**PROBLEM STATEMENT**

**Alumni Management System**

Many colleges maintain present and old students’ records’ manually. Recollecting this data in the manual system is very difficult. If students need to access some data about their elder students it might take a long time to go through the records and get to know about them. Hence overall collecting information about a student is a very tedious task.

Alumni Management system solves this problem by maintaining the records of old students in a database which is accessed by this software. It contains access to database and helps to create student records, edit them and maintain them and keeps it secured from others changing it.

**Software Requirement Specification**

**Table of Contents**

**1. Document Purpose**

**2. Document Overview**

**3. General Description of Project**

**3.1 Purpose of the project**

**3.2 Project context**

**3.3 Benefits**

**4. Functional Requirements**

**5. Non-Functional Requirements**

**5.1 User Interface Requirements**

**5.2 Security Requirements**

**1. Document Purpose:**

The purpose of this document is to describe the requirements of Alumni management software. This document provides the details of the software’s functional and non functional requirements and gives an overview about this project.

**2. Document Overview:**

This document contains three phases, the first giving the general description of this software product about the purpose of the project, the context and the benefits of the project.

The second chapter of this document gives details about the functional requirements which the software should meet. It also describes about the actors.

The third chapter of the document gives the non-functional aspects of this project.

**3. General Description of the project**

**3.1 Purpose of the Project**

The purpose of this software is to provide access to the alumni database to the students and the faculty. The project also lets u make entries and update the alumni info. The stored information is password protected hence the database remains accurate and the change is done by administrator hence the database remains safe.

**3.2 Project Context**

This software project is designed to have an access of the records of the students passing out of an institution or a university. It makes the data available to the students through which the can make a connection to the alumni of that particular institution.

3.3 Benefits

Using this software the data of alumni is stored in a secured database and the students and the faculty gain access to that database.

**4. Functional Requirements**

The project is divided into two modules and each module has an ability to have an access to a specific set of functions. The modules are Administrator Module and Student Module.

**Administrator Module**

1. **Creating an Entry**
2. **Insert in an existing entry**
3. **Updating an existing entry**
4. **Search for a student info**
5. **Displaying the details of students**
6. **Changing the Password**

**Student Module**

1. **Search for a student info**
2. **Displaying the details of students**

These are the functionalities of the required functions:

1. Creating an Entry: - This function creates a new entry of alumni.
2. Insert in an existing entry: - This Inserts alumni details in the same entry.
3. Updating an existing entry: - This Updates alumni details in the same entry.
4. Search for a student info: - This Search for alumni details.
5. Displaying the details of students: - This Displays alumni details.
6. Changing the Password: - This function lets the admin change the password for accessing the admin id.

**5. Non-Functional Requirements**

**5.1. USER INTERFACE REQUIREMENTS**

The user interface of the application must be user friendly, intuitive and easy to use, implementing the economics standards.

**5.2. SECURITY REQUIREMENTS**

In order to use certain features of the system, users must first authenticate themselves by name and passwords. The system shall not allow access if the user fails to provide correct login information.

Use Case Diagrams

The purpose of use case diagram is to capture the dynamic aspect of a system. However, this definition is too generic to describe the purpose, as other four diagrams (activity, sequence, collaboration, and State chart) also have the same purpose. We will look into some specific purpose, which will distinguish it from other four diagrams.

Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. Hence, when a system is analyzed to gather its functionalities, use cases are prepared and actors are identified.

When the initial task is complete, use case diagrams are modeled to present the outside view.

In brief, the purposes of use case diagrams can be said to be as follows −

* Used to gather the requirements of a system.
* Used to get an outside view of a system.
* Identify the external and internal factors influencing the system.
* Show the interaction among the requirements is actors.
* Actors can be a human user, some internal applications, or may be some external applications.
* When we are planning to draw a Use case diagram are drawn to capture the functional requirements of a system. After identifying the above items, we have to use the following guidelines to draw an efficient use case diagram
* The name of a use case is very important. The name should be chosen in such a way so that it can identify the functionalities performed.
* Give a suitable name for actors.
* Show relationships and dependencies clearly in the diagram.
* Do not try to include all types of relationships, as the main purpose of the diagram is to identify the requirements.
* Use notes whenever required to clarify some important points.

**USE CASE DIAGRAM**



*USE CASE DESCRIPTION*

(1)***Use case name***: Enter password

***Primary actor*** : Admin

***Description*** : It allows the admin to log in to the alumni system and perform operations.

***Main Flow*** : If the entered password is correct the system gives permission for the admin to perform the operations.

Alternate Flow : If the entered password is incorrect the user does not get access to perform the operations

Pre condition : The admin must have already created the password

Post condition :The admin can perform other system operations

(2)***Use case name*** : Insert

***Primary actor*** : Admin

***Description***  : It allows the admin to insert the student details in to the file.

***Main Flow*** : The admin enters the departed year, student name, roll number, branch , job status in to a new file of the alumni system.

***Alternate flow*** : If the file with the same student details is entered the details get replaced in the old file.

***Pre condition*** :The admin need to check whether there is already a same file or not.

***Post condition*** :If there is no previous file he can create a new one and enter the student details.

(3)***Use case name*** : Update

***Primary Actor*** : Admin

***Description*** : The admin will be able to update the details of the alumni.

***Main Flow*** : The admin enters the Departed year and the system checks for the availability of the file, if the file exists it searches for the student’s data in the file and updates the information of the student.

***Alternate Flow*** : After entering the Departed year if the file is not available the system will not proceed further.

***Pre Condition*** :The admin must check whether the student details are present or not

***Post condition*** :If the student details are present then he can update further details.

(4)***Use case name*** : Change Password

***Primary Actor*** : Admin

***Description*** : The admin will be able to change the password to login to the system.

***Main Flow*** : The system asks for the old password, if the entered password is correct the Admin can change the password and give a new password to login.

***Alternate Flow*** : If the old password entered is incorrect the system again asks for the correct old password and the admin can enter new password if the old password is correct.

***Pre Condition*** : The admin must know the previous old passwordin order to enter a new one

***Post condition*** : If the old password is correct he can give new password to the system.

(5)***Use case name*** : Display

***Actor***  : Admin , Alumni

***Description*** : The actors will be able to view the complete details of the student.

***Main Flow*** : The system asks for the departed year of the student if the file exists then the system gives the details of the student.

***Alternate Flow*** : If the file does not exit the system asks to enter the departed year again.

***Pre condition***  : The alumni and admin must know the departed year of the student for the system to display the details.

***Post condition*** : If the departed year is correct details are displayed.

(6)***Use case name*** : Search

***Actor*** : Admin , Alumni

***Description*** : The actors can search for the required information of the student.

***Main Flow*** : The system asks for the departed year of the student if the file exists then the system searches for the required student information and gives the details of the student.

***Alternate Flow*** : If the file does not exit the system asks to enter the departed year again.

If the file exists but the student information is not in the file the program gets terminated.

***Pre condition*** : The alumni and admin must know the departed year of the student for the system to display the details.

***Post condition*** : If the departed year is correct details are displayed.

Where to Use a Use Case Diagram?

As we have already discussed there are five diagrams in UML to model the dynamic view of a system. Now each and every model has some specific purpose to use. Actually these specific purposes are different angles of a running system.

To understand the dynamics of a system, we need to use different types of diagrams. Use case diagram is one of them and its specific purpose is to gather system requirements and actors.

Use case diagrams specify the events of a system and their flows. But use case diagram never describes how they are implemented. Use case diagram can be imagined as a black box where only the input, output, and the function of the black box is known.

These diagrams are used at a very high level of design. This high level design is refined again and again to get a complete and practical picture of the system. A well-structured use case also describes the pre-condition, post condition, and exceptions. These extra elements are used to make test cases when performing the testing.

Although use case is not a good candidate for forward and reverse engineering, still they are used in a slightly different way to make forward and reverse engineering. The same is true for reverse engineering. Use case diagram is used differently to make it suitable for reverse engineering.

In forward engineering, use case diagrams are used to make test cases and in reverse engineering use cases are used to prepare the requirement details from the existing application.

Use case diagrams can be used for −

* Requirement analysis and high level design.
* Model the context of a system.
* Reverse engineering.
* Forward engineering.

Purpose of Activity Diagrams :

The basic purpose of activity diagrams is similar to other four diagrams. It captures the dynamic behavior of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part.

It does not show any message flow from one activity to another. Activity diagram is sometimes considered as the flowchart. Although the diagrams look like a flowchart, they are not. It shows different flows such as parallel, branched, concurrent, and single.

The purpose of an activity diagram can be described as −

* Draw the activity flow of a system.
* Describe the sequence from one activity to another.
* Describe the parallel, branched and concurrent flow of the system.

***ACTIVITY DIAGRAM & DISCRIPTION***

(1) LOGIN :

Normal Flow Alternate Flow

* Normal Flow :
* The alumni/admin enters the password
* If the entered password is correct the system gives access to alumni/admin to perform further operations
* Alternate Flow :
* The system checks if the entered password is valid or not if not it asks the alumni/admin to enter password again
* If the password entered is correct then the user gets access to the system

***2. Inserting student details***:

*Normal Flow Exception Flow*

* Normal Flow :
* The students departed year is entered
* Student details are entered
* If the data is not present in the database the details are entered
* Alternate Flow :
* The students departed year is entered
* Student details are entered
* If the data is present in the database the details are not entered again

***(3)Displaying student details:***

* *

*Normal Flow Exception Flow*

* ***Normal Flow :***
* The students departed year is entered
* The availability of file is checked
* If the student file is present the student details are displayed
* ***Alternate Flow :***
* The students departed year is entered
* The availability of file is checked
* If the student file is not present it is not displayed and to display the information it must be inserted first

***(4)Searching Student Data:***

******  

*Normal Flow Exception Flow*

* ***Normal Flow :***
* Departed year of the student is entered
* The availability of the file is checked
* If the file is present the required data is searched
* If the required data is available the details are printed.
* ***Alternate Flow :***
* Departed year of the student is entered
* The availability of the file is checked if the file does not exist the user cannot proceed further
* If the file is present the required data is searched
* If the required data is not available the execution stops.

***(5)Updating student job status:***

 **

*Normal Flow Exception Flow*

* ***Normal flow :***
* Departed year of the student is entered
* The availability of the file is checked
* If the file is present the required data is searched
* If the required data is available the details are updated.
* ***Alternate Flow :***
* Departed year of the student is entered
* The availability of the file is checked if the file does not exist the user cannot proceed further
* If the file is present the required data is searched
* If the required data is not available the students details cannot be updated.

***(6)Change Login Password:***

** 

*Normal Flow* *Exception Flow*

* Normal Flow :
* The admin enters the old password
* If the entered password is correct the system gives access to admin to enter new password.
* Alternate Flow :
* The system checks if the entered password is valid or not if not it asks the admin to enter password again
* If the password entered is correct then admin can enter the new password.

Where to Use Activity Diagrams?

The basic usage of activity diagram is similar to other four UML diagrams. The specific usage is to model the control flow from one activity to another. This control flow does not include messages.

Activity diagram is suitable for modeling the activity flow of the system. An application can have multiple systems. Activity diagram also captures these systems and describes the flow from one system to another. This specific usage is not available in other diagrams. These systems can be database, external queues, or any other system.

We will now look into the practical applications of the activity diagram. From the above discussion, it is clear that an activity diagram is drawn from a very high level. So it gives high level view of a system. This high level view is mainly for business users or any other person who is not a technical person.

This diagram is used to model the activities which are nothing but business requirements. The diagram has more impact on business understanding rather than on implementation details.

Activity diagram can be used for −

* Modeling work flow by using activities.
* Modeling business requirements.
* High level understanding of the system's functionalities.

The Sequence Diagram

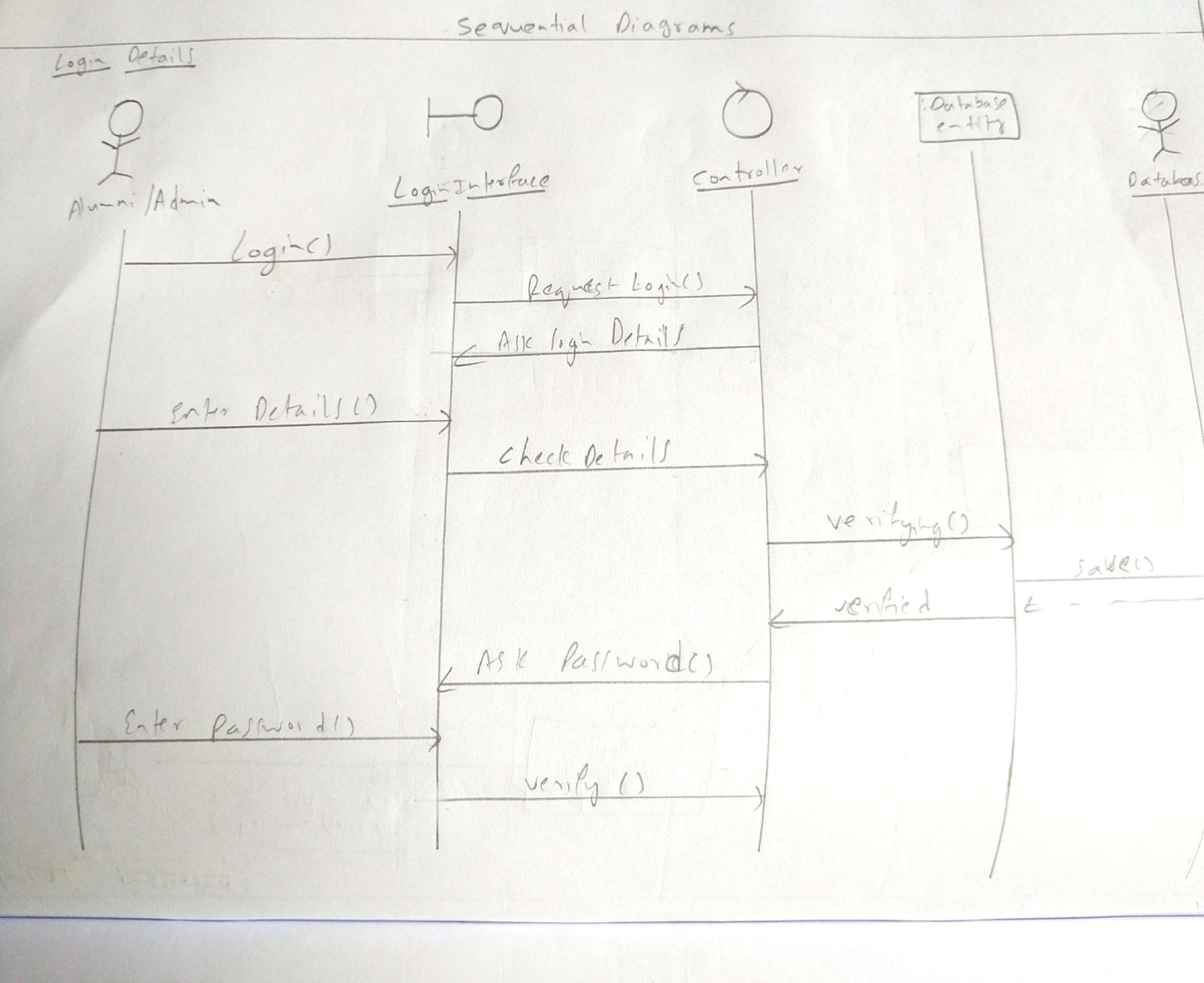
The sequence diagram has four objects (Customer, Order, Special Order and Normal Order).

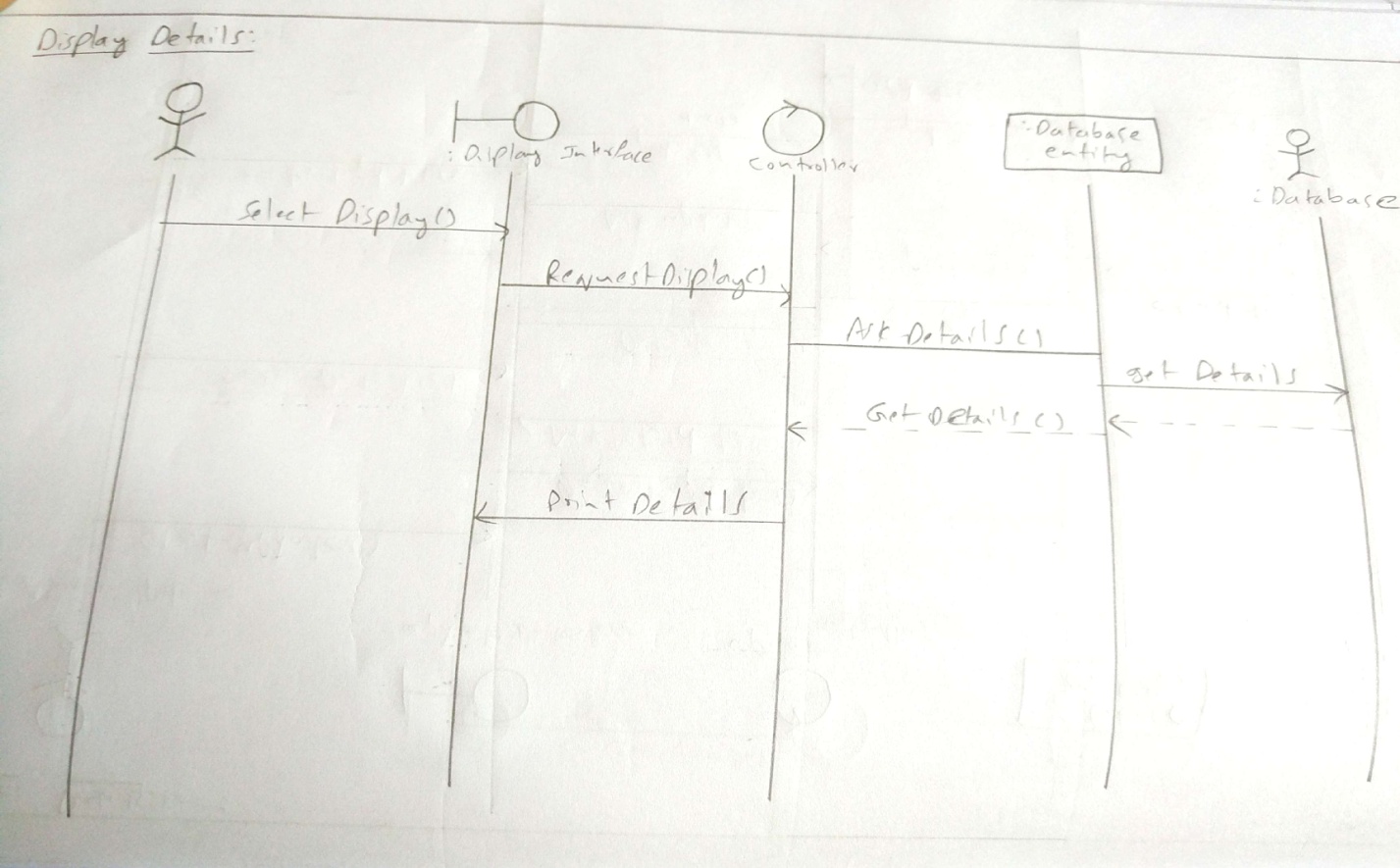
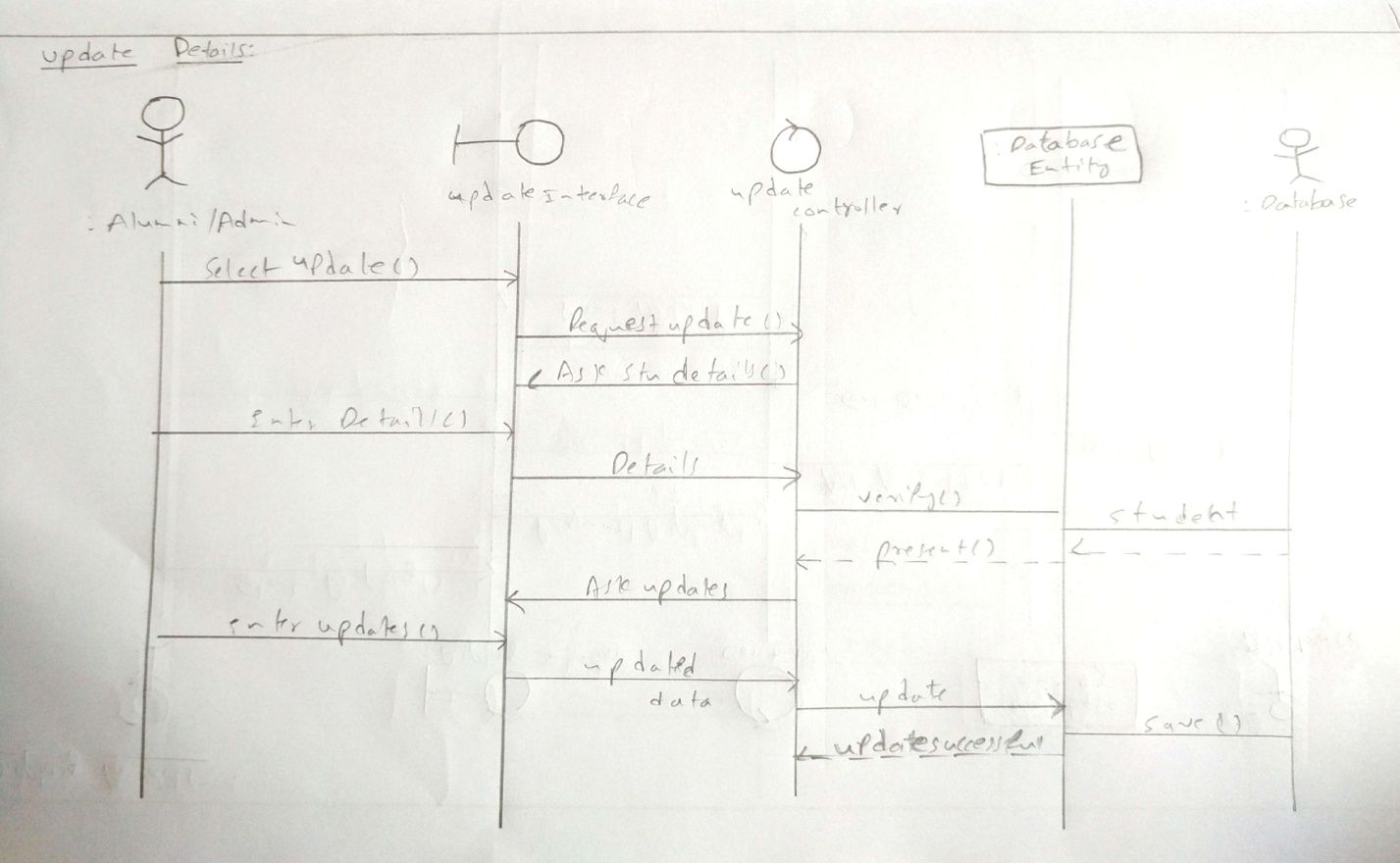
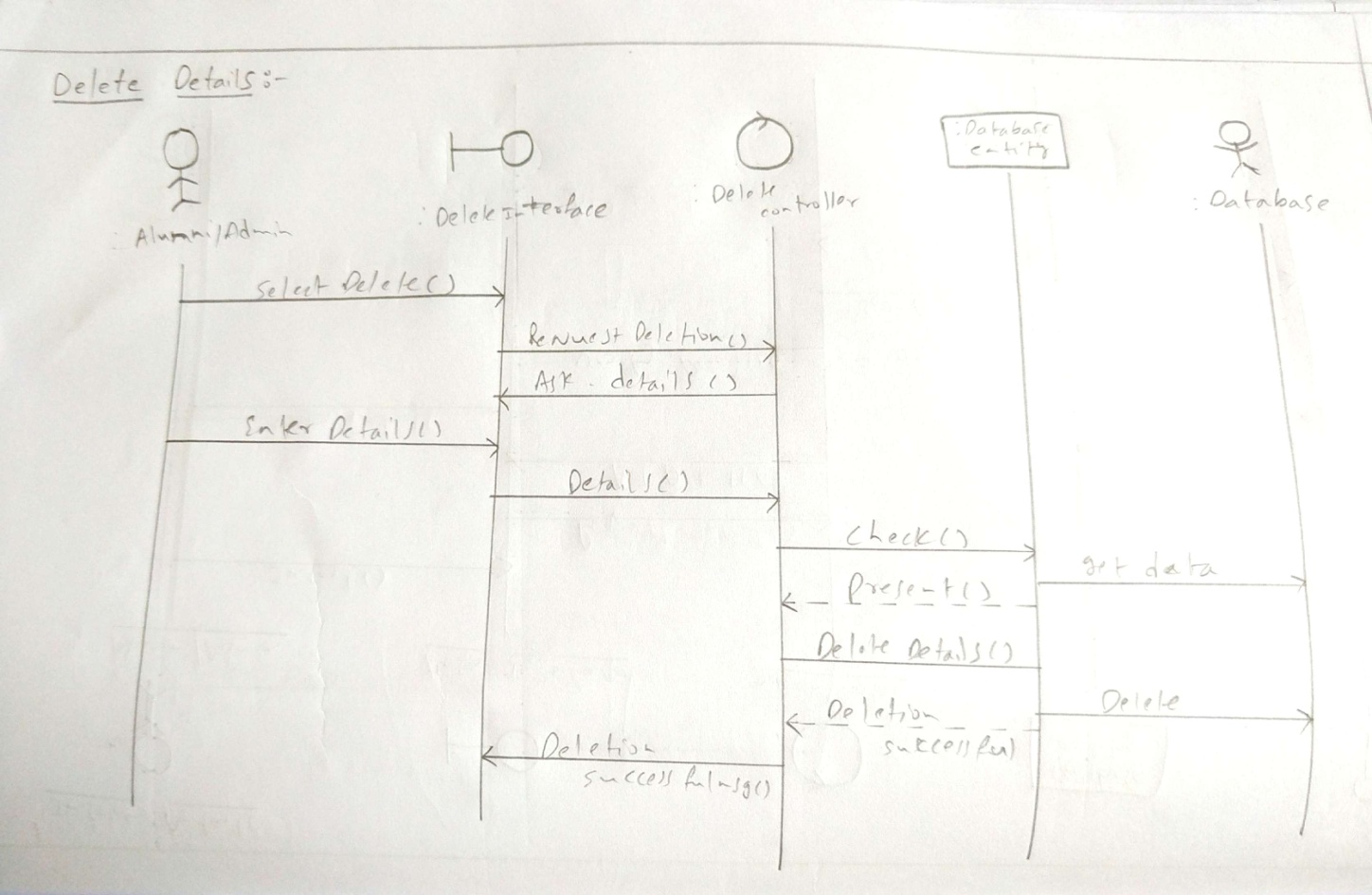
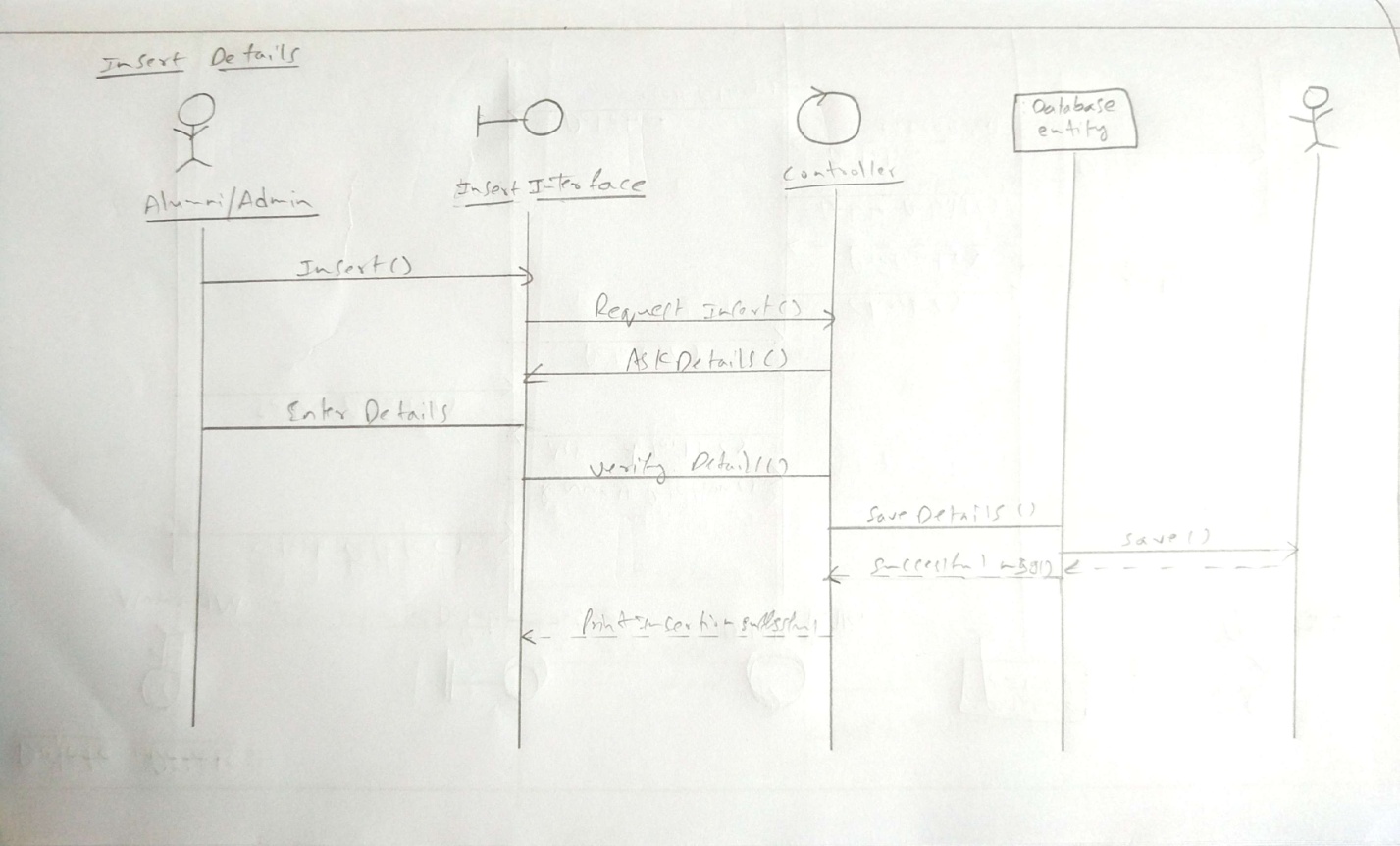
The following diagram shows the message sequence for Special Order object and the same can be used in case of Normal Order object. It is important to understand the time sequence of message flows. The message flow is nothing but a method call of an object.

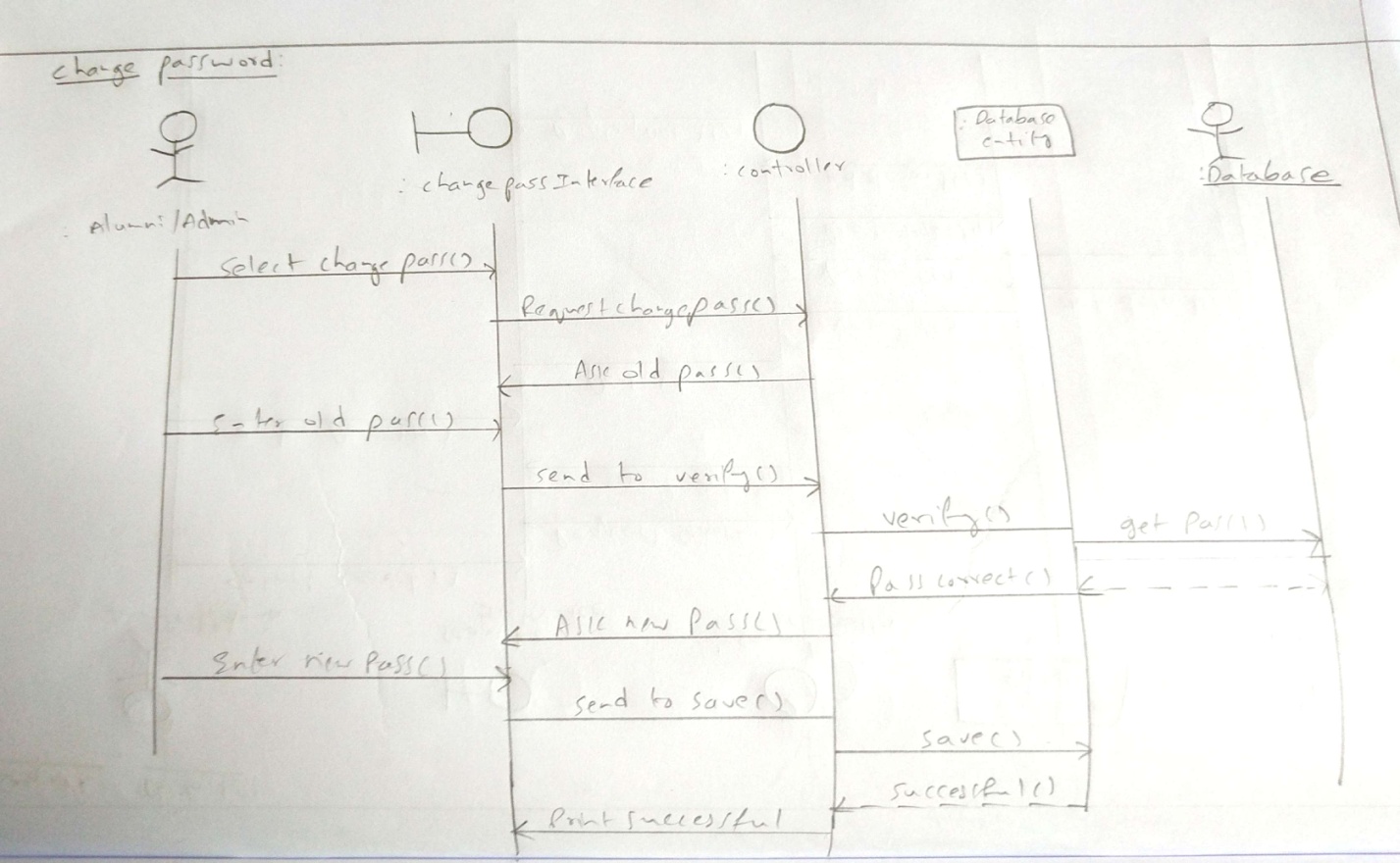
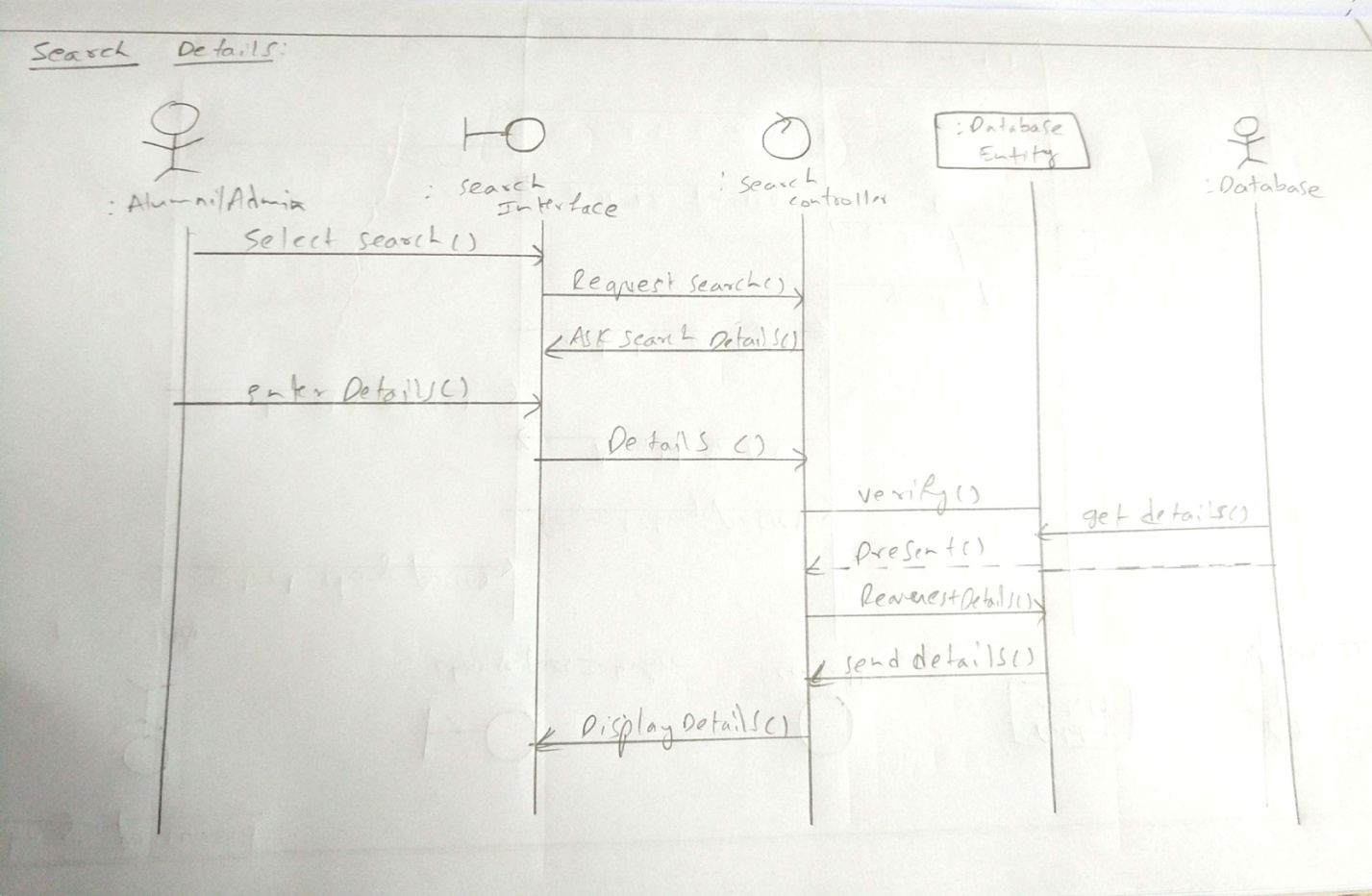
The first call is send Order () which is a method of Order object. The next call is confirm () which is a method of Special Order object and the last call is Dispatch () which is a method of Special Order object. The following diagram mainly describes the method calls from one object to another, and this is also the actual scenario when the system is running.

Deployment diagrams are used to visualize the topology of the physical components of a system, where the software components are deployed.

Deployment diagrams are used to describe the static deployment view of a system. Deployment diagrams consist of nodes and their relationships.





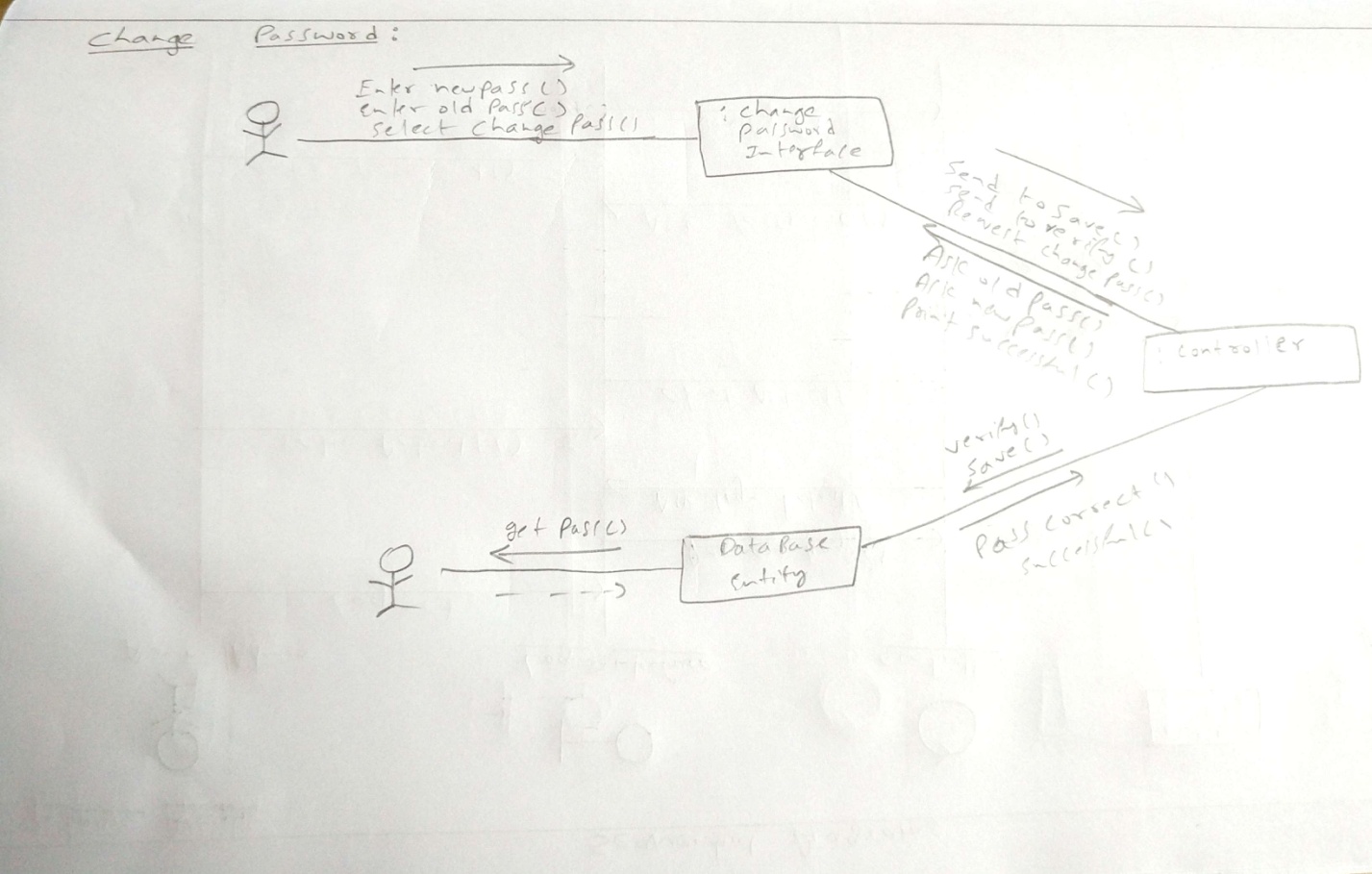
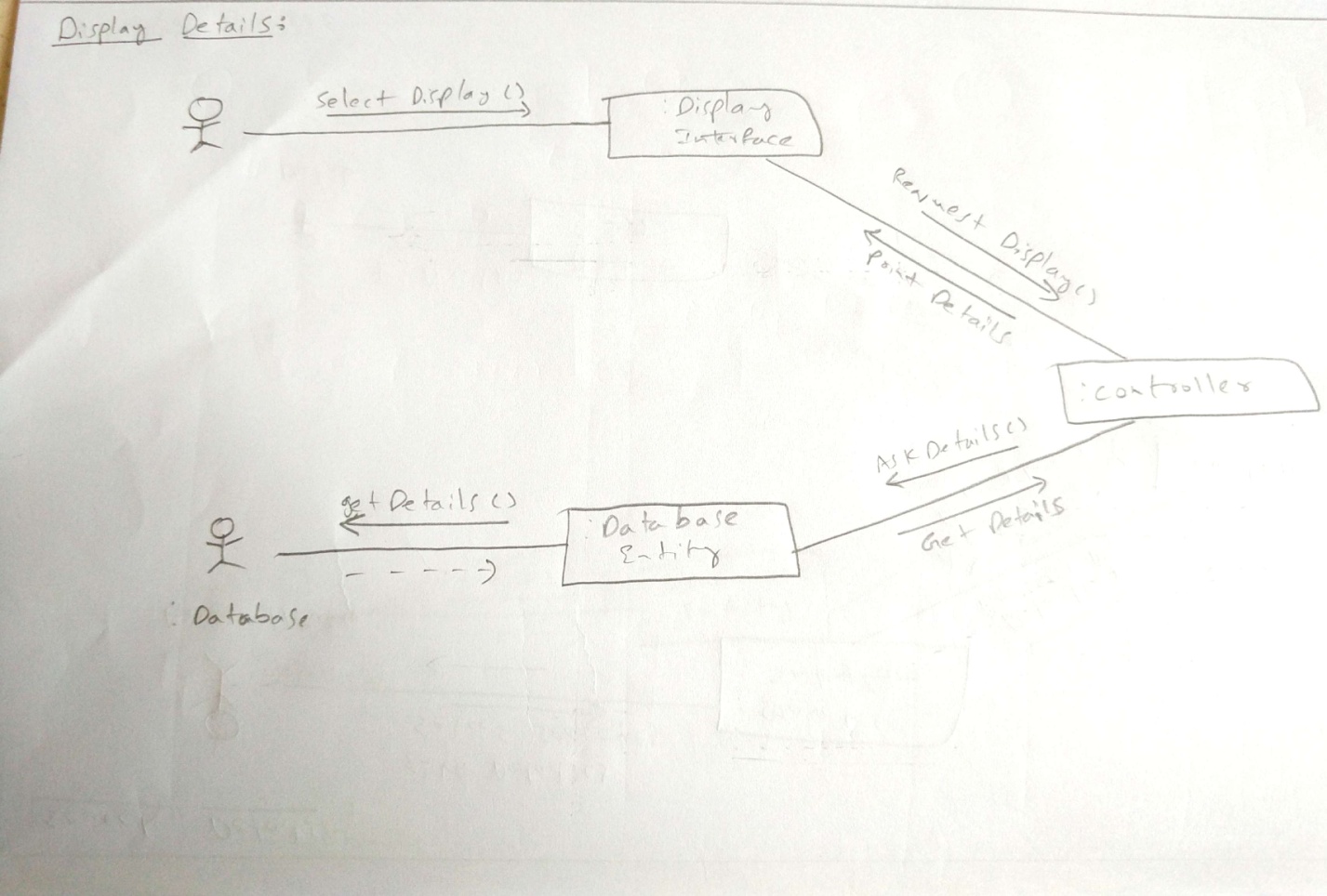
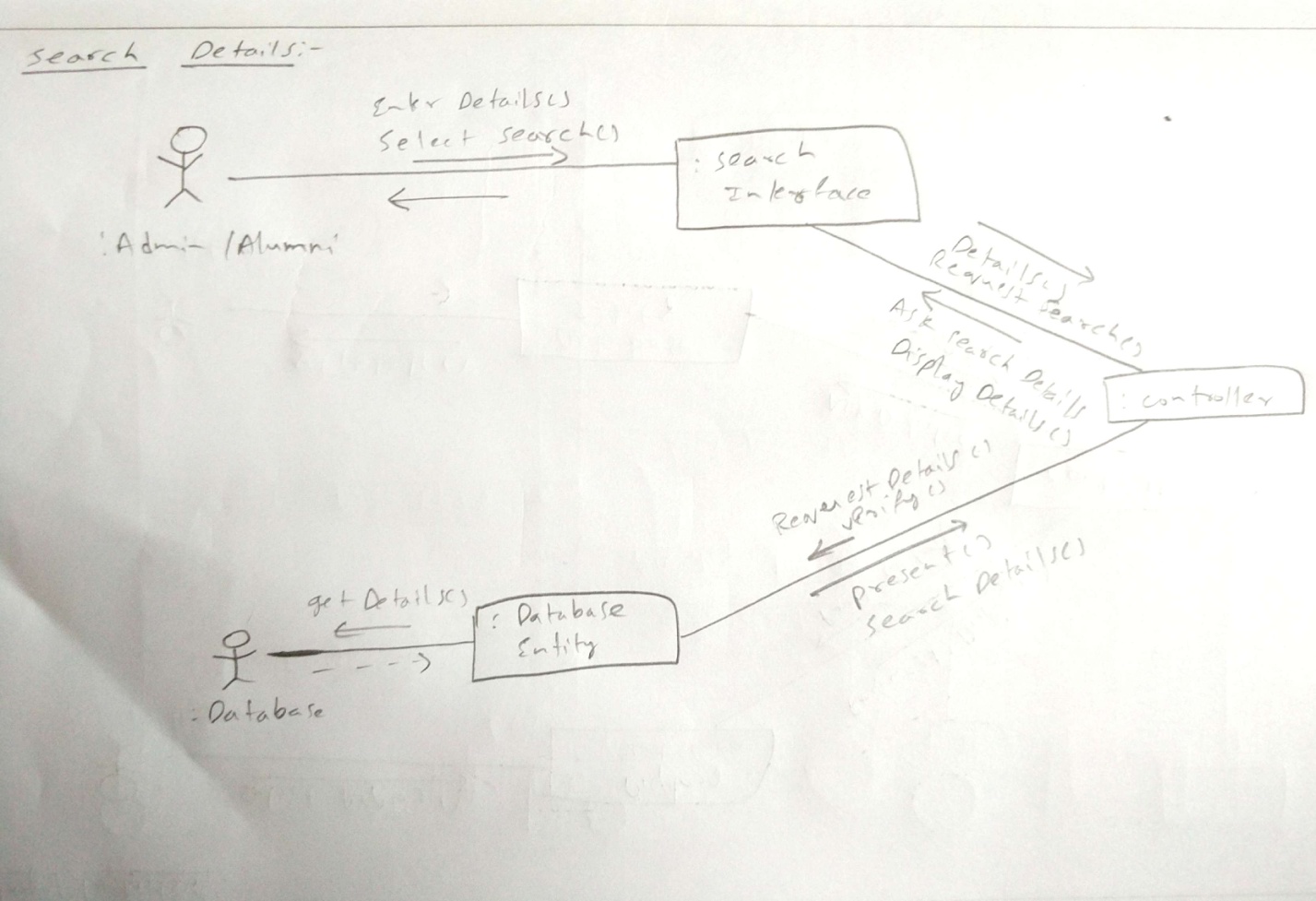
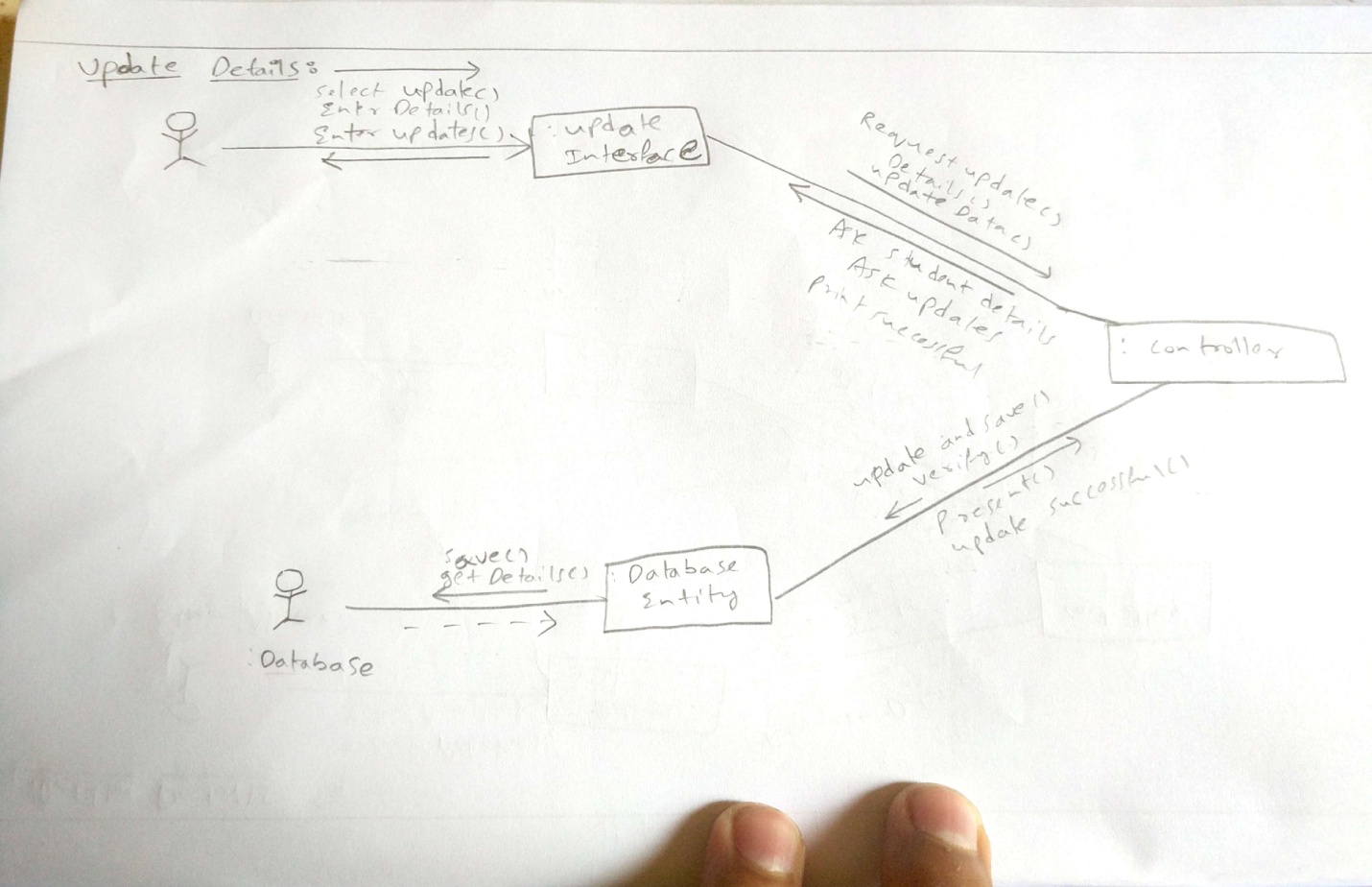
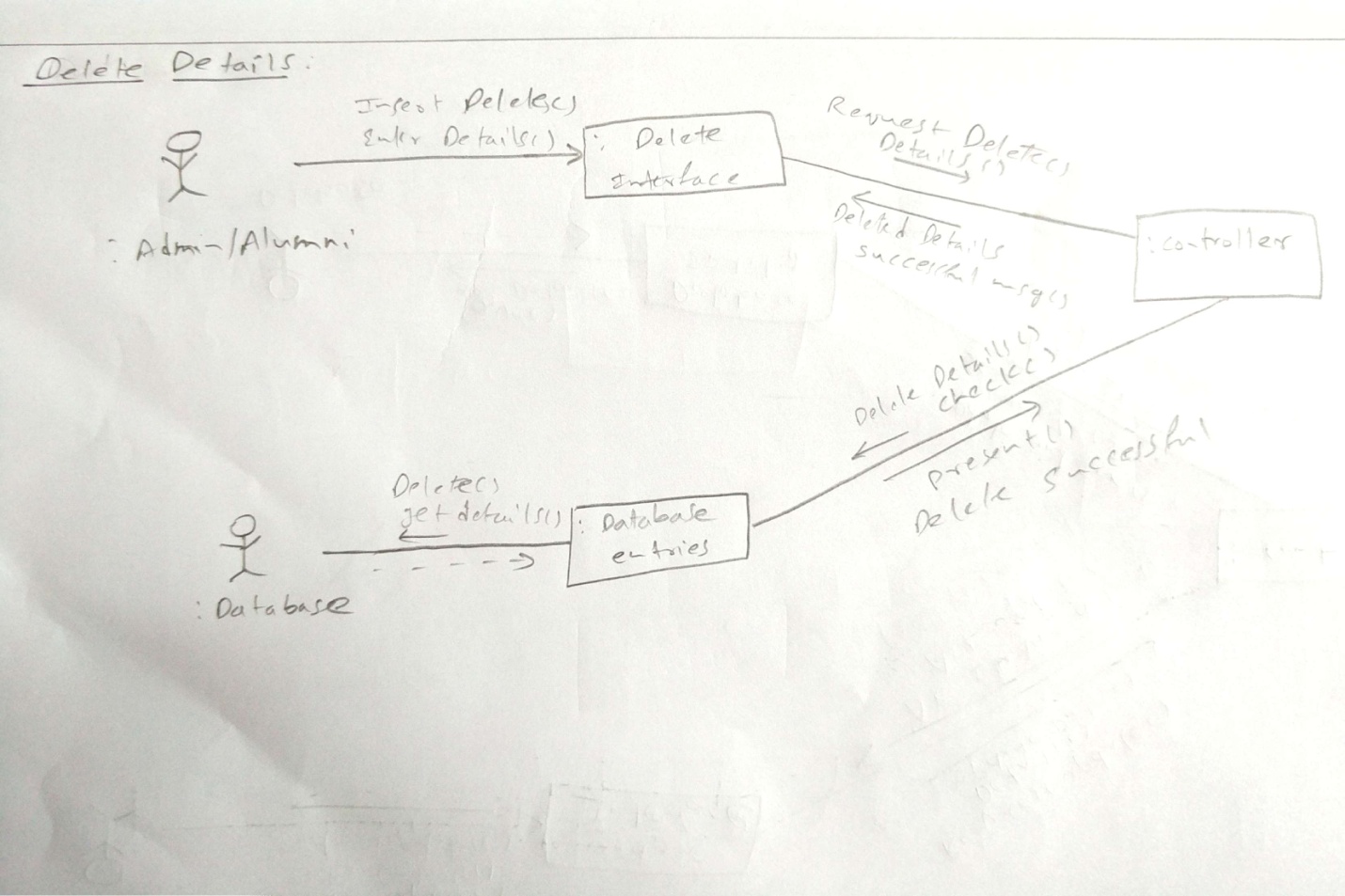
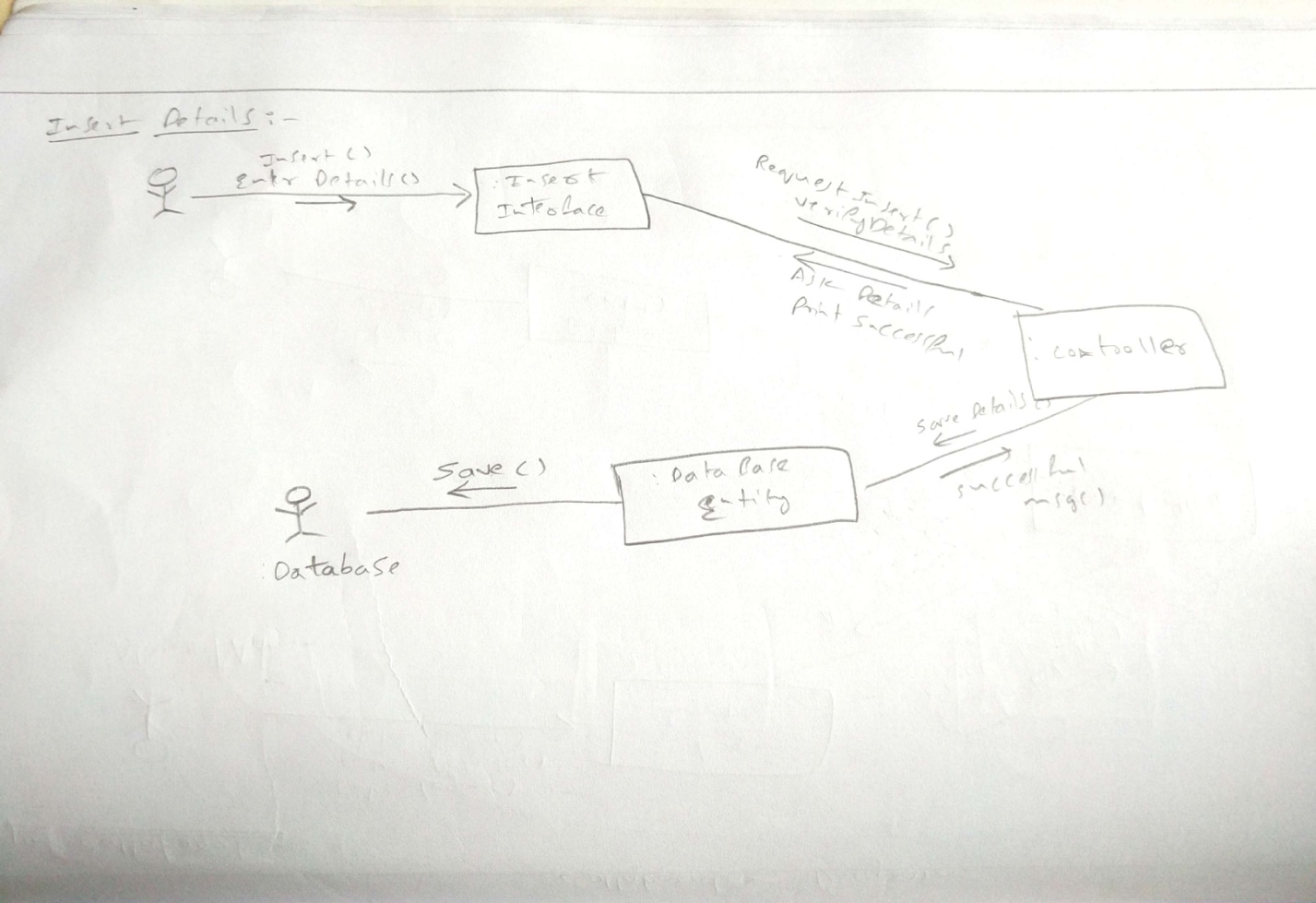
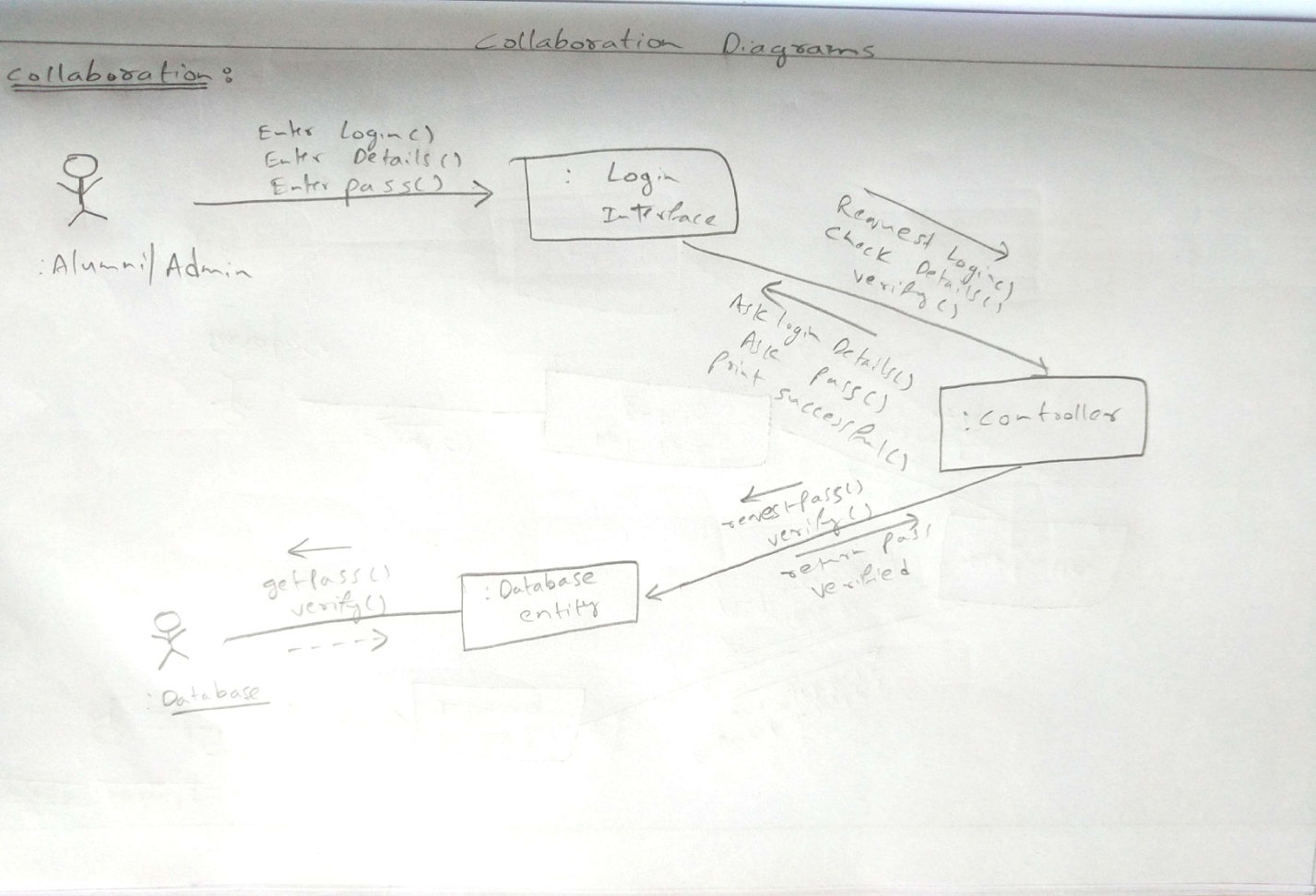


**The Collaboration Diagram**

The second interaction diagram is the collaboration diagram. It shows the object organization as seen in the following diagram. In the collaboration diagram, the method call sequence is indicated by some numbering technique. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram.

Method calls are similar to that of a sequence diagram. However, difference being the sequence diagram does not describe the object organization, whereas the collaboration diagram shows the object organization.

To choose between these two diagrams, emphasis is placed on the type of requirement. If the time sequence is important, then the sequence diagram is used. If organization is required, then collaboration diagram is used.



**Class Diagrams**

Class diagrams are widely used to describe the types of objects in a system and their relationships.

Class diagrams model class structure and contents using design elements such as classes, packages

and objects. Class diagrams describe three different perspectives when designing a system,

conceptual, specification, and implementation. These perspectives become evident as the diagram is

created and help solidify the design. This example is only meant as an introduction to the UML and

class diagrams. Classes are composed of three things: a name, attributes, and operations.

Below is an example of a class.

Class diagrams also display relationships such as containment, inheritance, associations and others.

Below is an example of an associative relationship:

The association relationship is the most common relationship in a class diagram. The association

shows the relationship between instances of classes. For example, the class Order is associated with

the class Customer. The multiplicity of the association denotes the number of objects that can

participate in then relationship. For example, an Order object can be associated to only one customer,

but a customer can be associated to many orders. Another common relationship in class diagrams is a

generalization. A generalization is used when two classes are similar, but have some differences.

In this example the classes Corporate Customer and Personal Customer have some similarities such

as name and address, but each class has some of its own attributes and operations. The class

Customer is a general form of both the Corporate Customer and Personal Customer classes. This

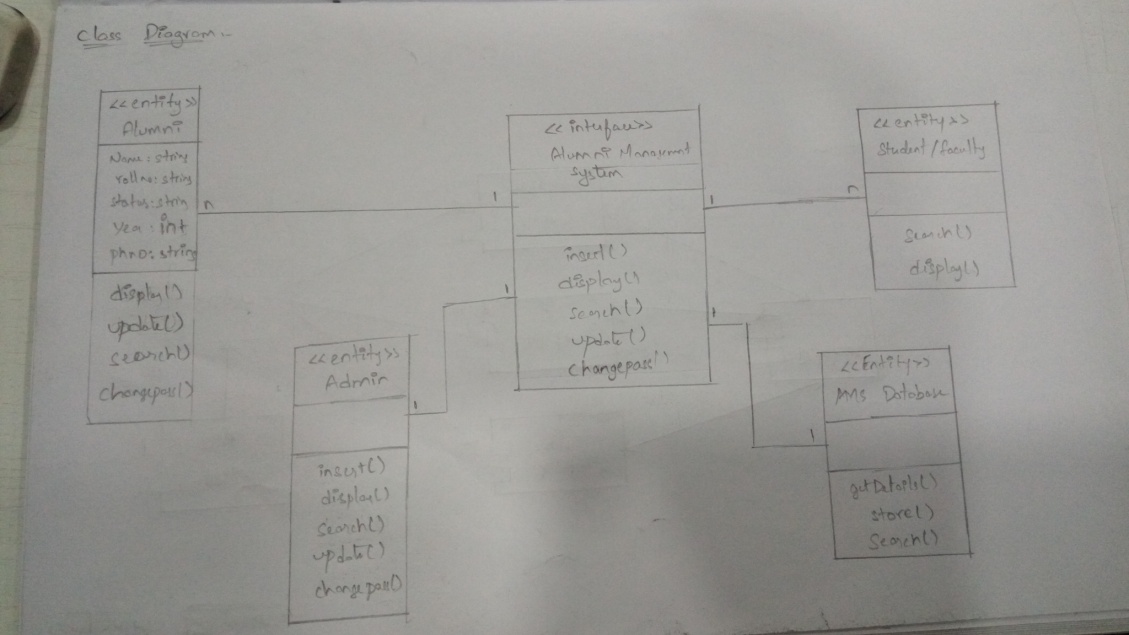
allows the designers to just use the Customer class for modules and do not require in-depth

Representation of each type of customer.

**When to Use: Class Diagrams**

Class diagrams are used in nearly all Object Oriented software designs. Use them to describe the

Classes of the system and their relationships to each other.



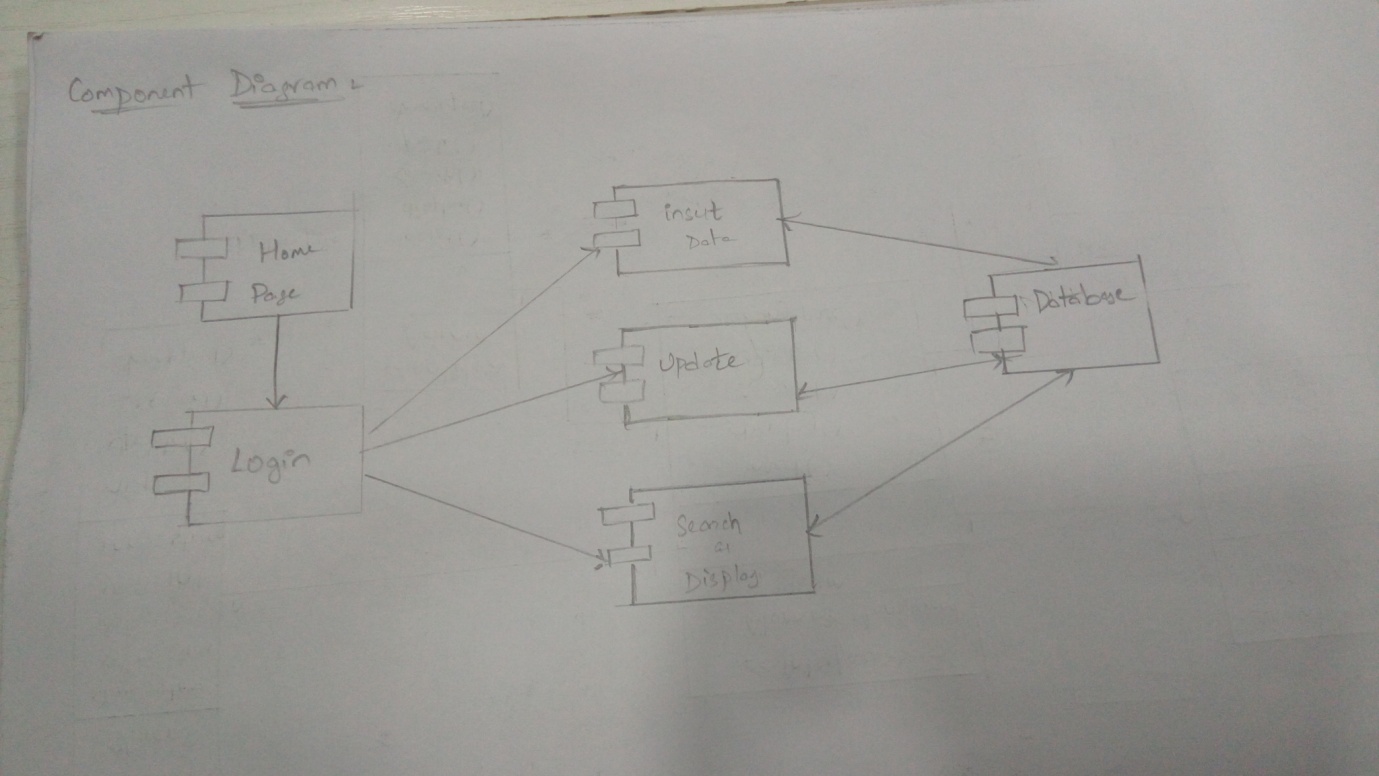
**Component Diagram**

It displays the high level packaged structure of the code itself.

Dependencies among components are shown, including source code components, binary code

components, and executable components. Some components exist at compile time, at link

time, at run times well as at more than one time.



**Deployment Diagrams**

The term Deployment itself describes the purpose of the diagram. Deployment diagrams are used for describing the hardware components, where software components are deployed. Component diagrams and deployment diagrams are closely related.

Component diagrams are used to describe the components and deployment diagrams shows how they are deployed in hardware.

UML is mainly designed to focus on the software artifacts of a system. However, these two diagrams are special diagrams used to focus on software and hardware components.

Most of the UML diagrams are used to handle logical components but deployment diagrams are made to focus on the hardware topology of a system. Deployment diagrams are used by the system engineers.

The purpose of deployment diagrams can be described as −

* Visualize the hardware topology of a system.
* Describe the hardware components used to deploy software components.
* Describe the runtime processing nodes.

