		
Arguments Received: **Data Types:**	**Notes:**	
Messages Sent & Arguments Passed: ClassName. MethodName:	**Data Type:**	**Notes:**
Supervisor.AvailSlot	List	Lists time slots for classes
Supervisor.Reserve		Reserve seats for the Course
Arguments Returned: **Data Type:**	**Notes:**	

Product Specialist	If the supervisor reserves seats for the SEs, The product specialist is returned.
Algorithm Specification: see below this form	
Misc. Notes:	

Algorithm Specification for Reserve seats:

```
Request list of available time slots
    If Supervisor picks available time slot
        Execute Reservation of seat in course
    Else
        Tell the Supervisor there is no space available


End
Return Product Specialist
```

Method Name: Create Training Items	**Class Name:** Product Specialist	**ID:** 2
Contract ID: 2	**Programmer:** TBD	**Date Due:** TBD
Programming Language: Java		

Triggers/Events: when there is a need for equipment		
Arguments Received: **Data Types:**	**Notes:**	
Messages Sent & Arguments Passed: ClassName. MethodName:	**Data Type:**	**Notes:**
ProductSpecialist.Survey	Survey	Shows what equipment is needed
ProductSpecialist.Create		Create the equipment
ProductSpecialist.Edit		Edit the Equipment
Arguments Returned: **Data Type:**	**Notes:**	
Algorithm Specification: see below this form		
Misc. Notes:		

Algorithm Specification for Create Training Equipment:

```
Request Survey to be filled out
    If Product Specialist Chooses to create equipment
        Execute creation of equipment
    Else
        Tell the Product Specialist it was unable to do so

End
```

For **Customer Relationship Management:**

Method Name: Create Service Order	**Class Name:** Client	**ID:** 1
Contract ID:	**Programmer:** TBD	**Date Due:** TBD
Programming Language: Java		
Triggers/Events: when Client submits a request for a service order		
Arguments Received: **Data Types:**	**Notes:**	

Messages Sent & Arguments Passed: ClassName. MethodName:	**Data Type:**	**Notes:**
Client.Showopenorders	List	Shows what service orders are open
Client.Create		Create Service Order
Arguments Returned: **Data Type:**	**Notes:**	
Create Task	This is done only when a service order has been created	
Algorithm Specification: see below this form		
Misc. Notes:		

Algorithm Specification for Create Service Order:

```
Create Service Order
       If Open
          SEs will check on order
         Else
            Service Order was Resolved


End
Return Create Task
```

Method Name: Create Task	**Class Name:** Service Order	**ID:** 2
Contract ID: 2	**Programmer:** TBD	**Date Due:** TBD
Programming Language: Java		
Triggers/Events: when there is a service order open		
Arguments Received: **Data Types:**	**Notes:**	
Messages Sent & Arguments Passed: ClassName. MethodName:	**Data Type:**	**Notes:**
SE.OpenService	list	Shows the open service orders
SE.Createtask		Create the task needed for the order
Arguments Returned: **Data Type:**	**Notes:**	
Algorithm Specification: see below this form		
Misc. Notes:		

Algorithm Specification for create task:

```
Bring up list of open service orders
    If SE Chooses to create Task
        Execute creation of Task
    Else
        Tell the SE it was unable to do so


End
```

For **Employee Records Management System:**

Method Name: Change Employee Information	**Class Name:** Admin	**ID:** 1
Contract ID: 1	**Programmer:** TBD	**Date Due:** TBD
Programming Language: Java		
Triggers/Events: When the status of an employee changes		
Arguments Received: **Data Types:**	**Notes:**	

Messages Sent & Arguments Passed: ClassName. MethodName:	**Data Type:**	**Notes:**
Admin.edit		Edit employee information
Admin.Create		Create employee information
Admin.Delete		Delete employee information
Arguments Returned: **Data Type:**	**Notes:**	
Algorithm Specification: see below this form		
Misc. Notes:		

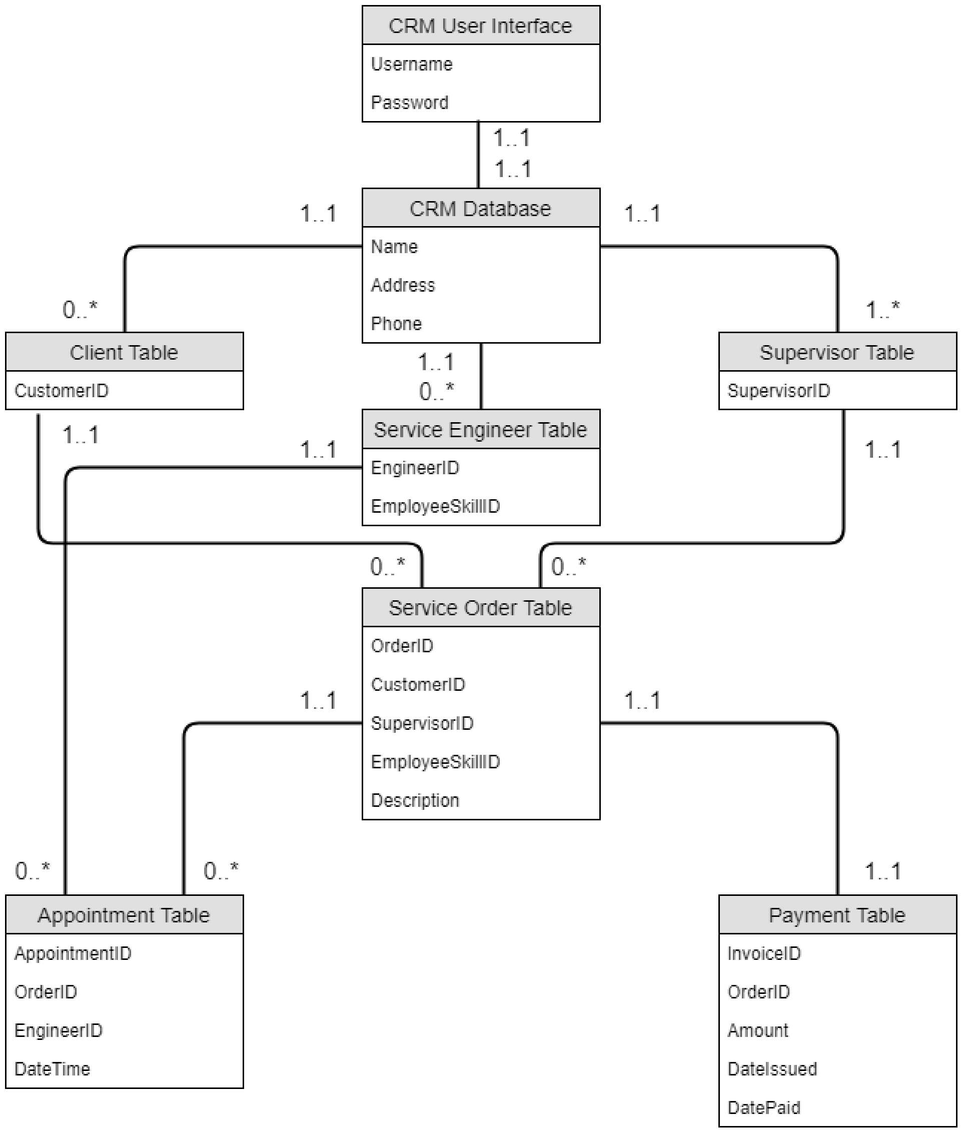
Algorithm Specification for Change employee information:

```
Request to create employee information
    If admin Chooses to create information
        Execute creation of employee nformation
    Else
        Tell the admin it was unable to do so
Request to edit employee information
    If admin chooses to edit
        Excute edit of information
    Else
        Cancel all changes
Request to delete employee information
    If Admin chooses to delete information
        Delete the information
        Else
          Cancel


End
```

Optimizing RDBMS

Using normalization, here is the optimized mapping diagram for the system:


CRM User Interface
Username
Password
1..1
1..1
CRM Database
Name
Address
Phone
1..1
1..1
0..*
1..*
Client Table
CustomerID
Supervisor Table
SupervisorID
1..1
0..*
Service Engineer Table
EngineerID
EmployeeSkillID
1..1
1..1
1..1
0..*
0..*
Service Order Table
OrderID
CustomerID
SupervisorID
EmployeeSkillID
Description
1..1
1..1
0..*
0..*
1..1
Appointment Table
AppointmentID
OrderID
EngineerID
DateTime
Payment Table
InvoiceID
OrderID
Amount
DateIssued
DatePaid

<u>User Interface Design</u>

Use Real Case Description

Use Case Name: Learning Management System	ID: 1	Importance Level: High
Primary Actor: Instructor	Use Case Type: Detail, Real	
Stakeholders and Interests: Instructor wants to ● create courses ● create course instances Employee wants to ● register to courses ● Instructor (or system) marks courses complete for employee		
Trigger: Instructor creates class. Type: Internal		
Relationships: Association: Instructor, Employee, Employee Records Management Include: Extend: Generalization:		
Normal Flow of Events: 1. Instructor starts up system. 2. System provides Instructor with Main menu for the System. 3. System asks Instructor if he/she wants to add a class a. If Instructor wants to add employee, he/she clicks on Add Class Link and executes **S-1** 4. If Employee wants to register for classes he/she starts up system 5. System provided Employee with Main menu 6. Employee selects Register Tab and executes to **S-2** 7. Employee is returned to main menu 8. System returns Admin to Main Menu of System		
SubFlows: **S-1:** New Class 1. System asks Instructor for relevant information (class instances) 2. Instructor enters relevant information 3. Admin submits information to the System **S-2:** Enroll in Class: 1. System asks for search information 2. Employee enters information about class 3. Employee submits information to the System		

a. If system finds list of classes that meet the search information, system presents available classes 4. Employee selects the class he/she wants into shopping cart 5. Employee selects Enroll link. a. If Employee fits class requirements, he/she is Enrolled b. Else Employee is enrolled in pre requisite classes 6. Employee attends enrolled class
Alternate/Exceptional Flows: System produces an Error Message

Use Case Name: Employee Skills Management	ID: 2	Importance Level: High
Primary Actor: Supervisor (SP)	Use Case Type: Detail, Real	
Stakeholders and Interests: SP wants to: ● register direct reports for courses ● manage skill and skill level of direct reports Product Specialist (PS) wants to: ● create/edit equipment/course/training items association Scheduler wants to: ● view any SE's skills and courses		
Brief Description: This use case describes how supervisors can manage employees' skills.		
Trigger: Employee completes new skill. Type: External		
Relationships: Association: Supervisor (SP), Admin, Schedulers, Product Specialist, Learning System Management, Employee Records Management, CRM Include: Extend: Generalization:		
Normal Flow of Events: 1. SP starts up system 2. SP is provided with main menu 3. The system asks if SP wants to direct reports or manage reports a. If SP wants to direct reports, he/she clicks on Direct Reports Link and executes **S-1**		

If SP wants to manage reports, he/she clicks on Manage Reports and executes **S-2**

4. System returns SP to Main Menu of System

SubFlows:

S-1: Direct reports

1. System asks for direct reports
2. SP enters relevant information to register reports
3. SP submits information to the System

S-2: Manage Reports

1. System asks for relevant information
2. SP enters Employee
3. System brings up employee and skill level
4. SP manages skill level and reports
5. SP submits information

S-3: New Items

1. System asks for relevant information to create new items
2. PS enters relevant information to create items
3. PS submits information to the System

S-4: Edit Items

1. System asks for relevant information to edit new item
2. PS enters relevant information to edit items
3. PS submits information to the System

S-5: View SE

1. System asks for relevant information to create new item
2. PS enters relevant information to create items
3. PS submits information to the System

S-6: View SE skills/Courses

1. System asks for relevant search information to find employee
2. Scheduler enters employee to search
3. System provides Scheduler with Employee information

Alternate/Exceptional Flows:

Alt- 1: PS flow

1. PS starts up system
2. PS is provided with main menu
3. The system asks if PS wants to create/edit equipment/course/training items association
   a. If PS wants to Create items he/she clicks on Create Items Link and executes to **S-3**
   b. If PS wants to edited equipment he/she clicks on Edit Item Link and executes to **S-4**
   c. If PS wants to Create Course/training items he/she clicks on Create Course/Training Items Link and executes to **S-5**
4. System returns PS to Main Menu of System

Alt -2: Scheduler Flow:

1. Scheduler starts up system
2. Scedularis provided with main menu
3. The system asks if Scheduler wants to view SE skills and courses
   a. If Scheduler wants to view SE skills he/she clicks on SE Skills and executes to **S-5**

b. If Scheduler wants to view SE skills he/she clicks on SE Skills and executes to **S-5** 4. System returns Scheduler to Main Menu of System **Alt-3:** 1. The System produces an Error Message.

Use Case Name: Employee Records Management	ID: 3	Importance Level: High
Primary Actor: Admin	Use Case Type: Detail, Real	
Stakeholders and Interests: Admin wants to record employee entry, change employee status, and retain employee records.		
Brief Description: This use case describes how employee's records are automatically updated when changes to other use cases occur.		
Trigger: Employee records change in a table schema. Type: Internal		
Relationships: Association: Admin, Learning System Management, Employee Skills Management, CRM Include: Extend: Generalization:		
Normal Flow of Events: 1. Admin starts up system. 2. System provides Admin with Main menu for the System. System asks Admin if he/she want to add employee, change employee records or retain employee records. b. If Admin wants to add employee, he/she clicks on Add Employee Link and executes **S-1** c. If Admin wants to change employe status, he/she clicks on Change Employee Records Link and executes **S-2** d. If Admin wants to retain employee records, he/she clicks on Find Employee Records Link and executes **S-3** 3. System returns Admin to Main Menu of System		
SubFlows: **S-1:** New Employee 1. System asks Admin for relevant information 2. Admin enters relevant information 3. Admin submits information to the System		

S-2: Update Records: 1. System asks Admin for employee information 2. Admin enters information about employee status 3. Admin submits information to the System **S-3:** Retain Records 1. System asks Admin for search information 2. Admin enters employee to search. 3. System provides requested employee information
Alternate/Exceptional Flows: The System produces an Error Message.

Use Case Name: Customer Relationship Management	ID: 4	Importance Level: High
Primary Actor: Supervisor (SP)	Use Case Type: Detail, Real	
Stakeholders and Interests: Customer wants a SE. SP wants to create SO find a SE match and schedule SE, alert when no SE is found, and record SOas serviced ● Creation and scheduling of Service Order's (SO) ● Searching for matching SE ● Alert when no SE is found for the SO ● Schedule the SE for the SO ● Record when the SO has been serviced		
Brief Description: This use case describes how customers issue orders and the SPMS fulfills the order with an SE.		
Trigger: Customer issues Service Order (SO). Type: External		
Relationships: Association: Supervisor (SP), Admin, Employee Skills Management, Employee Records Management Include: Extend: Generalization:		

Normal Flow of Events: 1. SP starts up system. 2. SP is provided with Main Menu. 3. System asks if SP would like to request an SE If SP wants to, he/she Clicks on Request SE and executes to S-1 4. System returns the SP to the main menu
SubFlows: **S-1:** Create SO 1. The system asks the SP for required information 2. SP enters required information 3. SP submits information. 4. IF system Finds a SE that meets the required information, the system searches a SE for the SP and returns the SE to main menu IF a SE is found, the system schedules the SE for the SP Else the system alerts the SP of no SE Found. 5. The system marks the SO as serviced.
Alternate/Exceptional Flows: The System produces an Error Message.

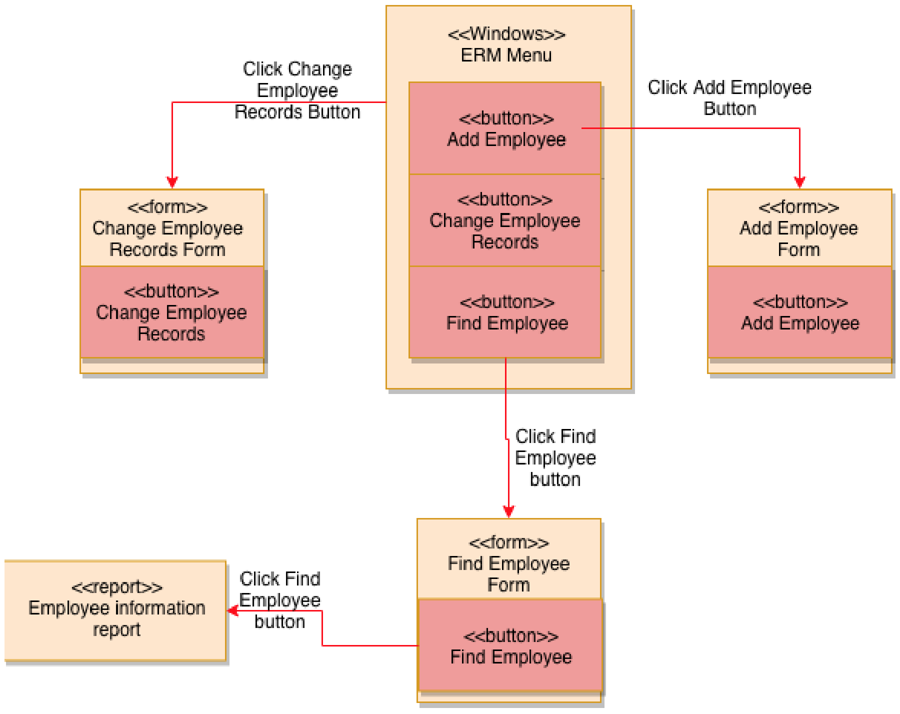
Use Scenario

Use Scenario: existing Employee needs to complete a course
1. Supervisor has allocated seats for the employee to fill according to Supervisor needs(0) 2. Employee will request a class(1) and

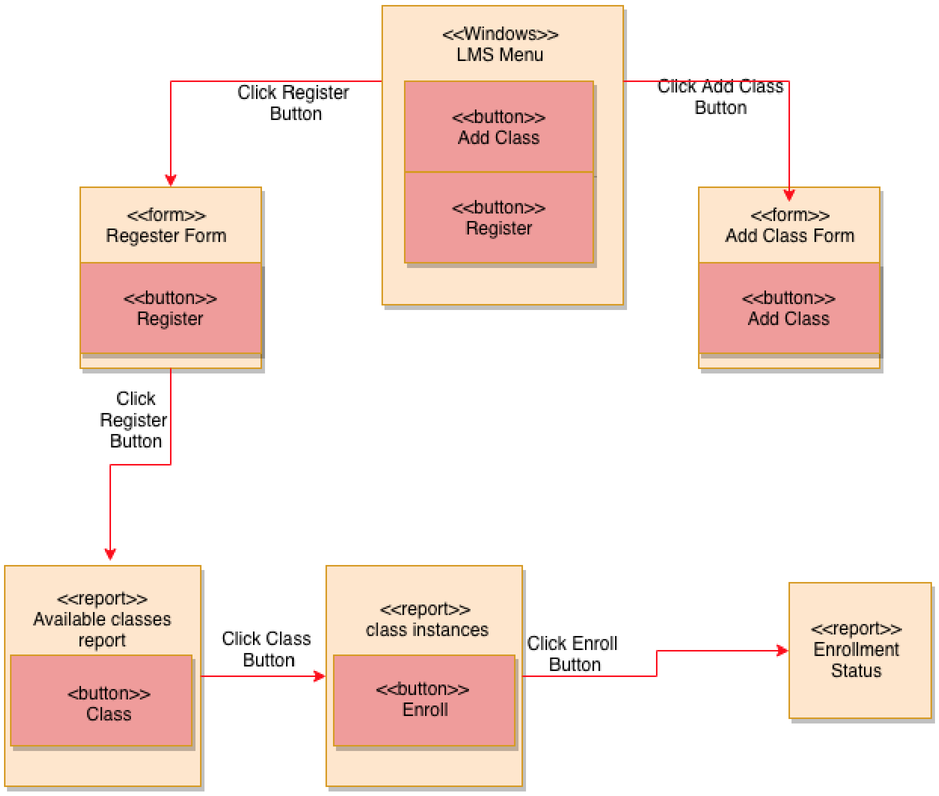
provide his/her name, role, department, Employee ID, status, and Supervisor ID. If employee matches class requirements, employee will be enrolled(2) 3. Product Specialist defines course requirements.(3)A prerequisite requirement will prevent registration for a course if it has not been met. Employee must take prerequisite classes.(4) 4. Once class is completed, Instructor or machine marks the course as satisfied.(5).Admin changes employee status(6). 5. Supervisor increases employee skill as they become more proficient(7).

Use Scenario: Customer needs a Service Engineer
1. Customer requests an Engineer to fulfill a need on a Service Order(1). A service order will include customerID, description, Employee Skilled ID, Product ID and Status. 2. Supervisor Receives Service order and tries to match an employee with the Service Order(2). 3. Scheduler looks over employee skills and employee status. 4. If there is a match, Schedule an engineer for a Service order(3). If there is no match(3.1) Alert customer that no Service engineer was found. 5. Else record that the service order has been serviced(4)

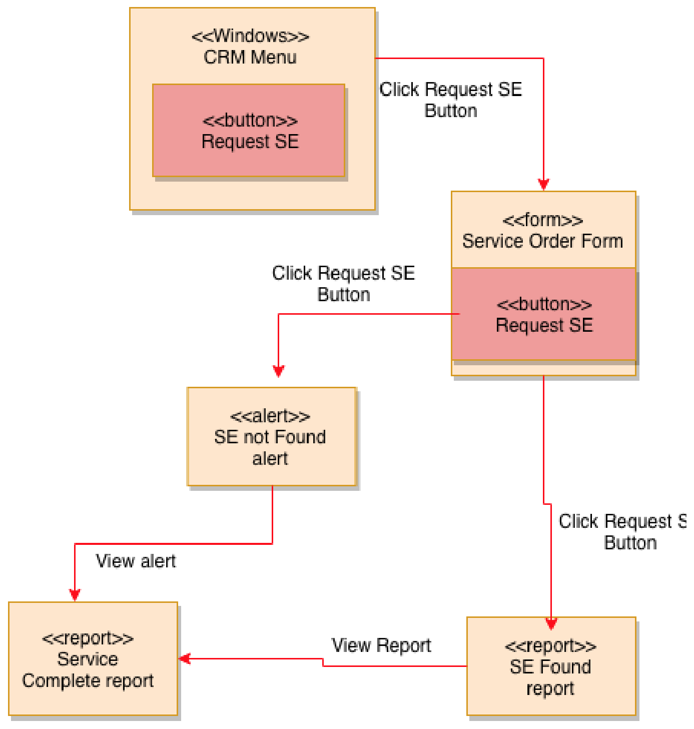
Web Navigation Design



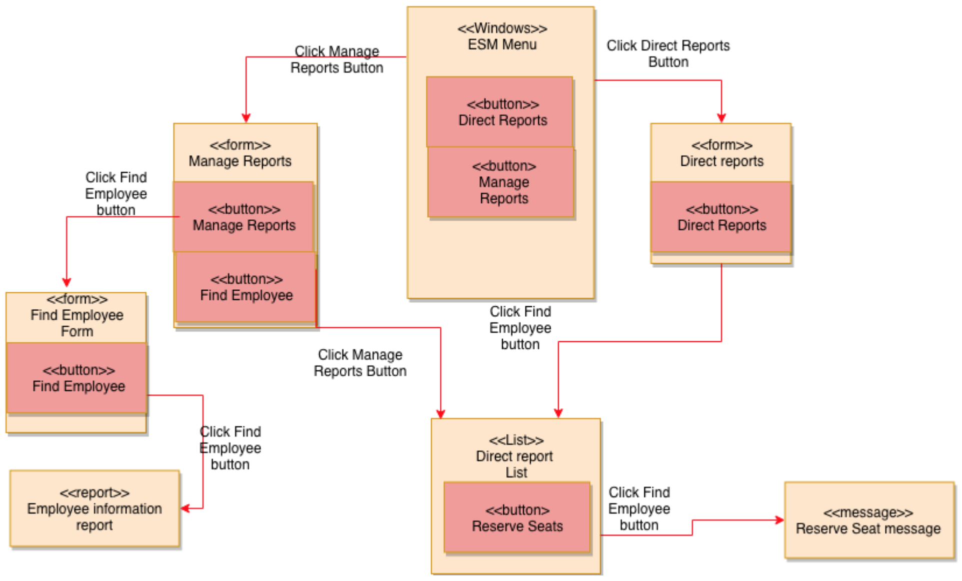
<<Windows>>
ERM Menu
<<button>>
Add Employee
<<button>>
Change Employee Records
<<button>>
Find Employee
Click Change Employee Records Button
<<form>>
Change Employee Records Form
<<button>>
Change Employee Records
Click Add Employee Button
<<form>>
Add Employee Form
<<button>>
Add Employee
Click Find Employee button
<<form>>
Find Employee Form
<<button>>
Find Employee
Click Find Employee button
<<report>>
Employee information report



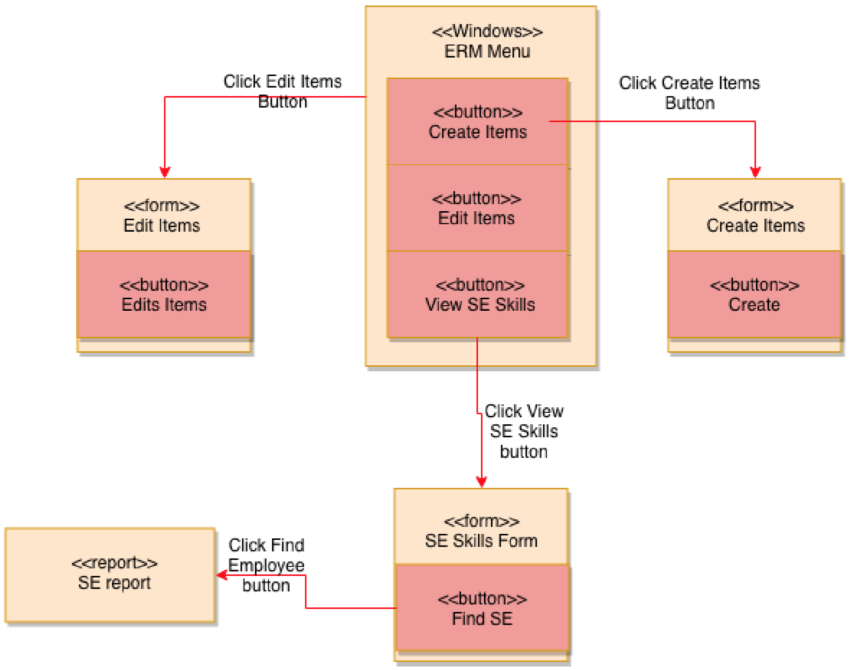
<<Windows>>
LMS Menu
<<button>>
Add Class
<<button>>
Register
Click Register
Button
Click Add Class
Button
<<form>>
Regester Form
<<button>>
Register
<<form>>
Add Class Form
<<button>>
Add Class
Click
Register
Button
<<report>>
Available classes
report
<button>>
Class
Click Class
Button
<<report>>
class instances
<<button>>
Enroll
Click Enroll
Button
<<report>>
Enrollment
Status



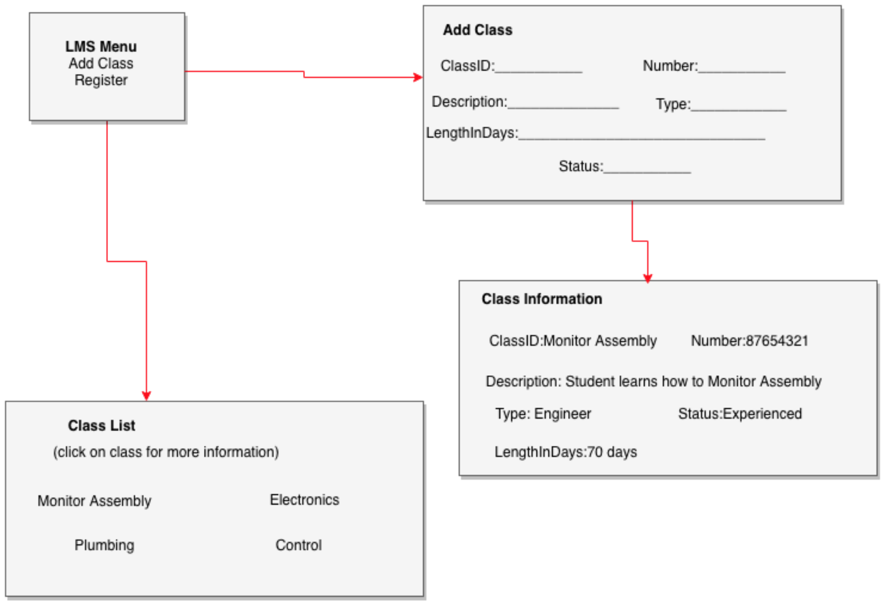
<<Windows>>
CRM Menu
<<button>>
Request SE
Click Request SE Button
<<form>>
Service Order Form
<<button>>
Request SE
Click Request SE Button
<<alert>>
SE not Found alert
Click Request S
Button
View alert
<<report>>
Service Complete report
View Report
<<report>>
SE Found report

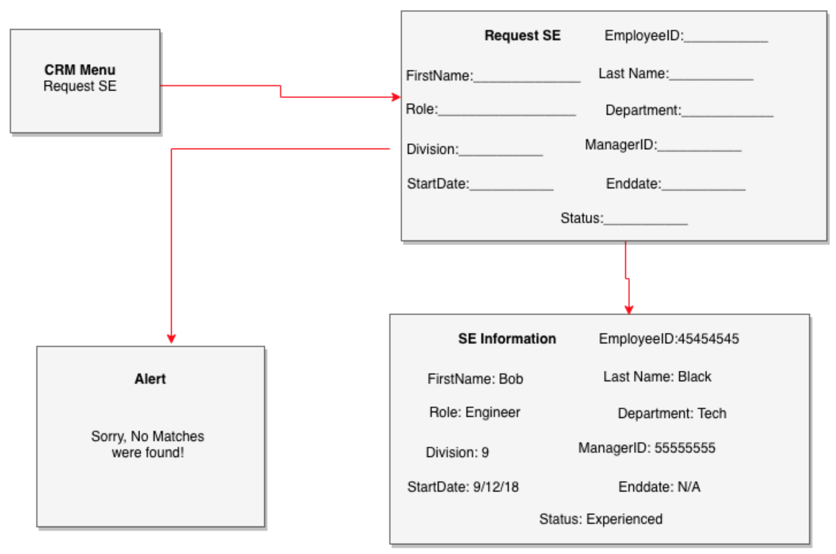


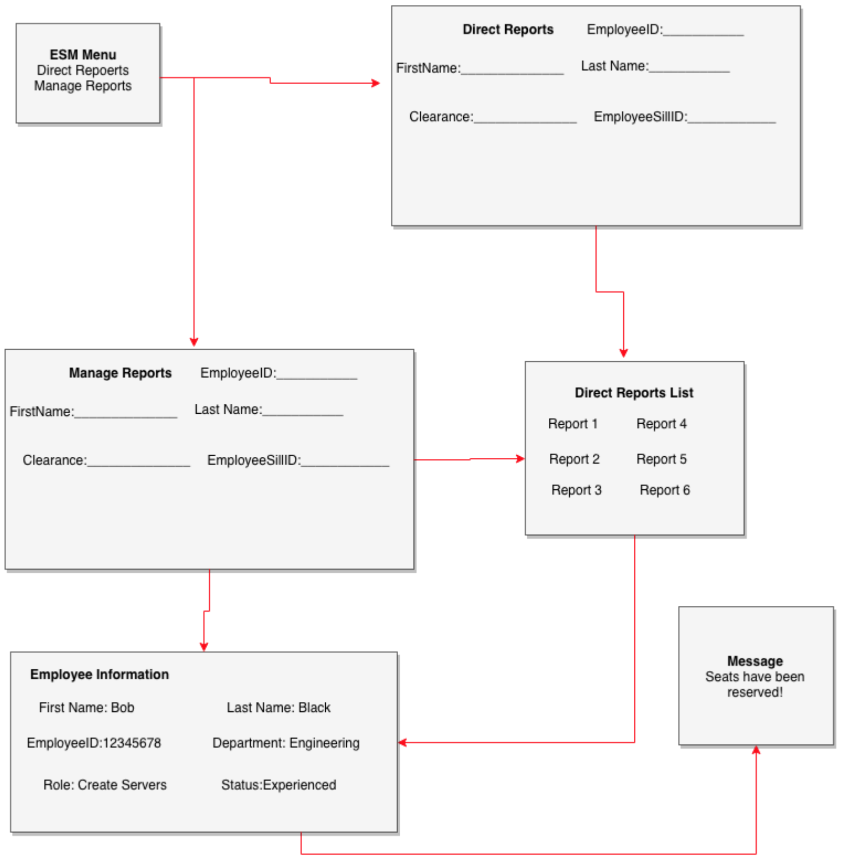
<<Windows>>
ESM Menu
<<button>>
Direct Reports
<<button>
Manage
Reports
Click Manage
Reports Button
Click Direct Reports
Button
<<form>>
Manage Reports
<<button>>
Manage Reports
<<button>>
Find Employee
<<form>>
Direct reports
<<button>>
Direct Reports
Click Find
Employee
button
<<form>>
Find Employee
Form
<<button>>
Find Employee
Click Find
Employee
button
Click Manage
Reports Button
Click Find
Employee
button
<<report>>
Employee information
report
<<List>>
Direct report
List
<<button>
Reserve Seats
Click Find
Employee
button
<<message>>
Reserve Seat message


<<Windows>>
ERM Menu
<<button>>
Create Items
<<button>>
Edit Items
<<button>>
View SE Skills
Click Edit Items
Button
Click Create Items
Button
<<form>>
Edit Items
<<button>>
Edits Items
<<form>>
Create Items
<<button>>
Create
Click View
SE Skills
button
<<form>>
SE Skills Form
<<button>>
Find SE
Click Find
Employee
button
<<report>>
SE report

StoryBoards




CRM Menu
Request SE
Request SE
EmployeeID:___________
FirstName:______________
Last Name:___________
Role:__________________
Department:____________
Division:___________
ManagerID:___________
StartDate:___________
Enddate:___________
Status:___________
Alert
Sorry, No Matches were found!
SE Information
EmployeeID:45454545
FirstName: Bob
Last Name: Black
Role: Engineer
Department: Tech
Division: 9
ManagerID: 55555555
StartDate: 9/12/18
Enddate: N/A
Status: Experienced

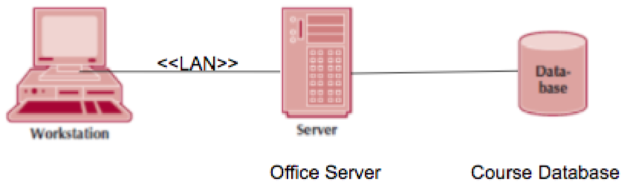

ESM Menu
Direct Repoerts
Manage Reports
Direct Reports
EmployeeID:___________
FirstName:______________
Last Name:___________
Clearance:______________
EmployeeSillID:____________
Manage Reports
EmployeeID:___________
FirstName:______________
Last Name:___________
Clearance:______________
EmployeeSillID:____________
Direct Reports List
Report 1
Report 4
Report 2
Report 5
Report 3
Report 6
Employee Information
First Name: Bob
Last Name: Black
EmployeeID:12345678
Department: Engineering
Role: Create Servers
Status:Experienced
Message
Seats have been reserved!

Physical Architecture Layer:

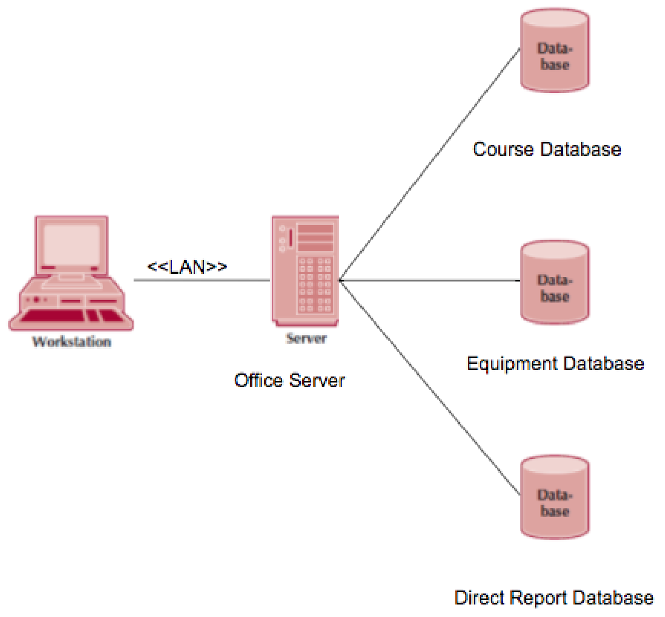
Our System will use Client-Server-Based Architecture because the cost of infrastructure is low even though the cost of development is high. The ease of development is low which means we can easily develop the architecture and the interface capabilities are high which means it will give us more options to develop the User interface and make it more user friendly. Also the security is't the best but its better having that problem because the other two architectures can become overwhelmed. It is also highly scalable which can increase or decrease the architecture's capacity. Also by using the client-Server-based architecture, we place more pressure on the network than on the server or client. Also We decided to go this direction because we also will be using cloud computing because we will have multiple databases for agilent technologies. By using cloud instead of other external devices that could be wasted, we instead are becoming more environmentally friendly since we're not putting the toxic materials into the environment.

Deployment Diagrams:

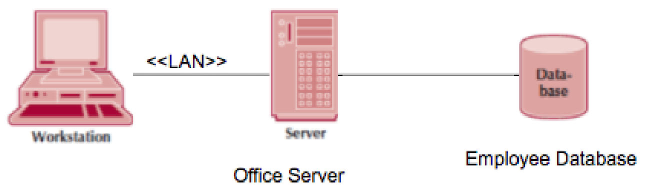
For Learning System Management



For Employee Skills Management

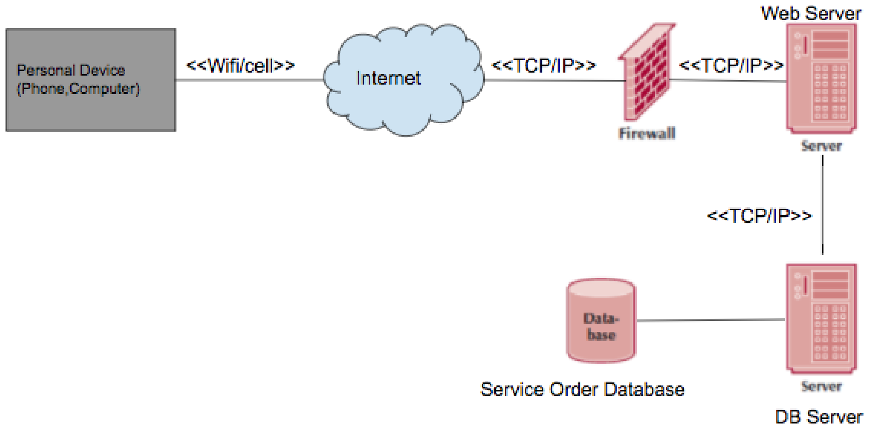


For Employee Records Management System

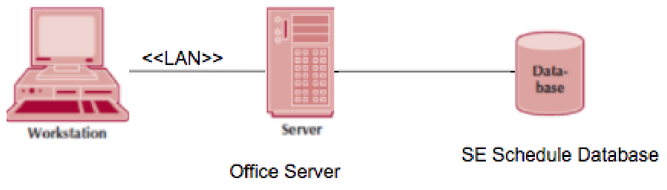


For Customer Relationship Management

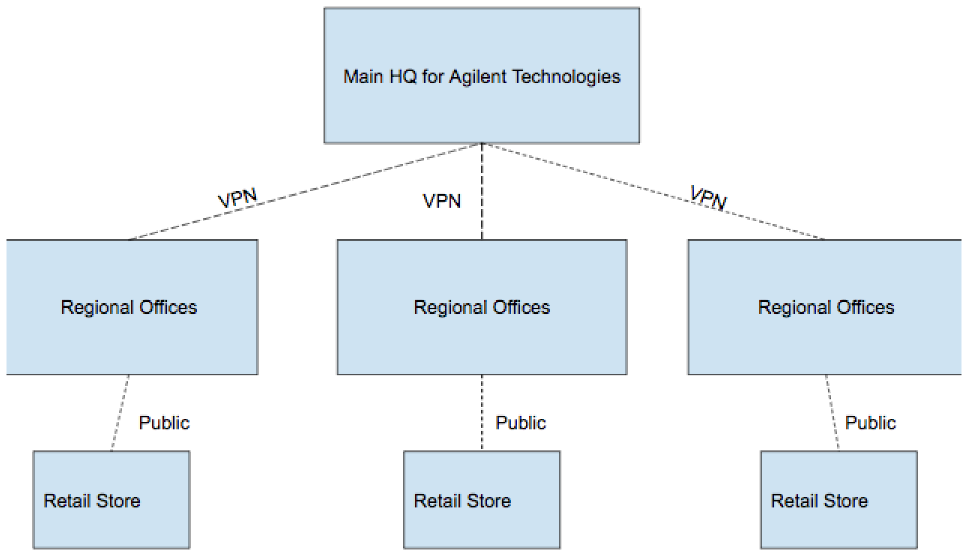
For clients



For SEs



Network Diagrams:



Note: There are multiple retail stores, not just one for each regional office. VPN stands for Virtual Private Network.

Functional & Nonfunctional Requirements

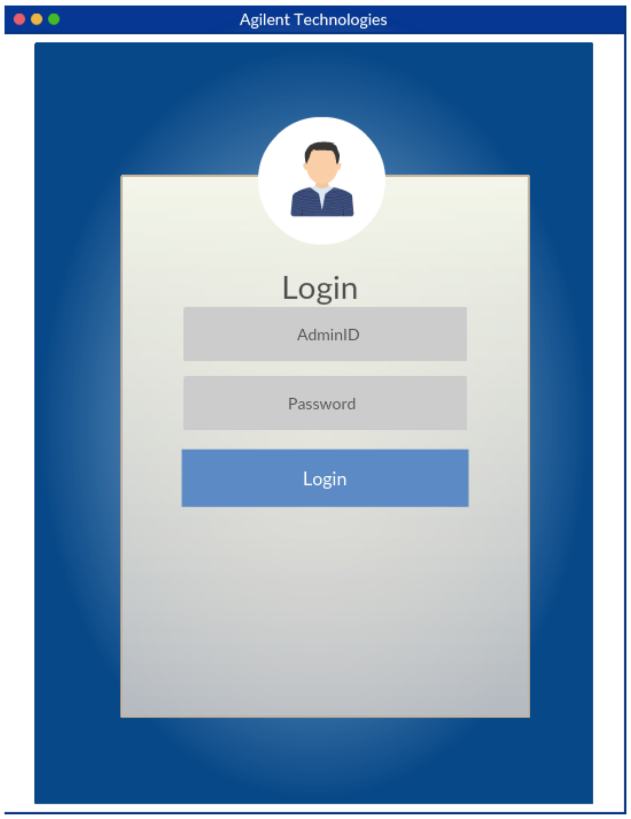
Specifications	Standard Client	Standard Web Server	Standard Application Server	Standard Database Server
Operating System	• Chrome	• Oracle	• Java	• Oracle
Special Software	• Acrobat Reader DC	• Micrsoft IIS	• Linux	• Oracle
Hardware	• 16 GB Memory • 1 TB Disk Drive • Intel Core i7 • 3- 25'' Monitor	• 16 GB Memory • 1 TB Disk Drive • Intel Core i7 • 1- 20'' Monitor	• 8 GB Memory • 1 TB Disk Drive • Intel Xenon • 1-20'' Monitor	• 32 GB Memory • 1 TB Disk Drive • Intel Xenon • 3- 25'' Monitor
Network	• 200 Mbps Ethernet	• 200 Mbps Ethernet	• 200 Mbps Ethernet	• 200 Mbps Ethernet

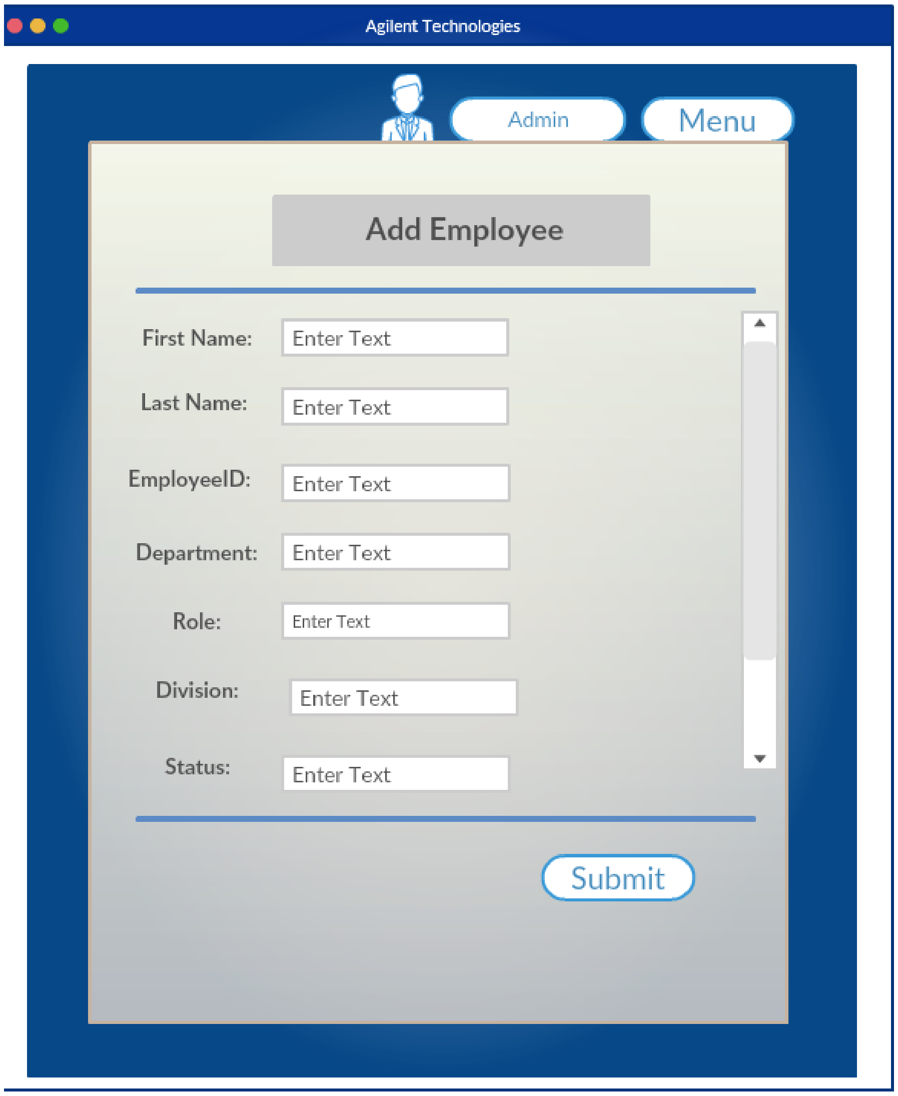
Type of Requirement	Definition	Examples
Technical Environment	• Network must be secure and readily available	• The system will work over the Web with Firefox, Safari Chrome, and IE • All locations have a secure network connection
System Integration	• The system must be fully integrated with Agilent's databases	• The system must be able to convert charts between Microsoft excel and google spreadsheets • The system must be able to export Word docs as Google Docs and Vice versa
Portability	• Must be comfortable operating with smart phones and tablets, depending on user input	• The system must operate with Android devices
Maintainability	• Must make alterations easy and immediate	• New versions will be released every 3 years • The system will make real time changes based on user input every night

Functions and Features	• 25′′ monitor • Compatibility with Google Chrome and Internet Explorer
Performance	• Intel Core i7-6700k • 5000 writes/sec
Legacy Databases and Systems	• Must adapt to all database post 2005
Hardware and OS Strategy	• Limit total number of vendors to 5
Cost of Ownership	• Set aside 2 million/year for salaries • Purchase annual flat rate company license for Microsoft and Adobe products
Political Preferences	• Do not alter UI, only the number of features per list • Place most often used features at the top of the list
Vendor Performance	• Cost optimization in terms of any supplier choice

Web Design/Html Layout

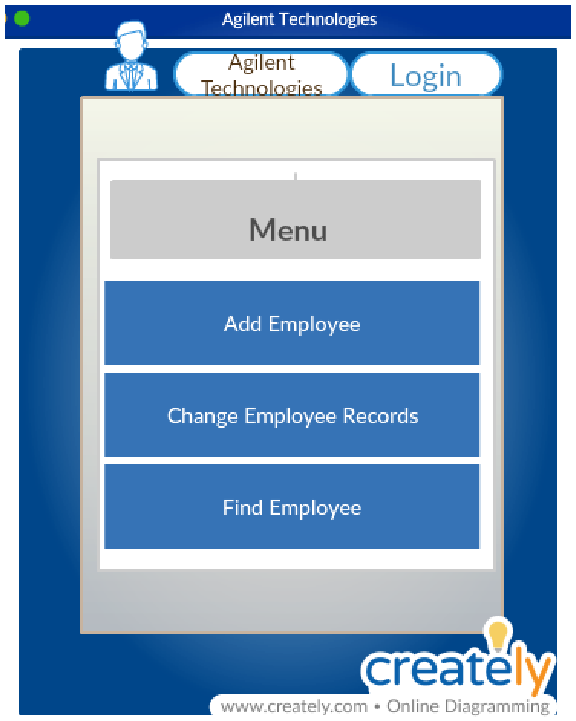
ERM

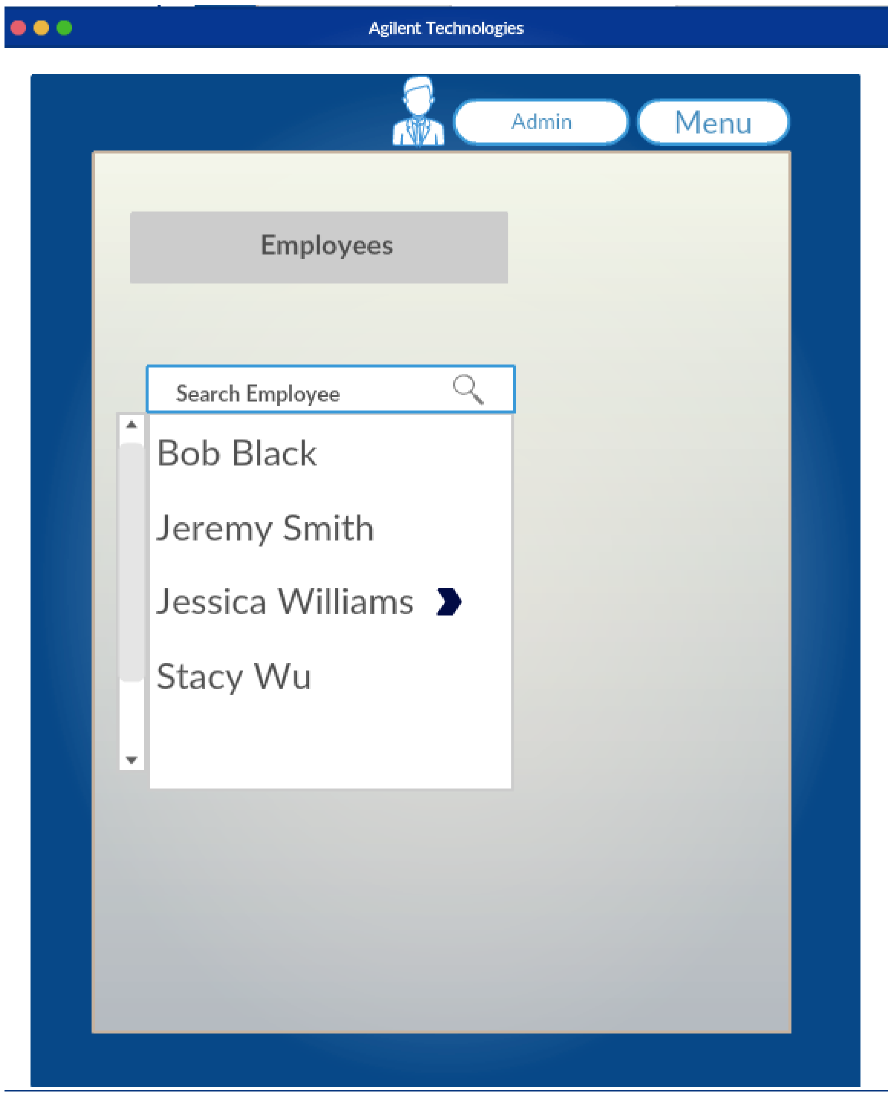


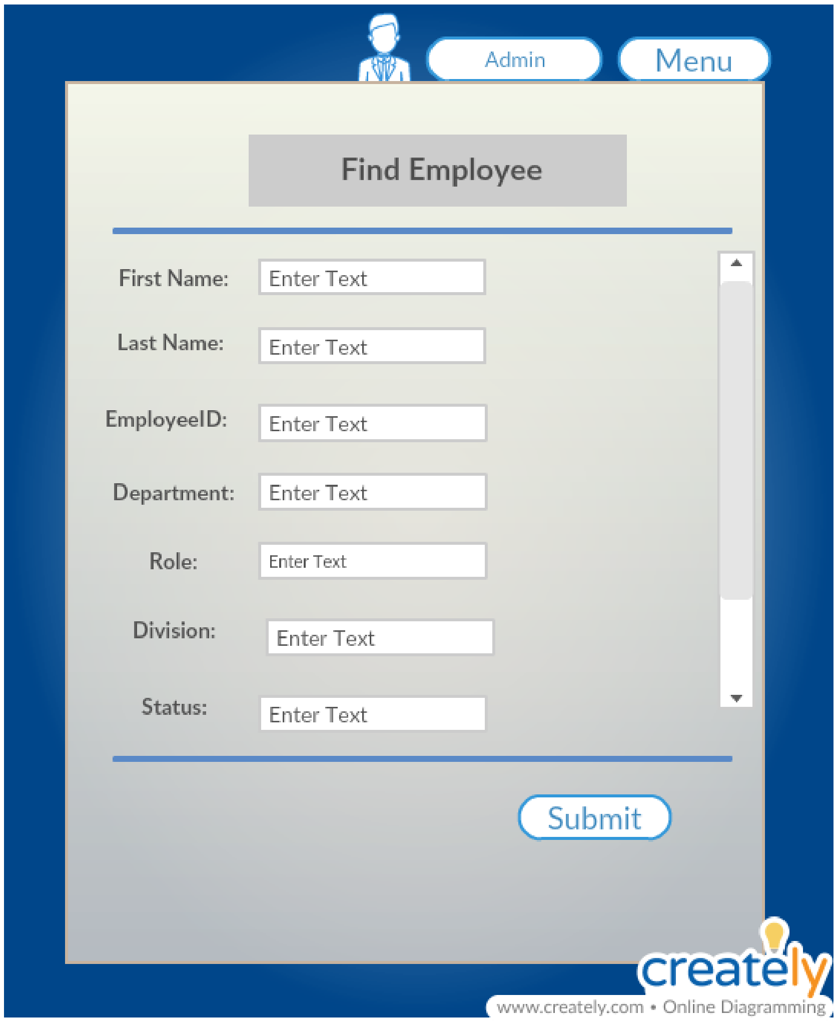


Add Employee

First Name:	Enter Text
Last Name:	Enter Text
EmployeeID:	Enter Text
Department:	Enter Text
Role:	Enter Text
Division:	Enter Text
Status:	Enter Text

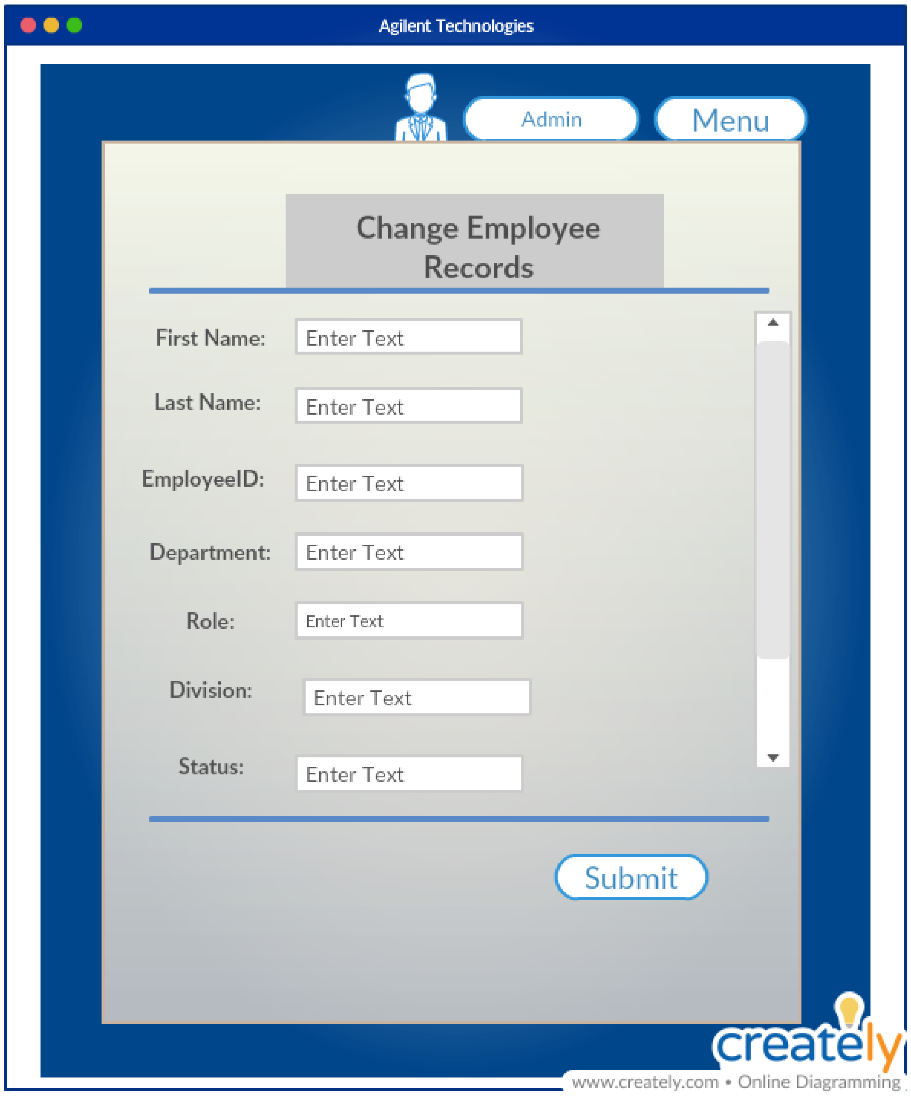

Agilent Technologies
Agilent Technologies
Login
Menu
Add Employee
Change Employee Records
Find Employee
creately
www.creately.com • Online Diagramming


Agilent Technologies
Admin
Menu
Employees
Search Employee
Bob Black
Jeremy Smith
Jessica Williams
Stacy Wu



Find Employee

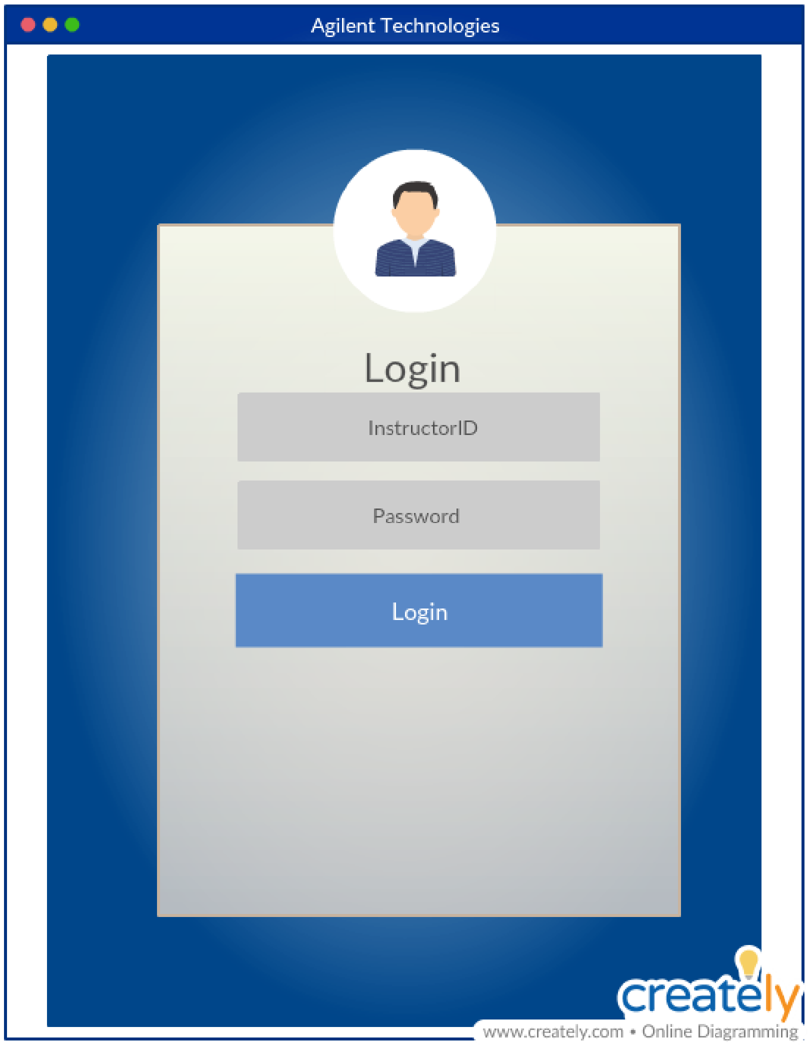
First Name:	Enter Text
Last Name:	Enter Text
EmployeeID:	Enter Text
Department:	Enter Text
Role:	Enter Text
Division:	Enter Text
Status:	Enter Text

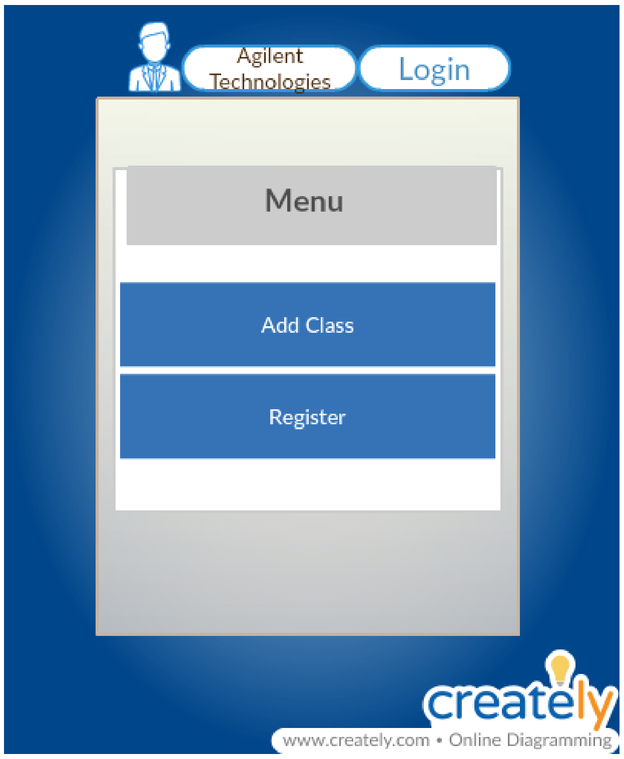


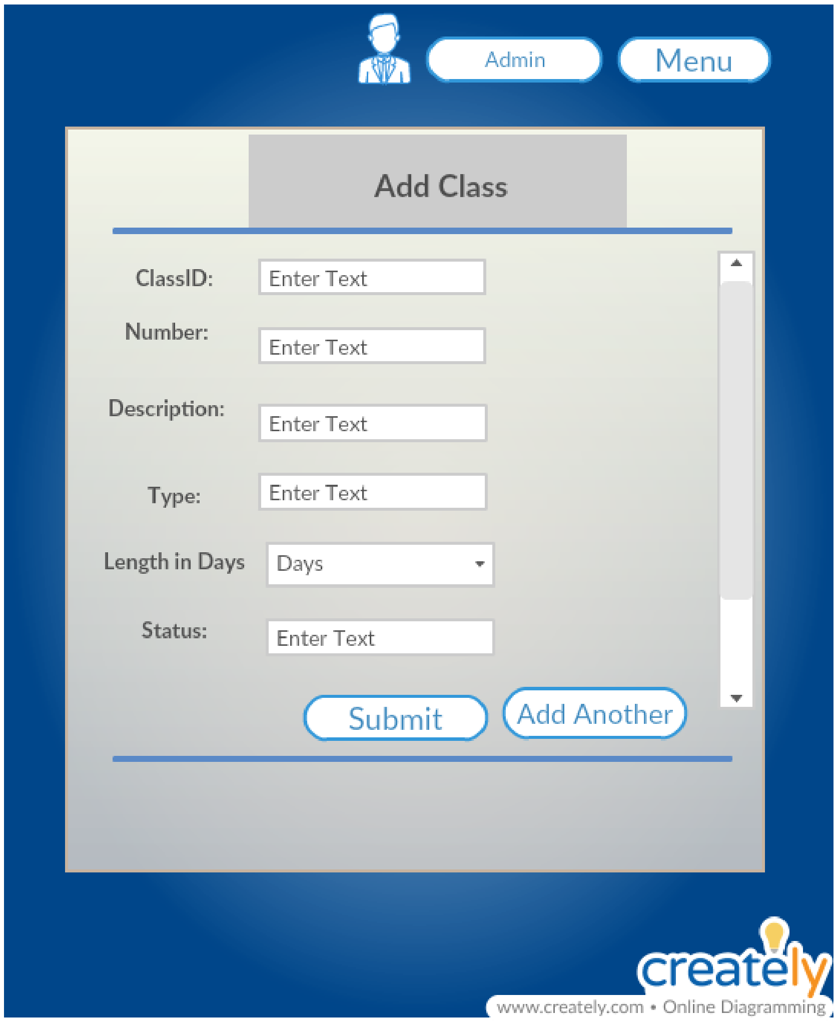
Change Employee Records

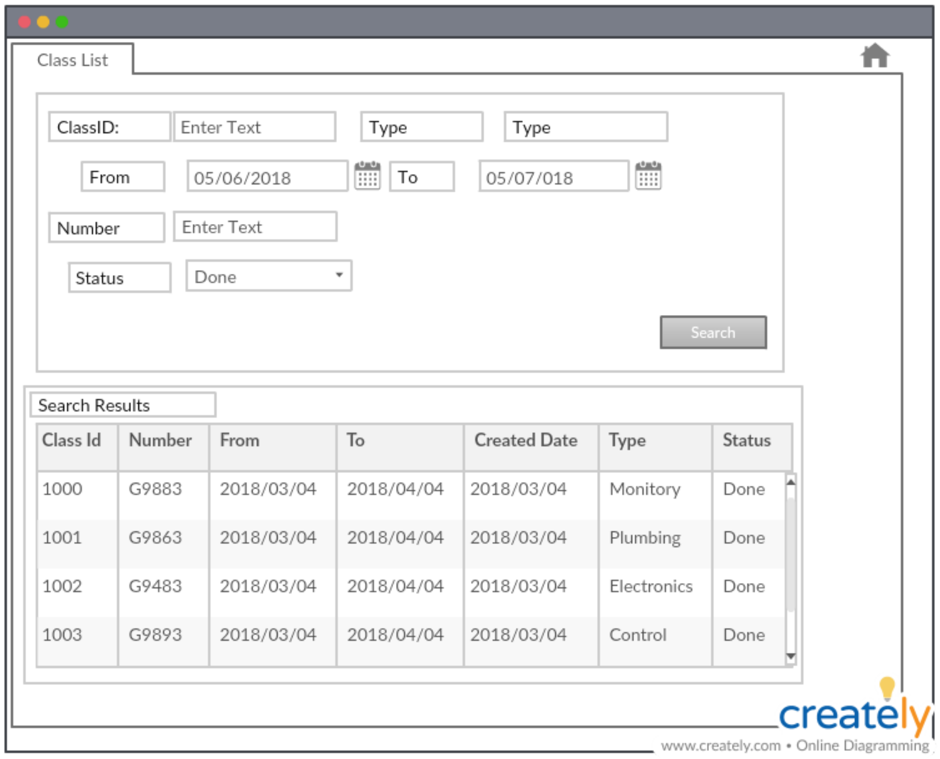
First Name:	Enter Text
Last Name:	Enter Text
EmployeeID:	Enter Text
Department:	Enter Text
Role:	Enter Text
Division:	Enter Text
Status:	Enter Text

LMS


Agilent Technologies
Login
InstructorID
Password
Login
creately
www.creately.com • Online Diagramming


Agilent Technologies
Login
Menu
Add Class
Register
creately
www.creately.com • Online Diagramming

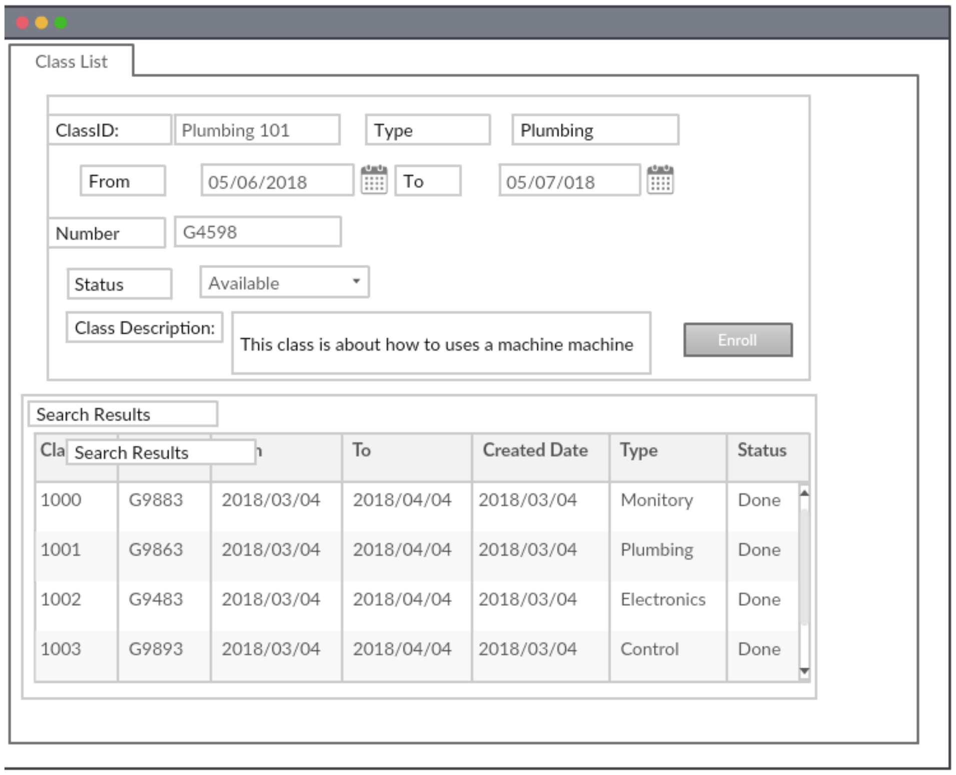

Admin
Menu
Add Class
ClassID:
Enter Text
Number:
Enter Text
Description:
Enter Text
Type:
Enter Text
Length in Days
Days
Status:
Enter Text
Submit
Add Another
creately
www.creately.com • Online Diagramming


Class List

ClassID:	Enter Text	Type	Type
From	05/06/2018	To	05/07/018
Number	Enter Text		
Status	Done		

Search Results

Class Id	Number	From	To	Created Date	Type	Status
1000	G9883	2018/03/04	2018/04/04	2018/03/04	Monitory	Done
1001	G9863	2018/03/04	2018/04/04	2018/03/04	Plumbing	Done
1002	G9483	2018/03/04	2018/04/04	2018/03/04	Electronics	Done
1003	G9893	2018/03/04	2018/04/04	2018/03/04	Control	Done



Class List

ClassID: Plumbing 101 | Type | Plumbing

From | 05/06/2018 | To | 05/07/018

Number | G4598

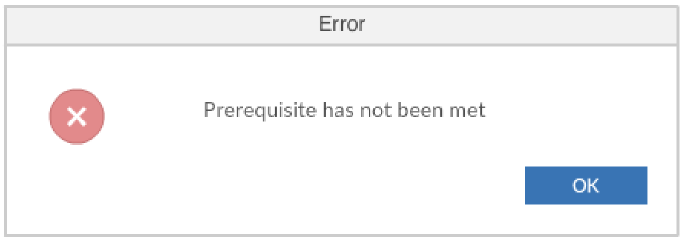
Status | Available

Class Description: | This class is about how to uses a machine machine

Enroll

Search Results

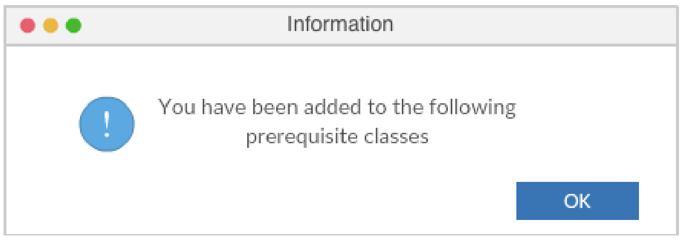
Cla	Search Results	n	To	Created Date	Type	Status
1000	G9883	2018/03/04	2018/04/04	2018/03/04	Monitory	Done
1001	G9863	2018/03/04	2018/04/04	2018/03/04	Plumbing	Done
1002	G9483	2018/03/04	2018/04/04	2018/03/04	Electronics	Done
1003	G9893	2018/03/04	2018/04/04	2018/03/04	Control	Done



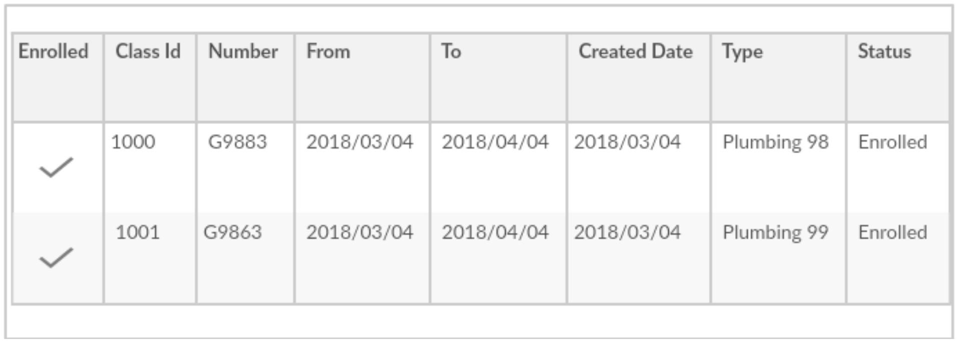
Error

Prerequisite has not been met

OK

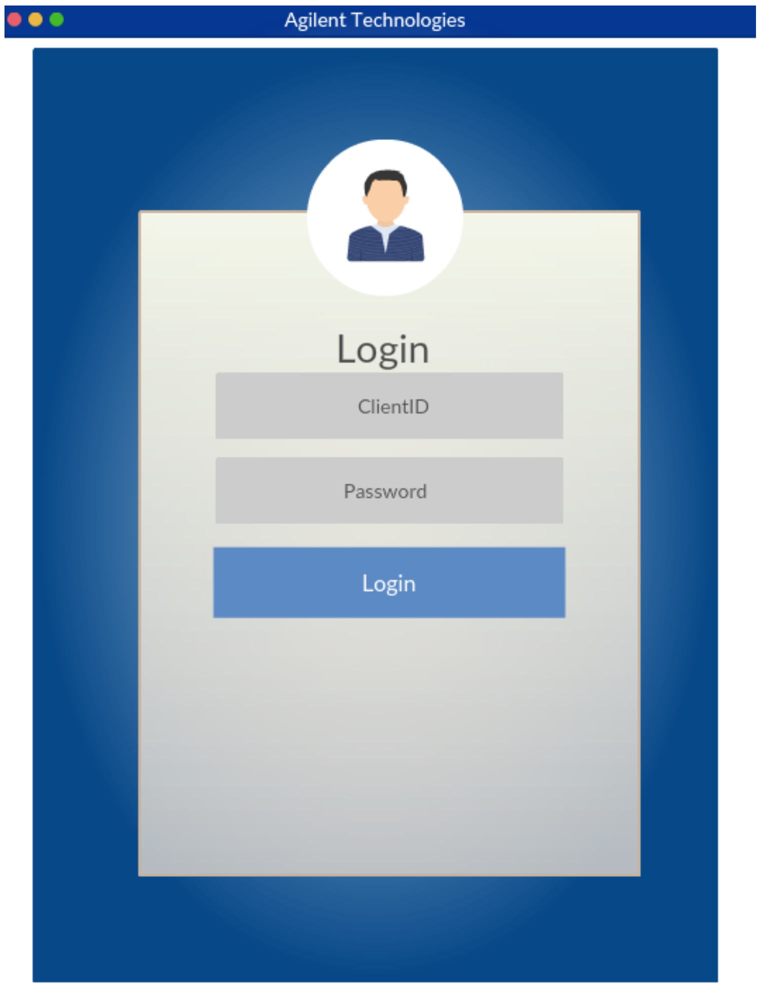


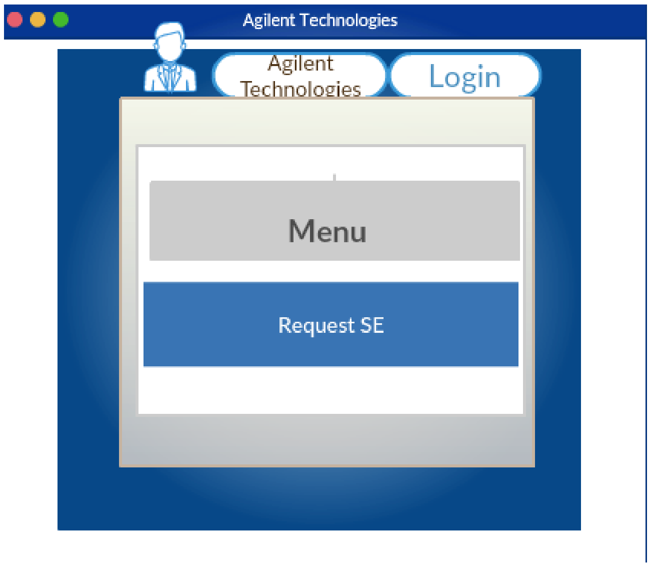
You have been added to the following prerequisite classes

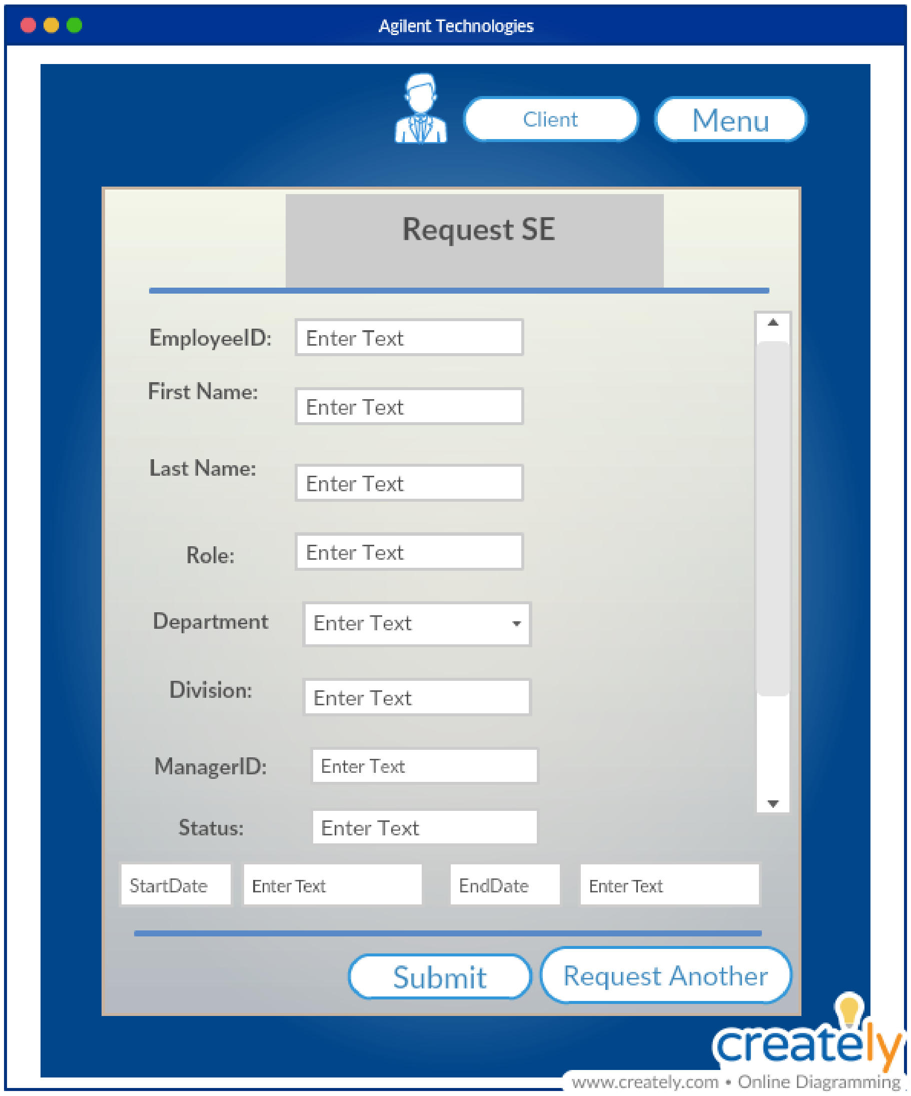


Enrolled	Class Id	Number	From	To	Created Date	Type	Status
✓	1000	G9883	2018/03/04	2018/04/04	2018/03/04	Plumbing 98	Enrolled
✓	1001	G9863	2018/03/04	2018/04/04	2018/03/04	Plumbing 99	Enrolled

CRM




Agilent Technologies
Agilent Technologies
Login
Menu
Request SE



Agilent Technologies

Client

Menu

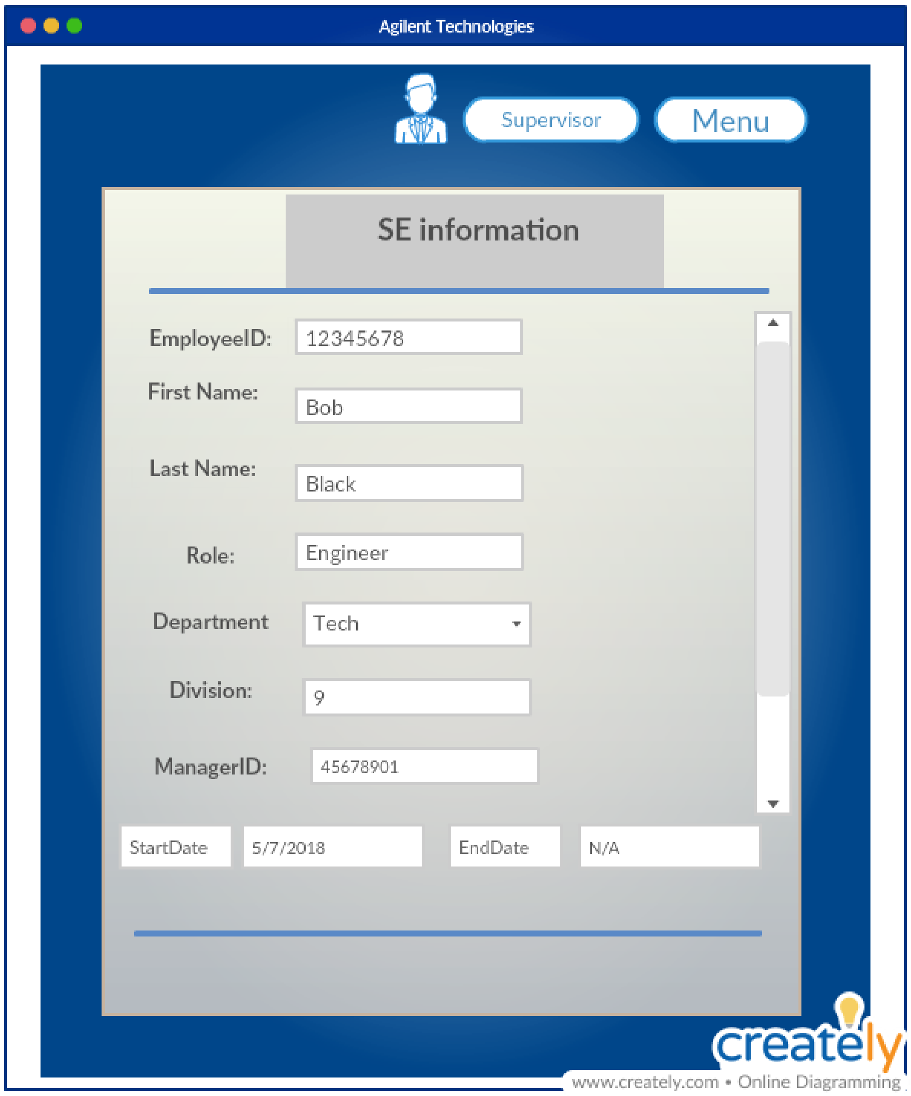
Request SE

EmployeeID:	Enter Text
First Name:	Enter Text
Last Name:	Enter Text
Role:	Enter Text
Department	Enter Text
Division:	Enter Text
ManagerID:	Enter Text
Status:	Enter Text

StartDate Enter Text EndDate Enter Text

Submit Request Another

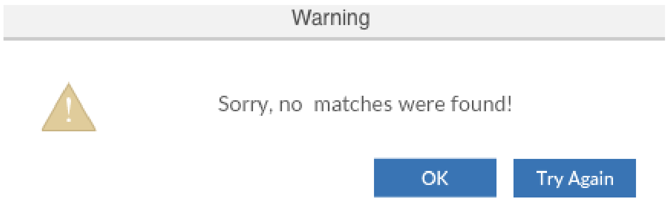
creately
www.creately.com • Online Diagramming



SE information

EmployeeID:	12345678
First Name:	Bob
Last Name:	Black
Role:	Engineer
Department	Tech
Division:	9
ManagerID:	45678901

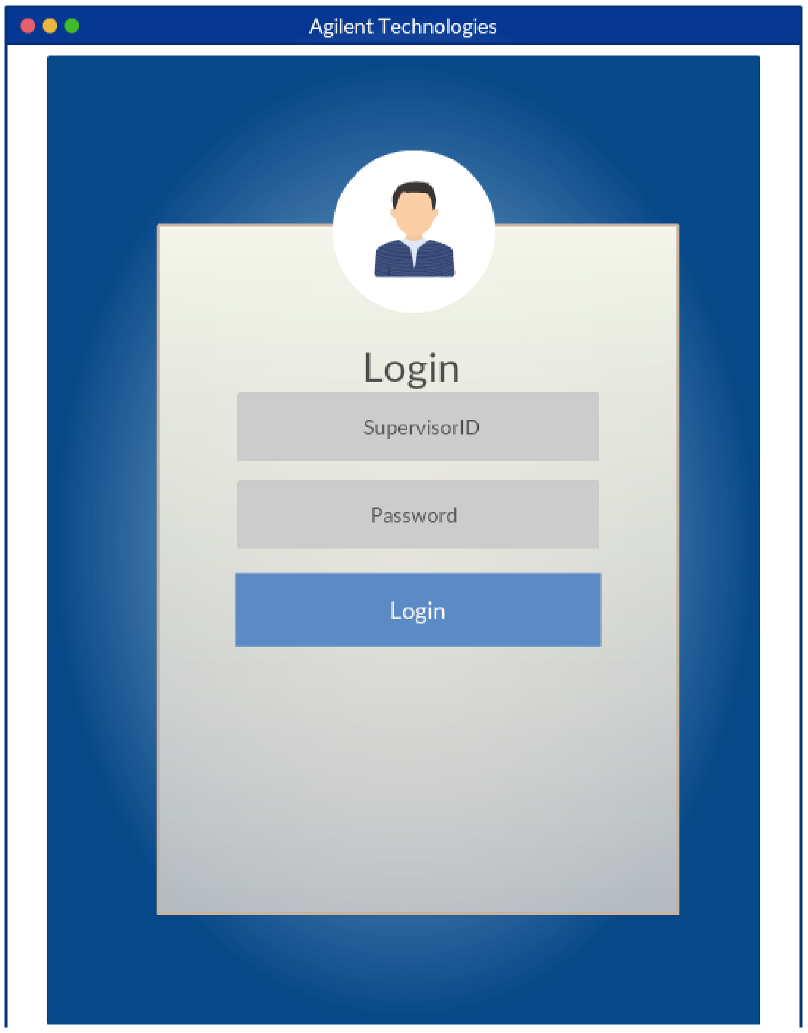
StartDate	5/7/2018	EndDate	N/A

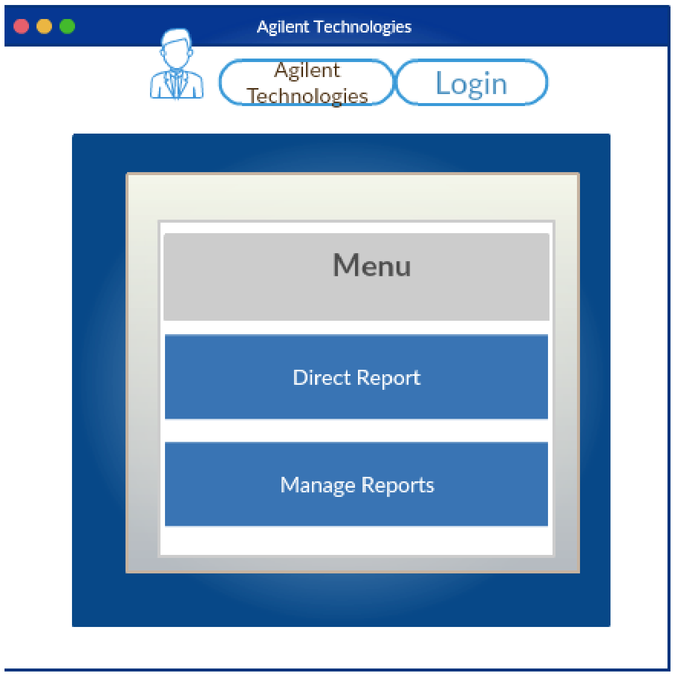


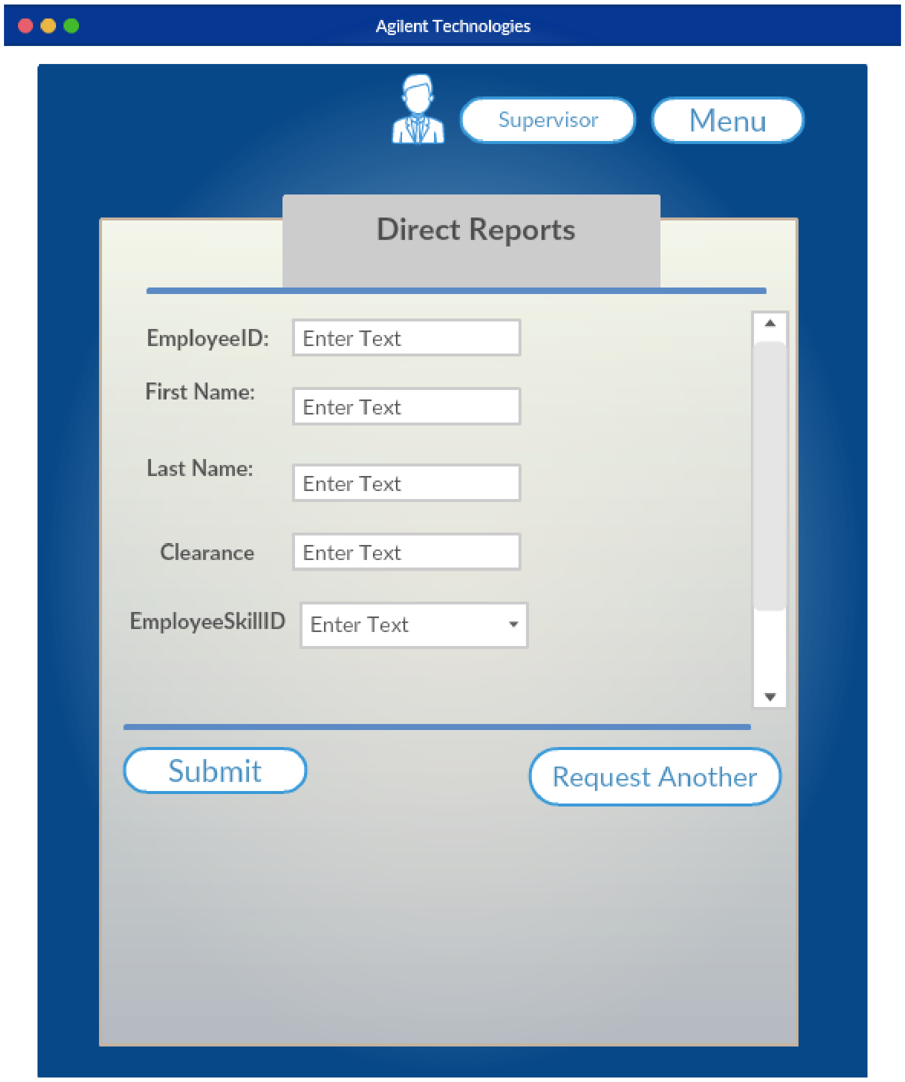
Warning

Sorry, no matches were found!

ESM

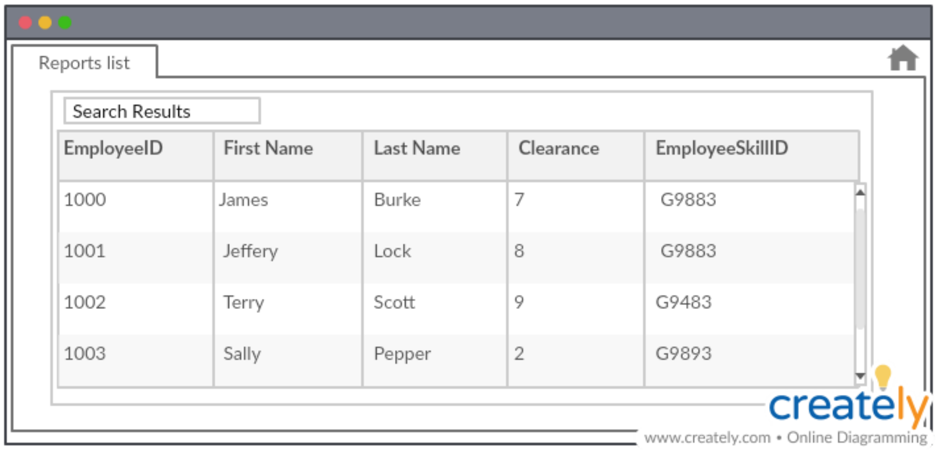



Agilent Technologies
Agilent Technologies
Login
Menu
Direct Report
Manage Reports

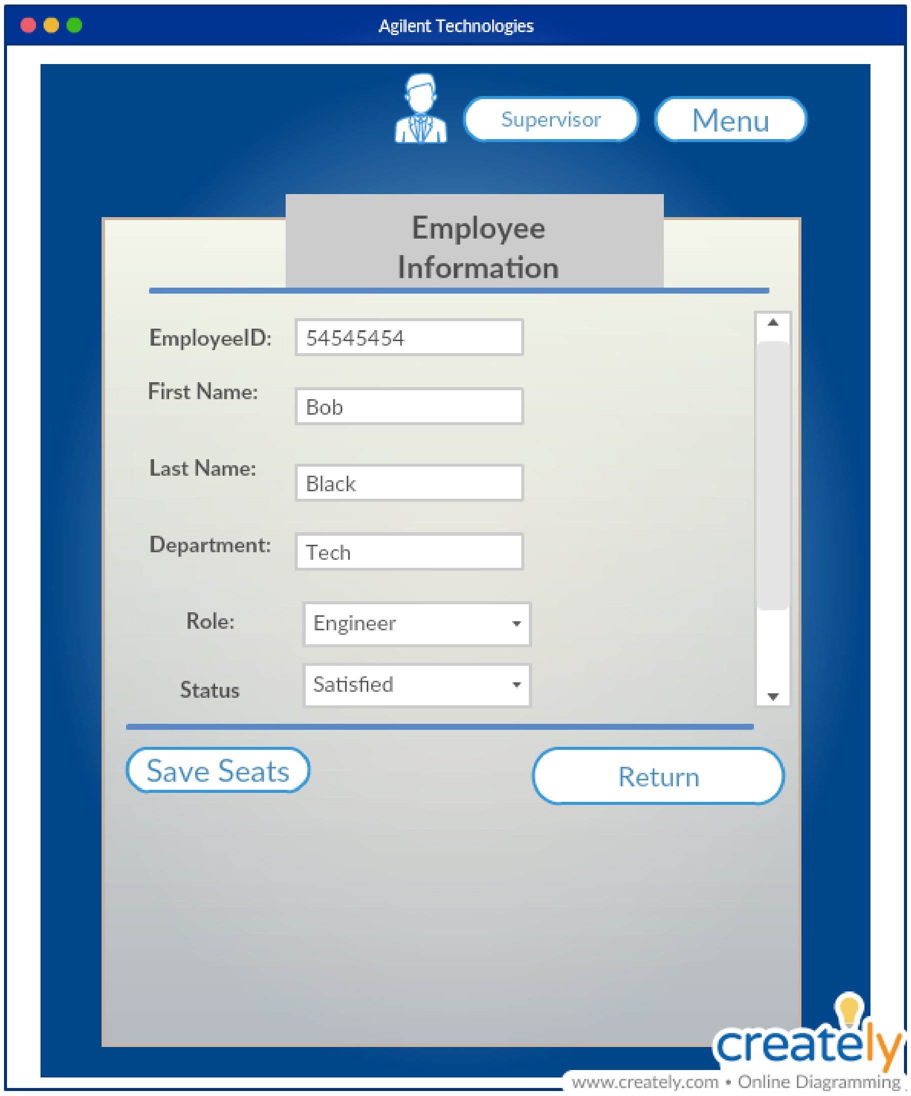


Direct Reports

EmployeeID:	Enter Text
First Name:	Enter Text
Last Name:	Enter Text
Clearance	Enter Text
EmployeeSkillID	Enter Text

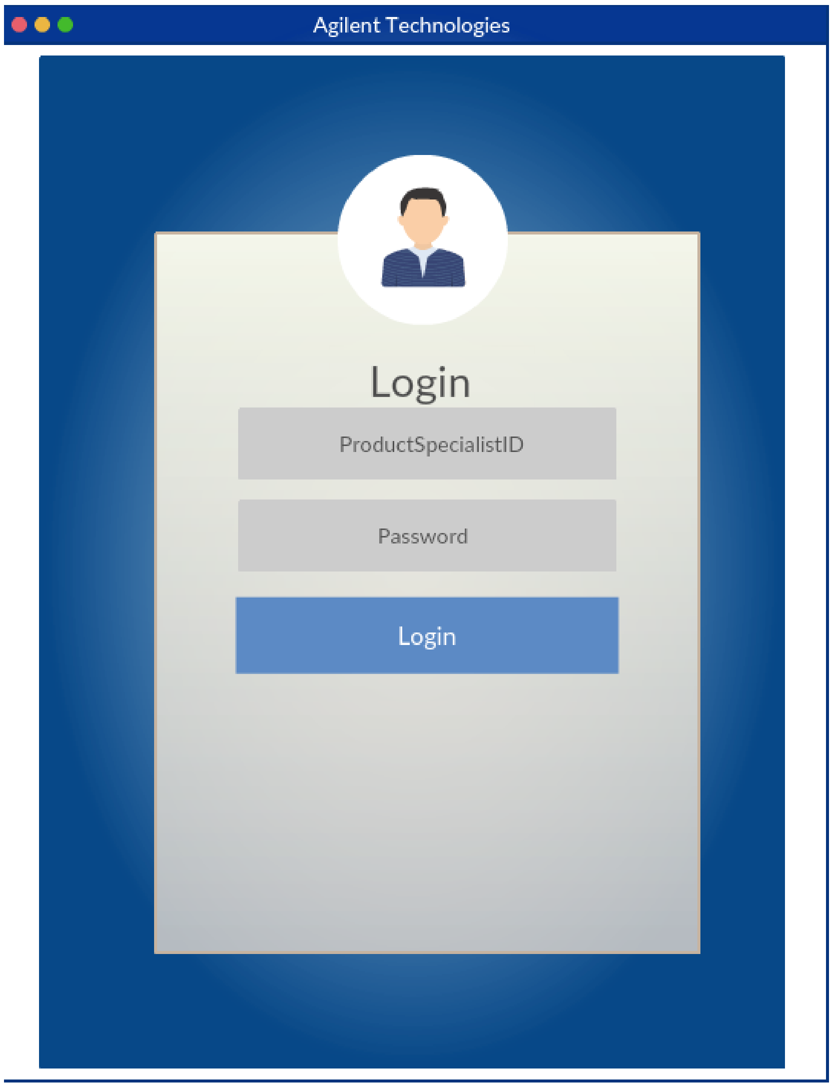


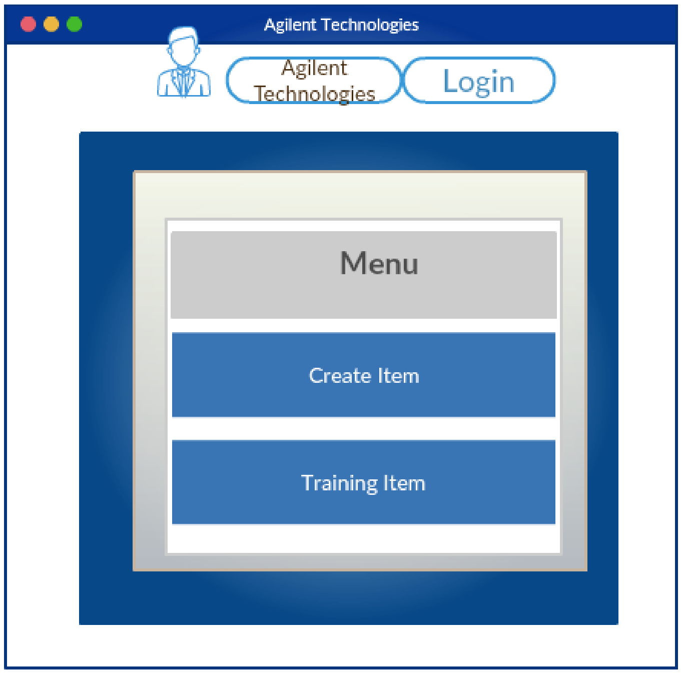
EmployeeID	First Name	Last Name	Clearance	EmployeeSkillID
1000	James	Burke	7	G9883
1001	Jeffery	Lock	8	G9883
1002	Terry	Scott	9	G9483
1003	Sally	Pepper	2	G9893

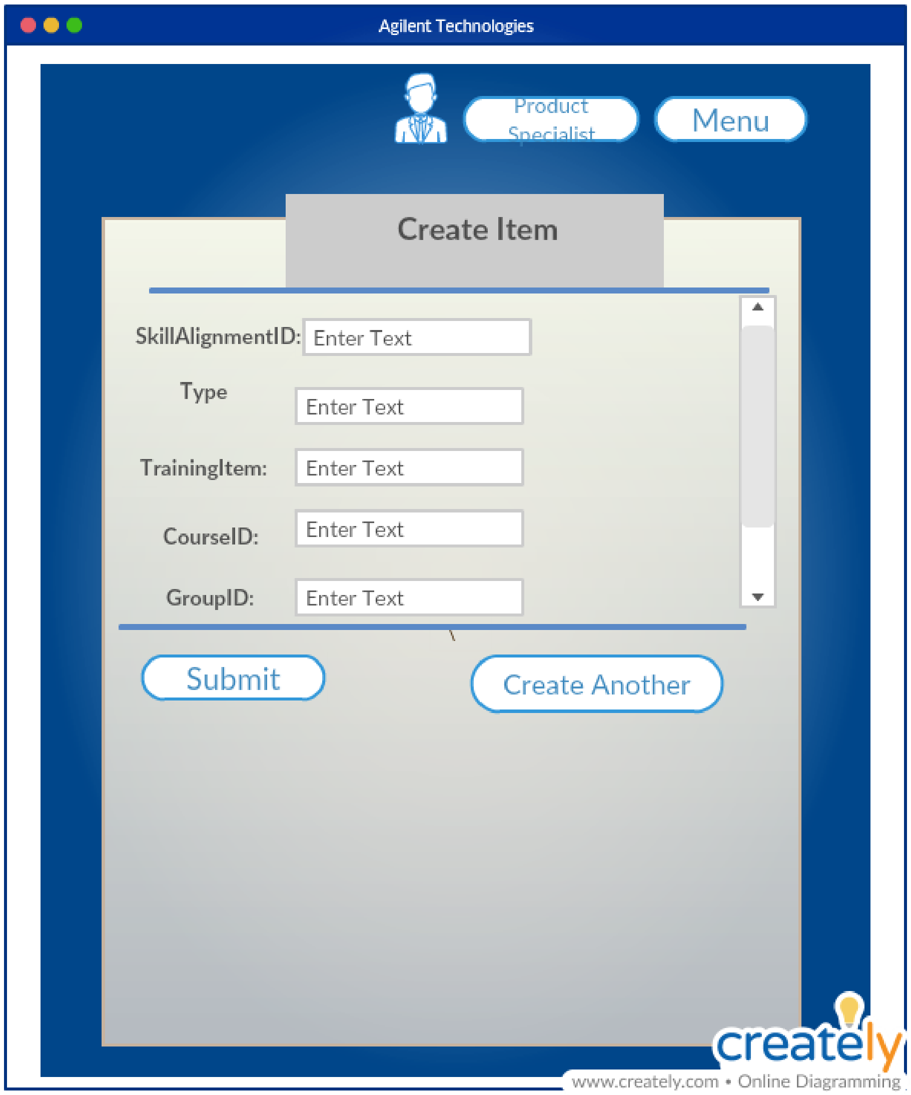


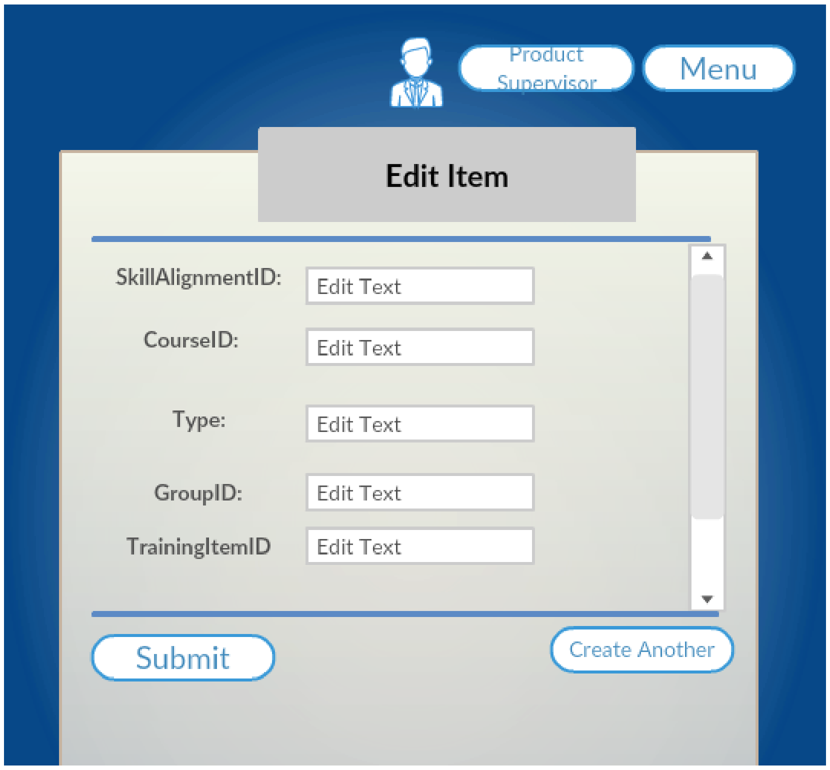
Employee Information

EmployeeID:	54545454
First Name:	Bob
Last Name:	Black
Department:	Tech
Role:	Engineer
Status	Satisfied


Agilent Technologies
Login
ProductSpecialistID
Password
Login

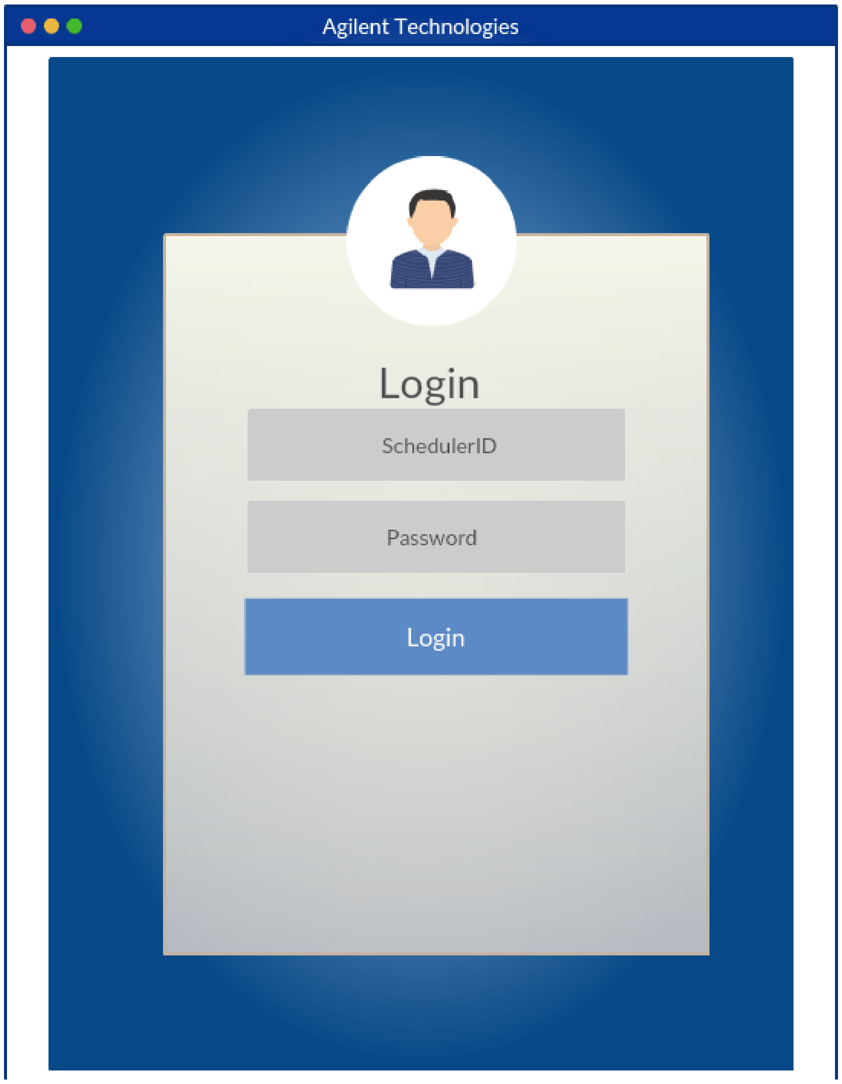

Agilent Technologies
Agilent Technologies
Login
Menu
Create Item
Training Item

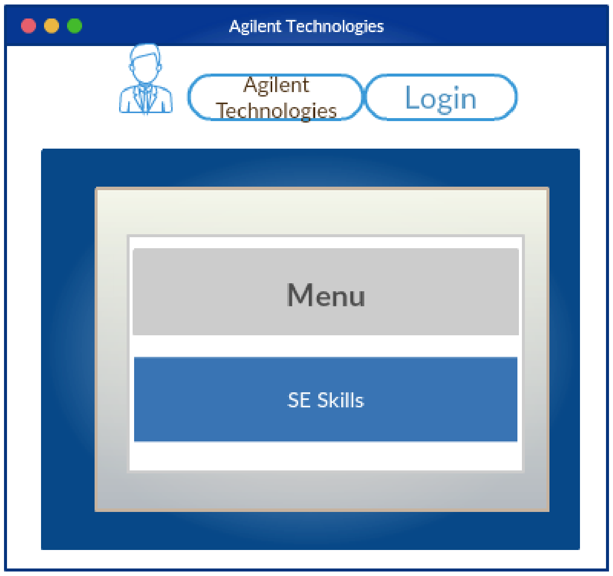

Agilent Technologies
Product Specialist
Menu
Create Item
SkillAlignmentID: Enter Text
Type Enter Text
TrainingItem: Enter Text
CourseID: Enter Text
GroupID: Enter Text
Submit
Create Another

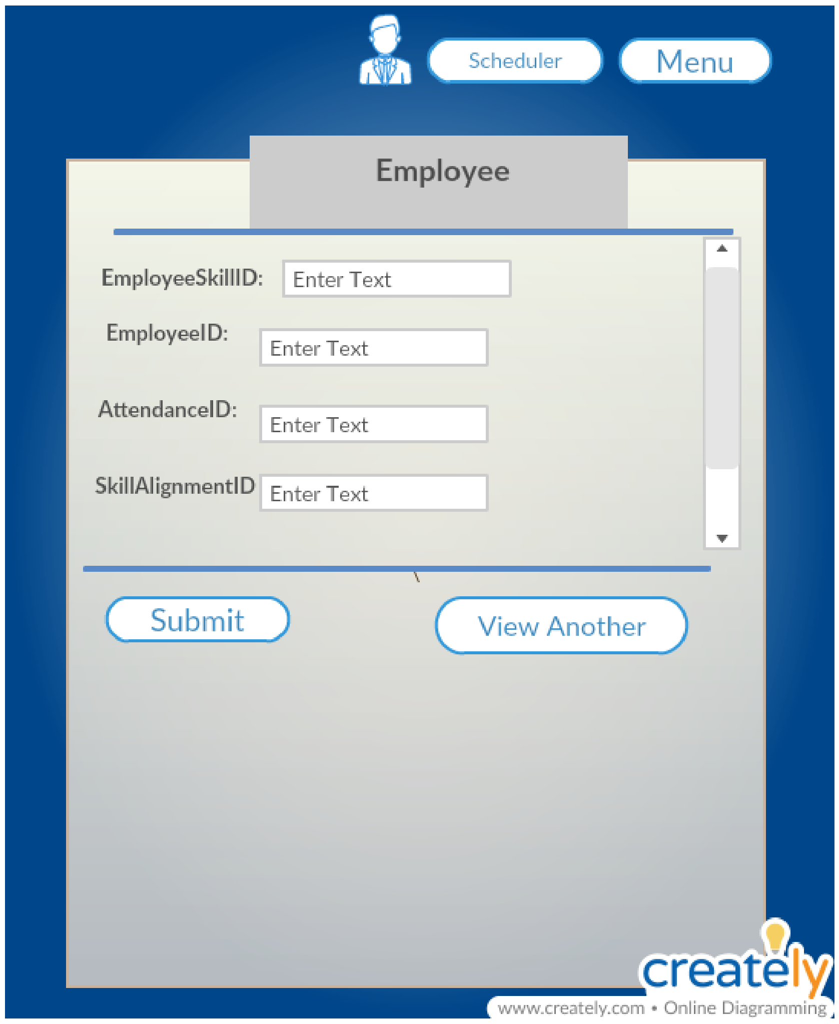


Edit Item

SkillAlignmentID:	Edit Text
CourseID:	Edit Text
Type:	Edit Text
GroupID:	Edit Text
TrainingItemID	Edit Text

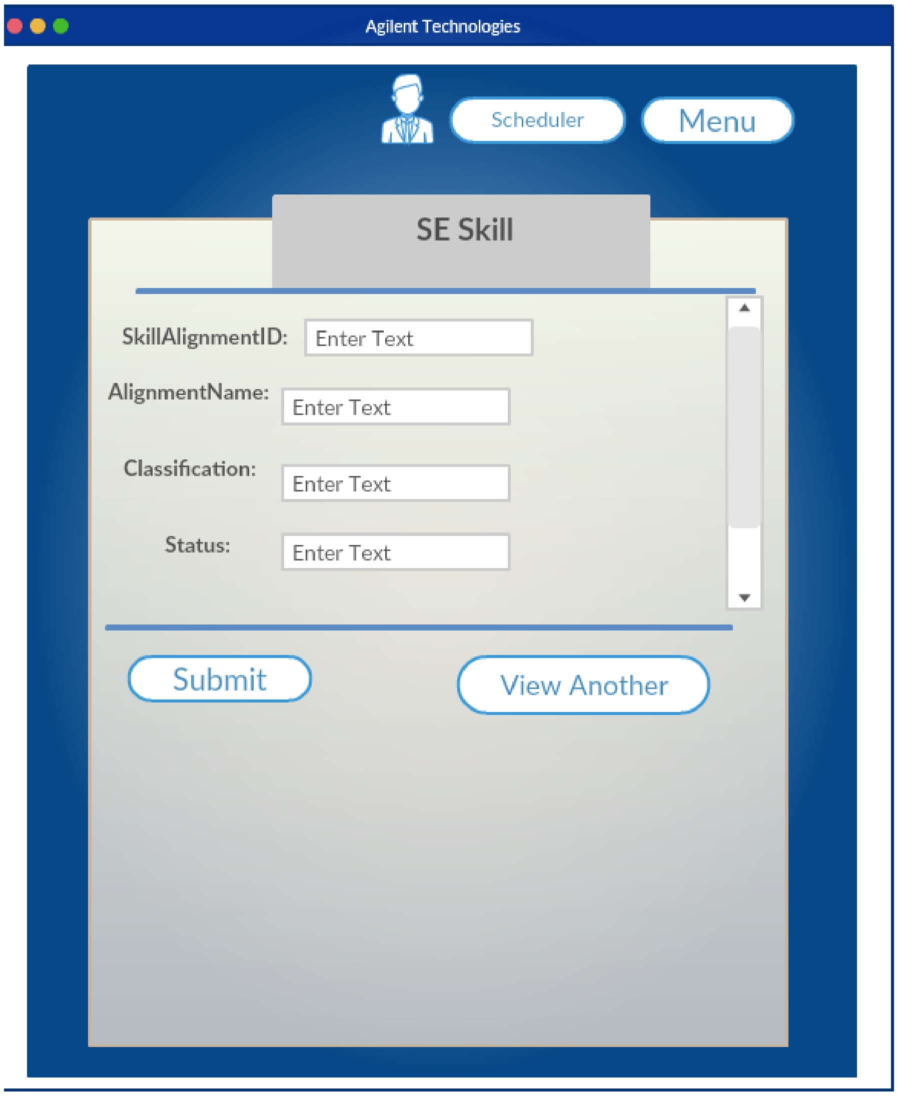

Agilent Technologies
Login
SchedulerID
Password
Login


Agilent Technologies
Agilent Technologies
Login
Menu
SE Skills

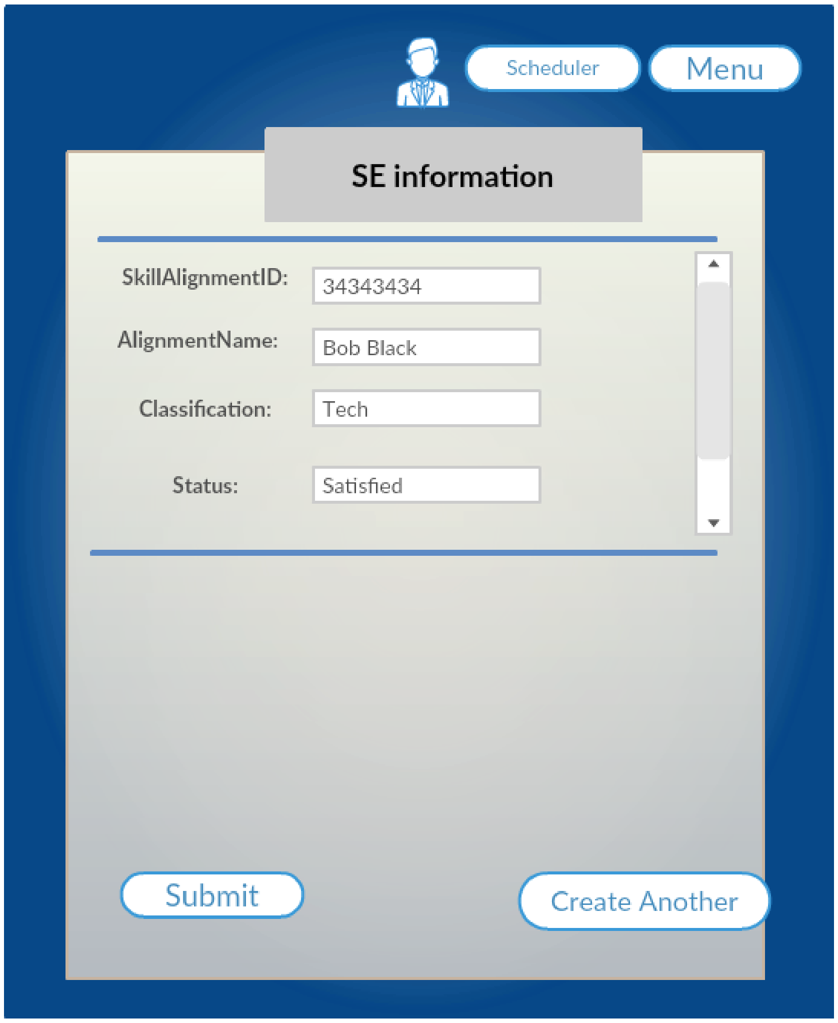


Employee

EmployeeSkillID:	Enter Text
EmployeeID:	Enter Text
AttendanceID:	Enter Text
SkillAlignmentID	Enter Text



SkillAlignmentID:	Enter Text
AlignmentName:	Enter Text
Classification:	Enter Text
Status:	Enter Text



SE information

SkillAlignmentID:	34343434
AlignmentName:	Bob Black
Classification:	Tech
Status:	Satisfied