**Assignment 3**

**By: Sohal Patel**

Team: Value Added

**March 26, 2020**

Table of Contents

[Grants Management System 4](#_Toc36067078)

[Use Case 1: Find Information 4](#_Toc36067079)

[Use Case Description – 1 4](#_Toc36067080)

[Use Case Diagram - 1 6](#_Toc36067081)

[Use Case Narrative - 1 6](#_Toc36067082)

[Prototype - 1 7](#_Toc36067083)

[Class Diagram - 1 8](#_Toc36067084)

[Class Diagram Narrative - 1 8](#_Toc36067085)

[Sequence Diagram – 1 9](#_Toc36067086)

[Use Case 2: Create Grants 10](#_Toc36067087)

[Use Case Description – 2 10](#_Toc36067088)

[Use Case Diagram – 2 12](#_Toc36067089)

[Use Case Narrative – 2 12](#_Toc36067090)

[Prototype – 2 13](#_Toc36067091)

[Class Diagram – 2 14](#_Toc36067092)

[Class Diagram Narrative – 2 14](#_Toc36067093)

[Sequence Diagram - 2 15](#_Toc36067094)

[Use Case 3: Modify Grants 16](#_Toc36067095)

[Use Case Description – 3 16](#_Toc36067096)

[Use Case Diagram – 3 19](#_Toc36067097)

[Use Case Narrative – 3 19](#_Toc36067098)

[Prototype – 3 20](#_Toc36067099)

[Class Diagram – 3 21](#_Toc36067100)

[Class Diagram Narrative – 3 21](#_Toc36067101)

[Sequence Diagram - 3 22](#_Toc36067102)

[Use Case 4: Delete Grants 23](#_Toc36067103)

[Use Case Description – 4 23](#_Toc36067104)

[Use Case Diagram – 4 26](#_Toc36067105)

[Use Case Narrative – 4 26](#_Toc36067106)

[Prototype – 4 27](#_Toc36067107)

[Class Diagram – 4 28](#_Toc36067108)

[Class Diagram Narrative – 4 28](#_Toc36067109)

[Sequence Diagram -4 29](#_Toc36067110)

[Use Case 5 - Create Student Application 30](#_Toc36067111)

[Use Case Description – 5 30](#_Toc36067112)

[Use Case Diagram – 5 32](#_Toc36067113)

[Use Case Narrative – 5 32](#_Toc36067114)

[Prototype – 5 33](#_Toc36067115)

[Class Diagram – 5 34](#_Toc36067116)

[Class Diagram Narrative – 5 34](#_Toc36067117)

[Sequence Diagram - 5 35](#_Toc36067118)

[Use Case 6 – Login User 36](#_Toc36067119)

[Use Case Diagram – 6 36](#_Toc36067120)

[Use Case Narrative - 6 36](#_Toc36067121)

# Grants Management System

# Use Case 1: Find Information

## Use Case Description – 1

**Use Case Specification: Find Information**

**Use-Case Name: Find Information**

**Brief Description**

This use case allows the student to find information from the Grants Management System. The system is designed to help them applying for the grant.

**Flow of Events**

**Basic Flow**

1. The student selects "Find Grant Information" interface from the menu options.
2. The system displays "Find Grant Information" interface.
3. The student enters search criterion of the required information.
4. The system displays student’s required information on the interface.
5. The student views his required information on the interface.

**Alternative Flows**

**First Alternative Flow**

3.1 The system does not find any record for the given search criterion.

3.2 The system displays “No Record Found”.

**An Alternative Subflow**

N.A.

**Second Alternative Flow**

N.A.

**Special Requirements**

N.A.

**First Special Requirement**

N.A.

**Pre-conditions**

**Pre-condition One**

The student has browsed the website from internet.

**Post-conditions**

Post-condition One

The system has successfully displayed the student’s required information.

**Extension Points**

N.A.

**Name of Extension Point**

N.A.

**System Requirements**

**Operational**

1. The system operates on Windows 7, 8 & 10, Linux 2005 Minimum, or MAC OS 10.0.

2. The system has a 2.8 GHz processor with 4 GB of free disk space.

4. The system has an internet connection for software activation.

5. The system is integrated with the existing University of Louisville Research database

6. The system also works on mobile devices.

**Performance**

1. The system runs 24 hours per day, 365 days per year.

2. The system supports the capacity of the University of Louisville Research Database.

3. The system handles 100 transactions per hour.

**Security**

1. The system allows users to see their transaction history.

2. The system encrypts all transactions.

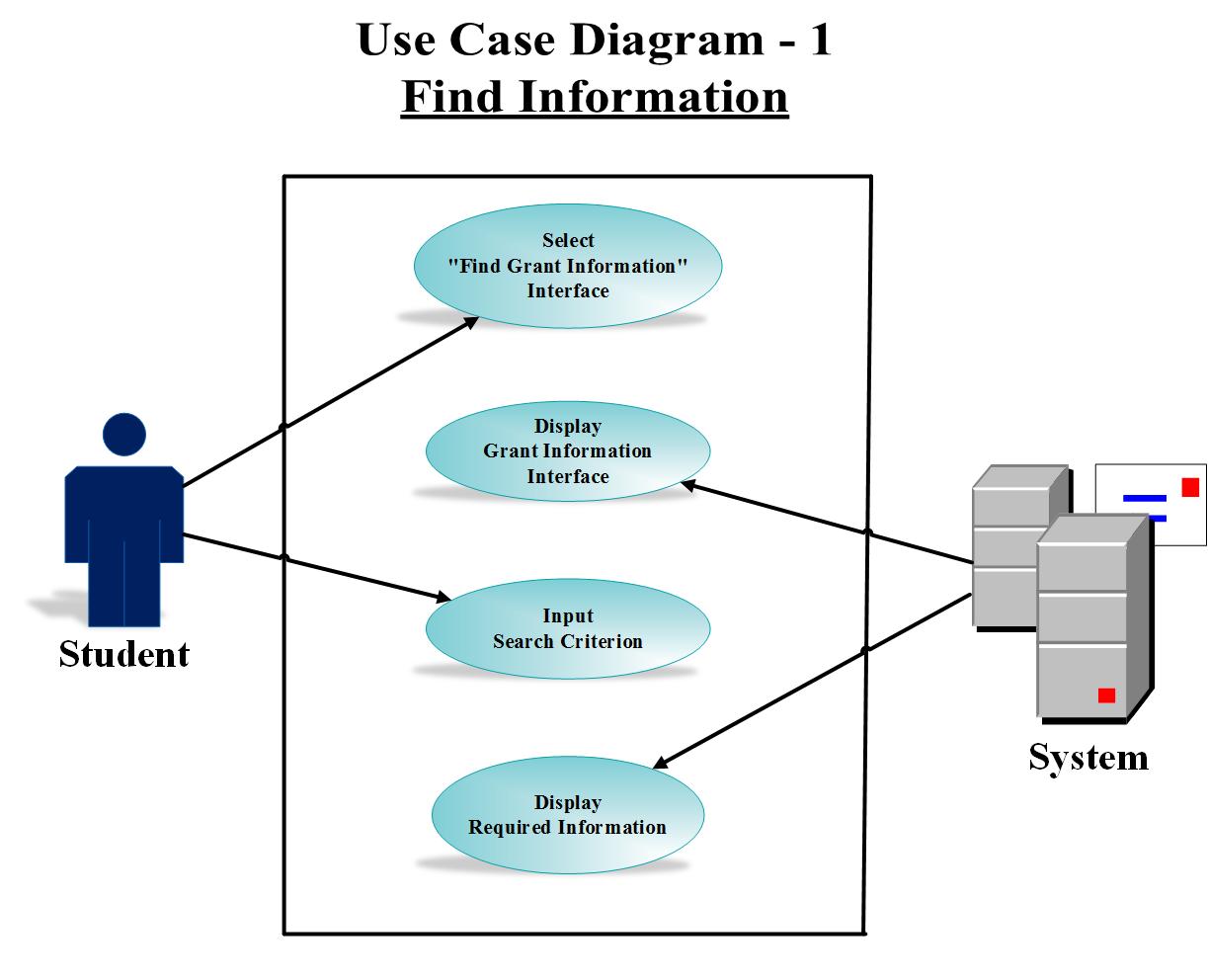
**Functional**

1. The system displays “Find Information” interface to the user.
2. The system displays all required information to the user.

**Trace Matrix**

The system is user friendly for all the users using it specially the student who is the main actor of our use case “find Information”. The system allows student to view the grants information once the student enters the search criteria.

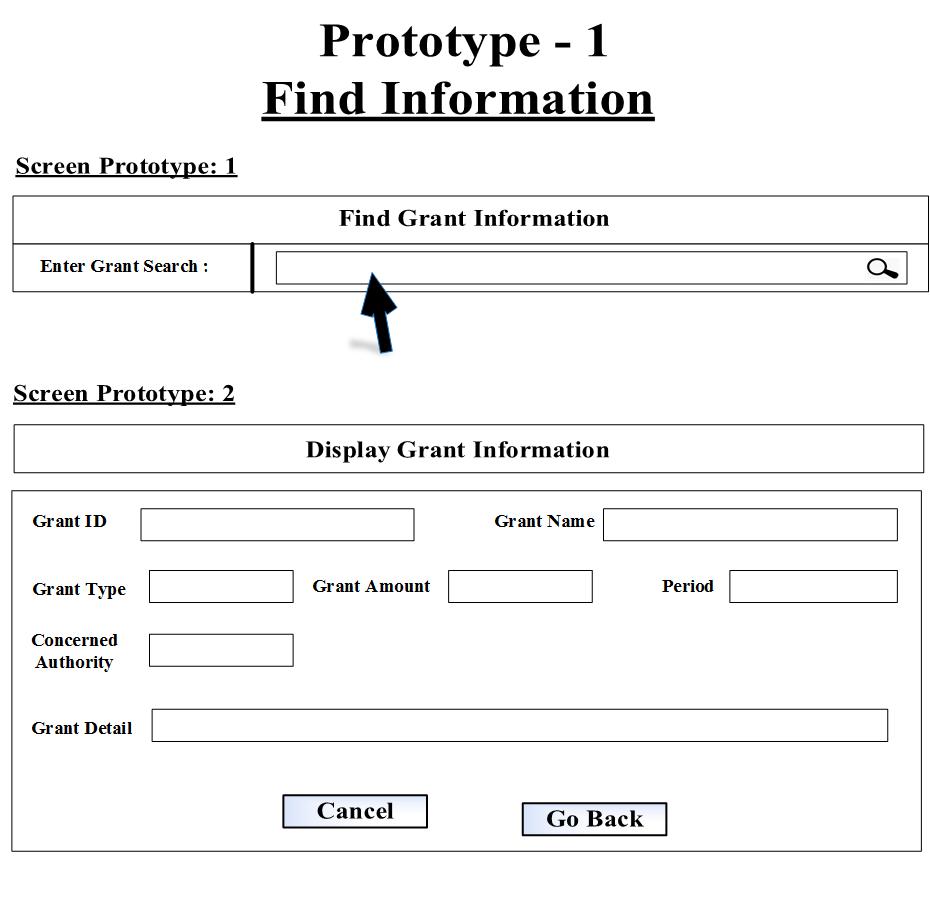
## Use Case Diagram - 1



## Use Case Narrative - 1

This use case provides the client a flow of actions through which the main actor student requires to find the grant information. The student opts to find information interface and the system displays the interface. The use case requires the student to input search criterion. The student inputs search criteria and the system display the required information.

## Prototype - 1

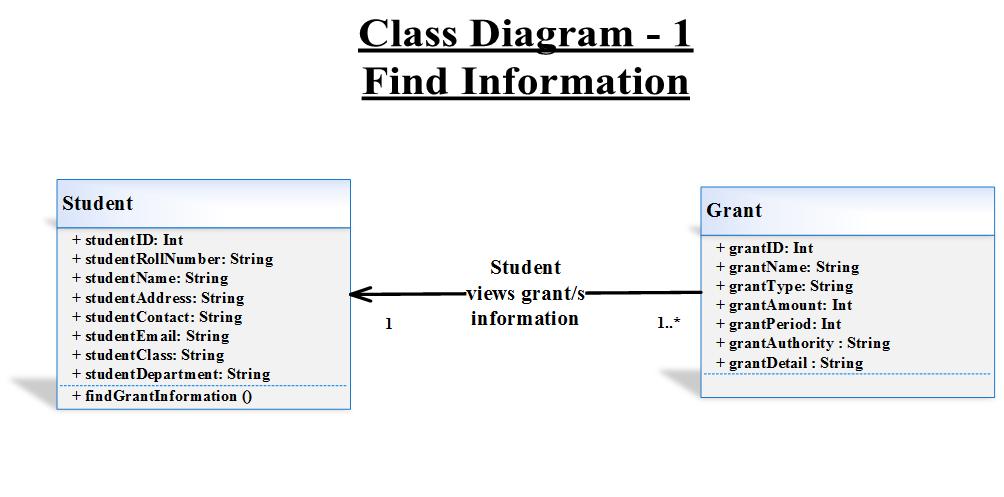


Url: <http://pictureintext.net/pro/use/Design-1/>

Or

CLICK HERE TO SEE THE PROTOTYPE - 1 IN ACTION

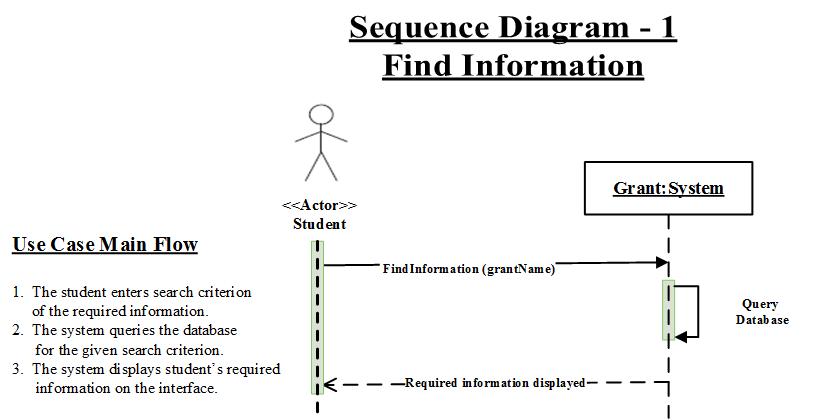
## Class Diagram - 1



## Class Diagram Narrative - 1

The find information class diagram explains the client two classes: Student and Grant. The student class holds student’s attributes and the grant class holds grants attributes. The client after viewing class model will become aware of the specific information and the associated methods regarding find information use case. The student class interacts with grants class to find information having one to many multiplicity relations. Both classes have association relationship with each other.

## Sequence Diagram – 1



# Use Case 2: Create Grants

## Use Case Description – 2

**Use Case Specification: Create Grants**

**Use-Case Name: Create Grants**

**Brief Description**

This use case allows the admin to create new grants in the Grants Management System. The university students, then, will be able to apply for the grants.

**Flow of Events**

**Basic Flow**

1. The admin selects "Create Grants" option from the menu options.
2. The system displays " Create Grants" interface.
3. The admin enters all information of the new grant.
4. The admin presses “Submit” button.
5. The system saves a new grant record in the database.

**Alternative Flows**

**First Alternative Flow**

4.1. The admin presses “Cancel” button.

4.2. The system prompts “Confirm Cancel” dialog box with “Yes” and “No” options.

**An Alternative Subflow**

4.2.1. The admin presses “Yes” button.

4.2.2. The system clears all information from the interface.

**An Alternative Subflow**

4.2.3. The admin presses “No” button.

4.2.4. The system takes no action.

**Second Alternative Flow**

N.A.

**Special Requirements**

N.A.

**Pre-conditions**

**Pre-condition One**

The admin has received all information of the new grant from the external actor “Industry Partners”.

**Post-conditions**

**Post-condition One**

The system is ready to modify the created grant.

**Extension Points**

N.A.

**Name of Extension Point**

N.A.

**System Requirements**

**Operational**

1. The system operates on Windows 7, 8 & 10, Linux 2005 Minimum, or MAC OS 10.0.

2. The system has a 2.8 GHz processor with 4 GB of free disk space.

4. The system has an internet connection for software activation.

5. The system is integrated with the existing University of Louisville Research database.

6. The system also works on mobile devices.

**Performance**

1. The system runs 24 hours per day, 365 days per year.

2. The system supports the capacity of the University of Louisville Research Database.

3. The system handles 100 transactions per hour.

**Security**

1. The system encrypts all grant transactions.

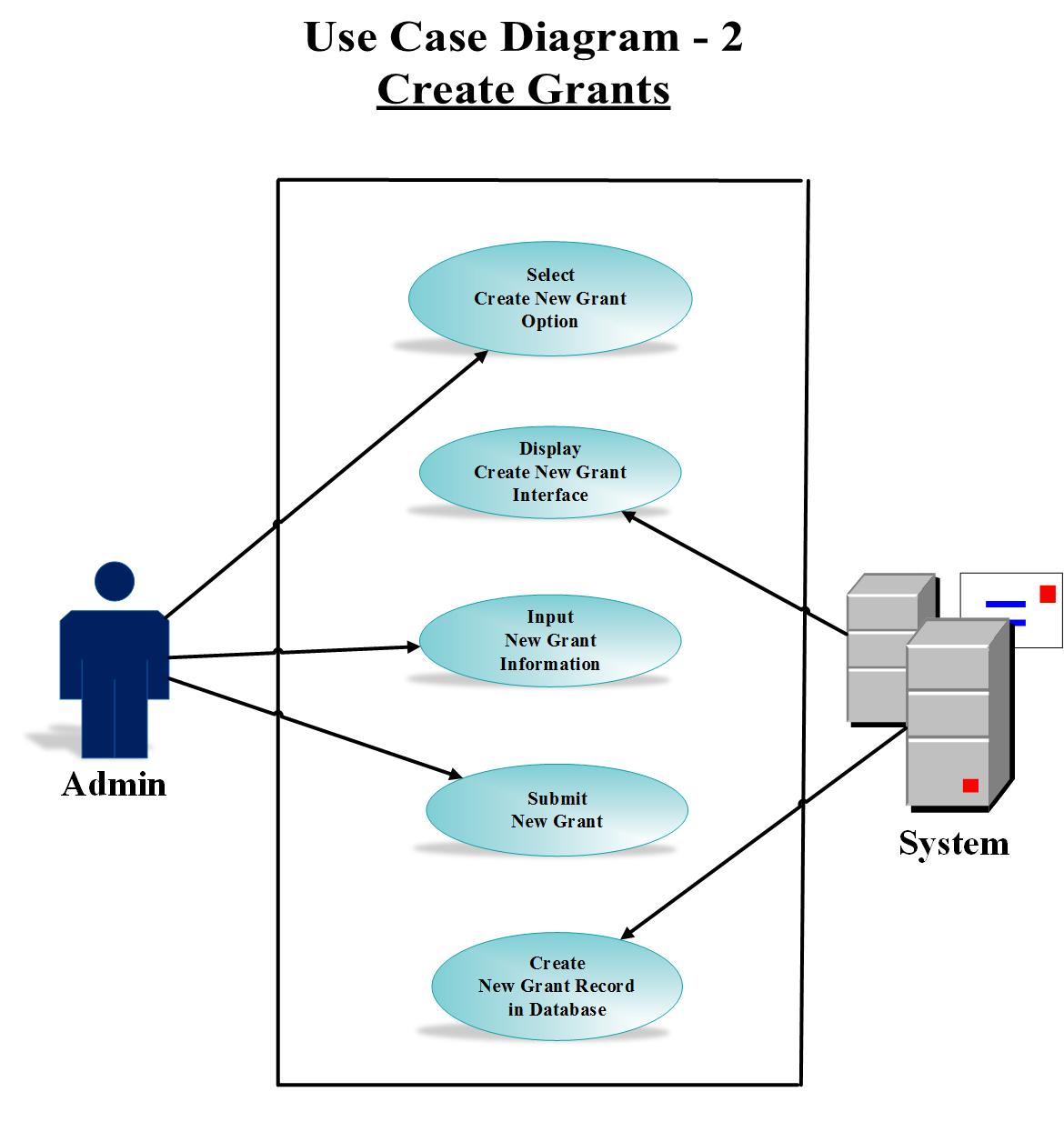
**Functional**

1. The system displays “Create Grants” interface to the user.
2. The system successfully creates new grants in the database.

**Trace Matrix**

The system is user friendly for all the users including admin who is responsible for creating grants using the “Create Grants” Interface. The system allows admin to add all the grants details and submit it.

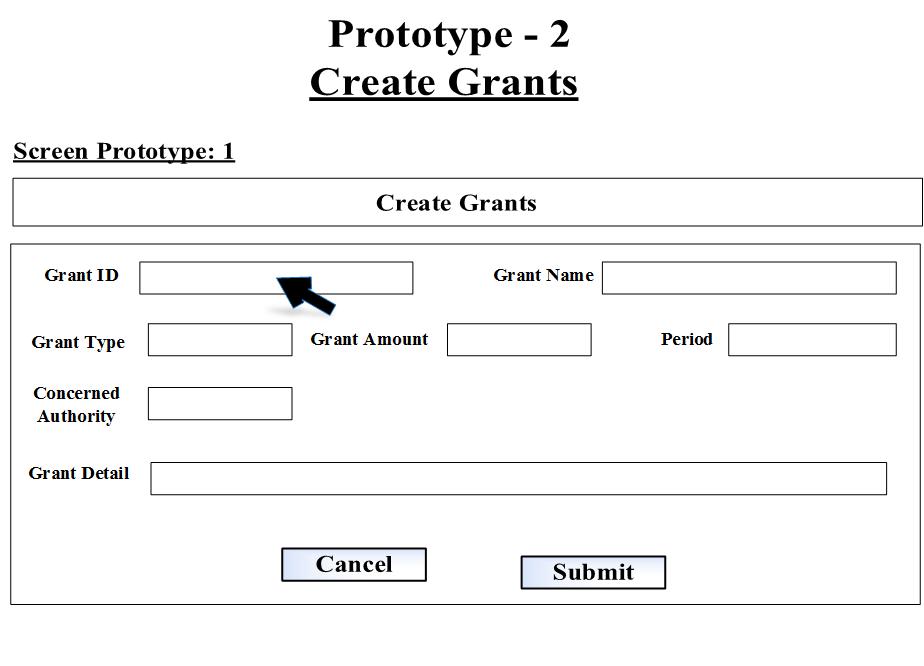
## Use Case Diagram – 2



## Use Case Narrative – 2

This use case provides the client a flow of actions through which the actor admin involves adding a new grant. The admin opts to create a new grant interface and the system displays the interface. The use case requires the admin to input grant information and submit them. The system then creates a new grant record in the database.

## Prototype – 2

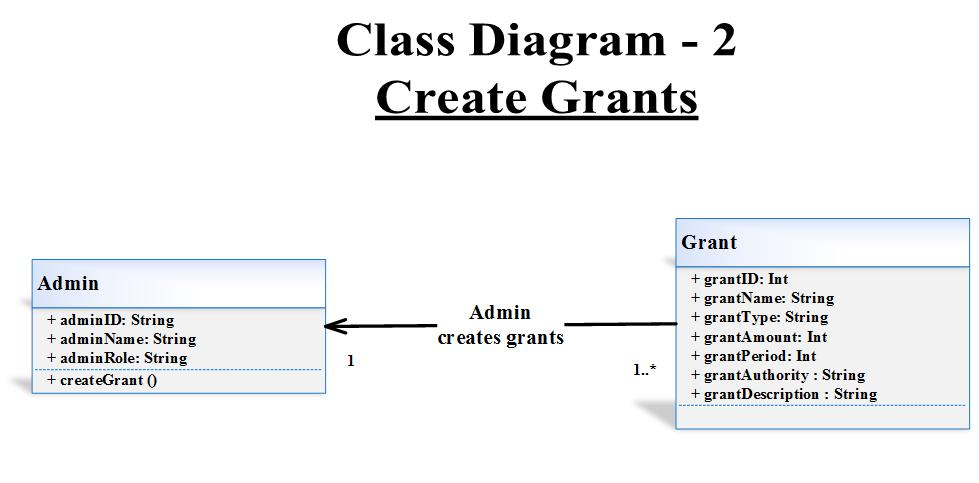


Url: <http://pictureintext.net/pro/use/Design-2/>

Or

CLICK HERE TO SEE THE PROTOTYPE - 2 IN ACTION

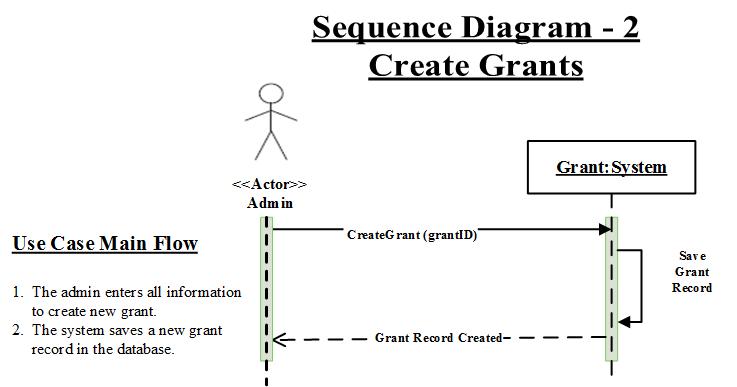
## Class Diagram – 2



## Class Diagram Narrative – 2

The create grants class diagram explains the client two classes: Admin and Grant. The admin class holds admin’s attributes and the grant class holds grants attributes. The client after seeing this class model will develop awareness of the detailed attributes and the associated method regarding create grants use case. The admin class interacts with grants class to create new grant having one to many multiplicity relations. Both classes have association relationship with each other.

## Sequence Diagram - 2



# Use Case 3: Modify Grants

## Use Case Description – 3

**Use Case Specification: Modify Grants**

**Use-Case Name: Modify Grants**

**Brief Description**

This use case allows the admin to modify already existing grants in the Grants Management System. The university students, then, will be able to apply on the modified grant.

**Flow of Events**

**Basic Flow**

1. The admin selects "Modify Grants" option from the menu options.
2. The system displays "Modify Grants" interface.
3. The admin enters grant ID to access the grant information.
4. The system displays the grant information.
5. The admin changes grant information.
6. The admin presses “Modify” button.
7. The system updates the grant record in the database with modifications.

**Alternative Flows**

**First Alternative Flow**

6.1. The admin presses “Cancel” button.

6.2. The system prompts “Confirm Cancel” dialog box with “Yes” and “No” options.

**An Alternative Subflow**

6.2.1. The admin presses “Yes” button.

6.2.2. The system clears the screen.

**An Alternative Subflow**

6.2.3. The admin presses “No” button.

6.2.4. The system clears all information from the interface.

**Second Alternative Flow**

**Special Requirements**

N.A.

**First Special Requirement**

N.A.

**Pre-conditions**

**Pre-condition One**

The admin has created a grant in the system.

**Post-conditions**

**Post-condition One**

The users can apply on the modified grant.

**Extension Points**

N.A.

**Name of Extension Point**

N.A.

**System Requirements**

**Operational**

1. The system operates on Windows 7, 8 & 10, Linux 2005 Minimum, or MAC OS 10.0.

2. The system has a 2.8 GHz processor with 4 GB of free disk space.

4. The system has an internet connection for software activation.

5. The system is integrated with the existing University of Louisville Research database.

6. The system also works on mobile devices.

**Performance**

1. The system runs 24 hours per day, 365 days per year.

2. The system supports the capacity of the University of Louisville Research Database.

3. The system handles 100 transactions per hour.

**Security**

1. The system encrypts all modified grant transactions.

**Functional**

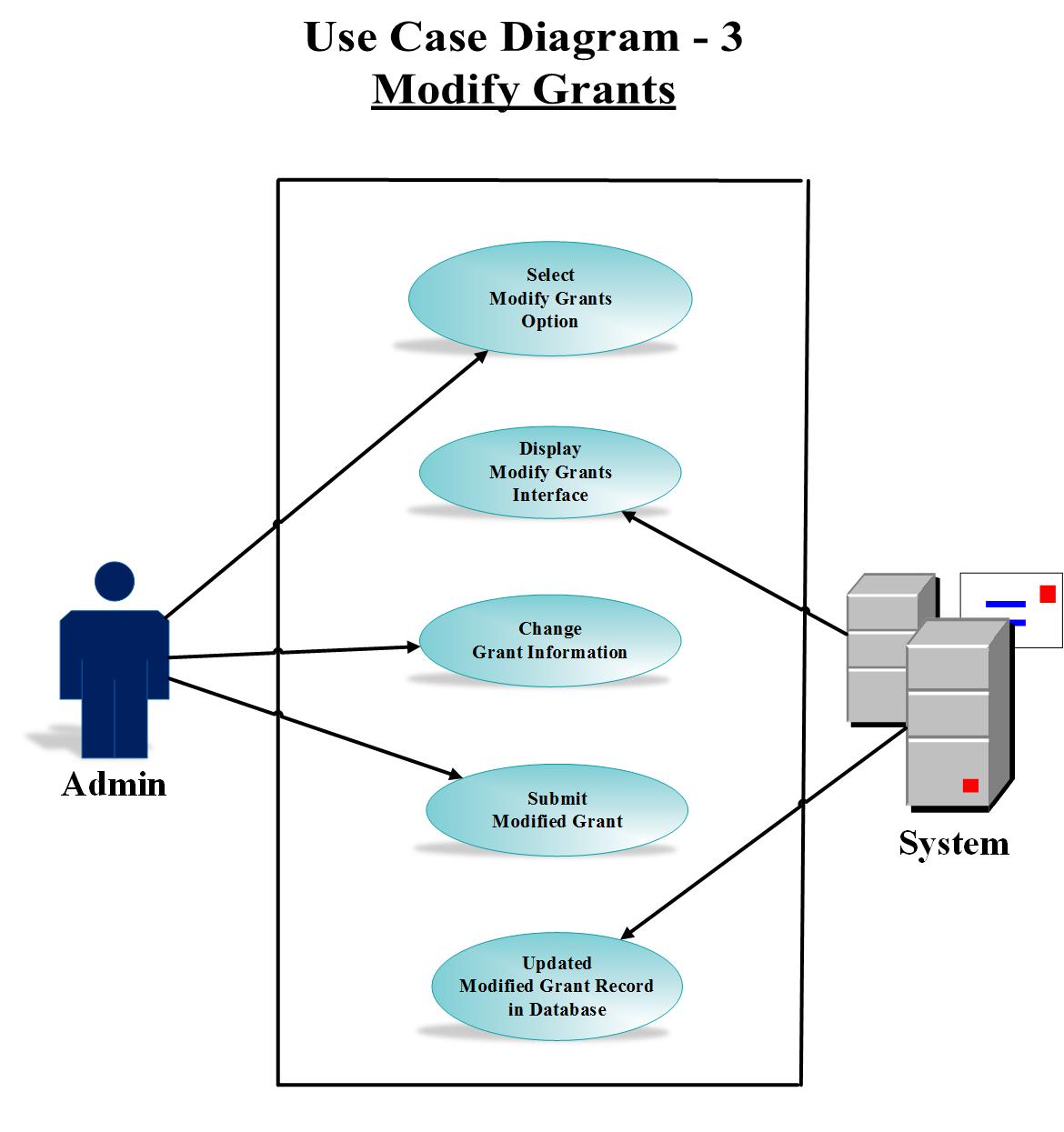
1. The system displays “Modify Grants” interface to the user.

1. The system successfully modifies grants in the database.

**Trace Matrix**

The system is user approachable for all the users including admin who is the main actor for modifying grants using the “Modify Grants” Interface. The system allows admin to make the required changes of the grant information and save them permanently.

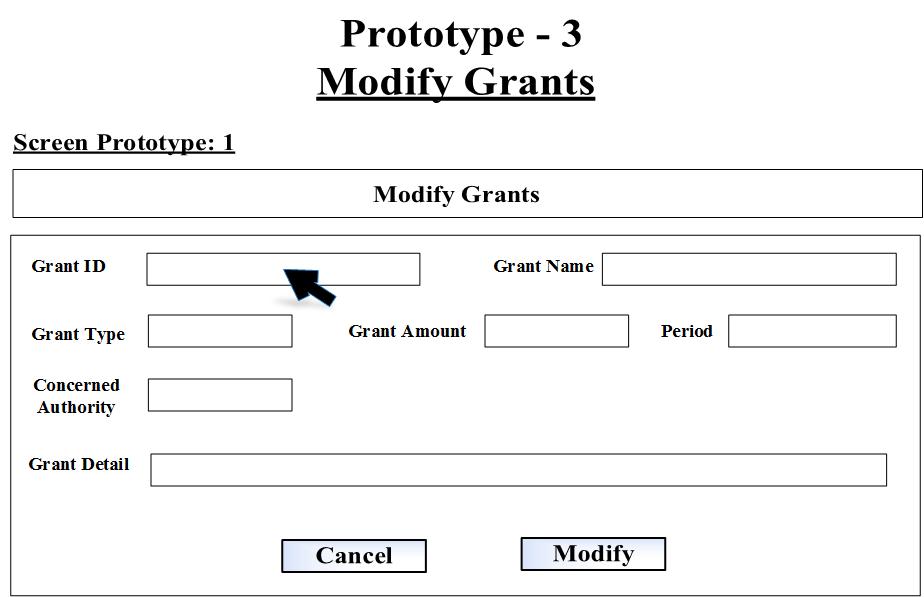
## Use Case Diagram – 3



## Use Case Narrative – 3

This use case provides the client a flow of actions through which the actor admin involves modifying an already existing grant. The admin opts to modify grant interface and the system displays the interface. The use case requires the admin to modify grant information and submit them. The system then modifies grant record in the database.

## Prototype – 3

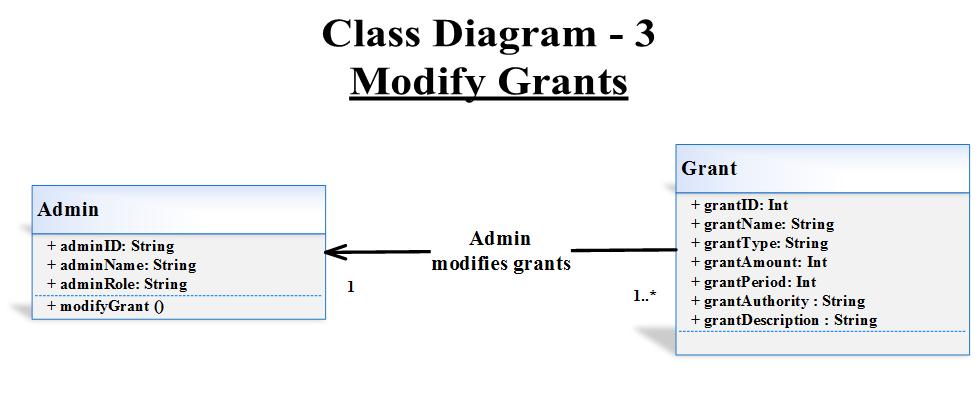


Url: <http://pictureintext.net/pro/use/Design-3/>

Or

CLICK HERE TO SEE THE PROTOTYPE - 3 IN ACTION

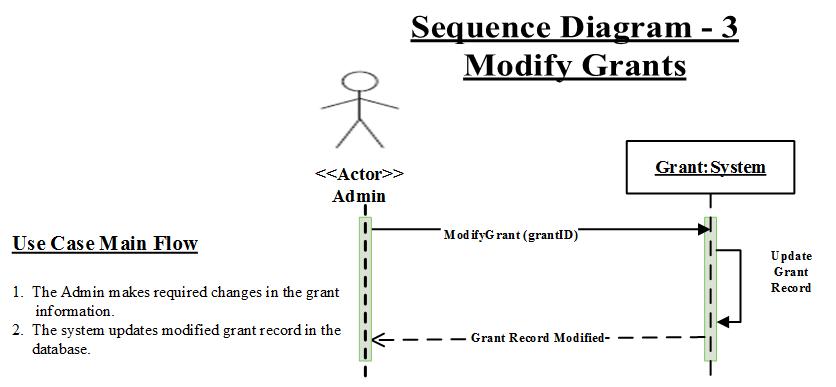
## Class Diagram – 3



## Class Diagram Narrative – 3

The modify grants class diagram explains the client two classes: Admin and Grant. The admin class holds admin’s attributes and the grant class holds grants attributes. The client after seeing this class model will get impression of the detailed attributes and the associated method regarding modify grants use case. The admin class interacts with grants class to modify the already existing grant having one to many multiplicity relations. Both classes have association relationship with each other.

## Sequence Diagram - 3



# Use Case 4: Delete Grants

## Use Case Description – 4

**Use Case Specification: Delete Grants**

**Use-Case Name: Delete Grants**

**Brief Description**

This use case allows the admin to delete already existing grants from the Grants Management System. The university students, then, will not be able to apply on the grant that does not exist for them, anymore.

**Flow of Events**

**Basic Flow**

1. The admin selects "Delete Grants" option from the menu options.
2. The system displays "Delete Grants" interface.
3. The admin enters grant ID to access the grant.
4. The system displays the grant information.
5. The admin presses “Delete” button.
6. The system prompts “Confirm Delete” dialog box with “Yes” and “No” options.
7. The admin presses “Yes” button to confirm deletion.
8. The system deletes the grant record from the database.

**Alternative Flows**

**First Alternative Flow**

5.1. The admin presses “Cancel” button.

5.2. The system prompts “Confirm Cancel” dialog box with “Yes” and “No” options.

**An Alternative Subflow**

5.2.1. The admin presses “Yes” button.

5.2.2. The system clears the screen.

**An Alternative Subflow**

5.2.3. The admin presses “No” button.

5.2.4. The system takes no action.

**Second Alternative Flow**

6.1. The admin presses “No” button.

6.2. The system takes no action.

**Special Requirements**

N.A.

**First Special Requirement**

N.A.

**Pre-conditions**

**Pre-condition One**

The admin has already created a grant in the system.

**Post-conditions**

**Post-condition One**

The users can’t find the information of deleted grant to apply.

**Extension Points**

N.A.

**Name of Extension Point**

N.A.

**System Requirements**

**Operational**

1. The system operates on Windows 7, 8 & 10, Linux 2005 Minimum, or MAC OS 10.0.

2. The system has a 2.8 GHz processor with 4 GB of free disk space.

4. The system has an internet connection for software activation.

5. The system is integrated with the existing University of Louisville Research database.

6. The system also works on mobile devices.

**Performance**

1. The system runs 24 hours per day, 365 days per year.

2. The system supports the capacity of the University of Louisville Research Database.

3. The system handles 100 transactions per hour.

**Security**

1. The system encrypts all deleted grant transactions.

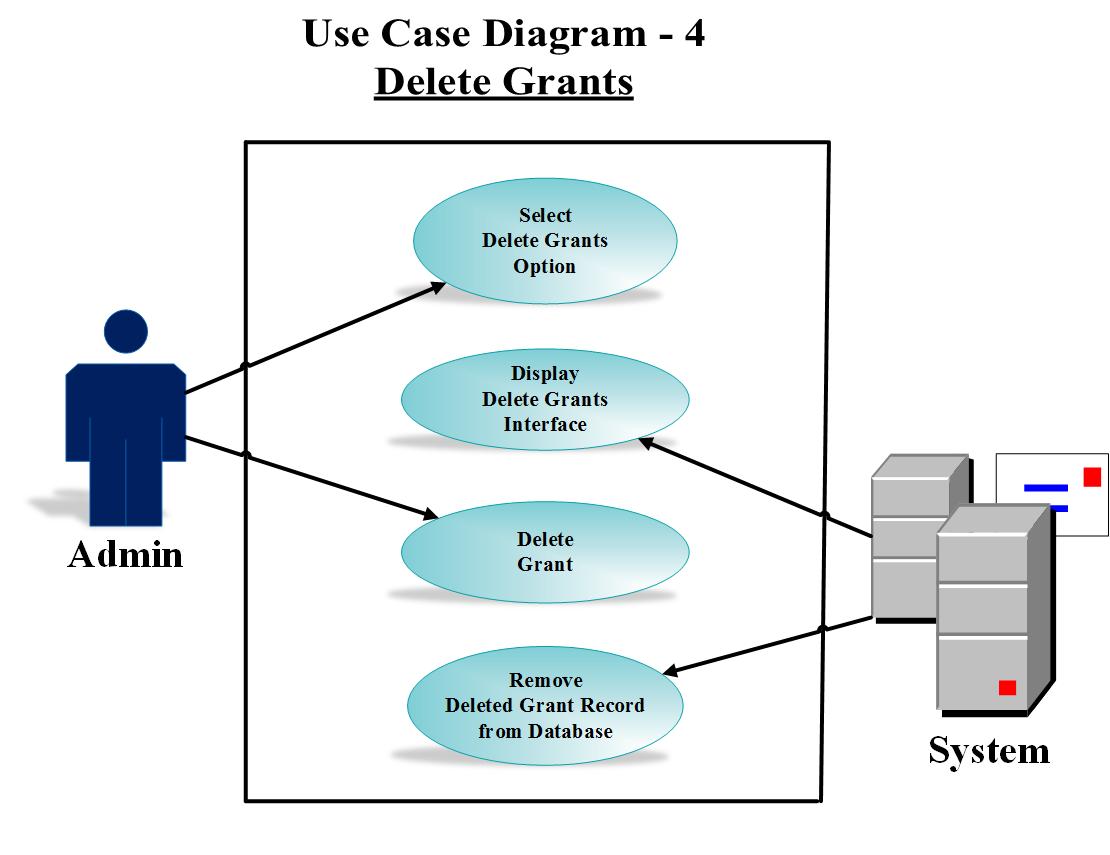
**Functional**

1. The system displays “Delete Grants” interface to the user.
2. The system successfully deletes grants from the database.

**Trace Matrix**

The system is user friendly for all the users including admin who is the main actor for deleting grants using the “Delete Grants” Interface. The system allows admin to view the grant information and delete it so that the users should not view it.

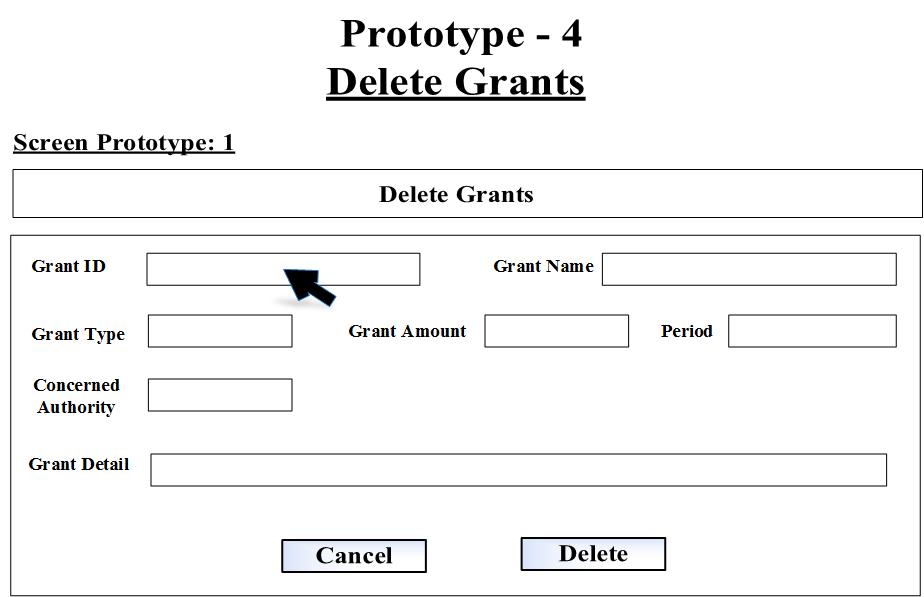
## Use Case Diagram – 4



## Use Case Narrative – 4

This use case provides the client a flow of actions through which the actor admin involves deleting a previously existing grant record. The admin chooses delete grant interface and the system displays the interface. The use case requires the admin to view grant information and delete the grant. The system then also deletes / removes the grant record from the database.

## Prototype – 4

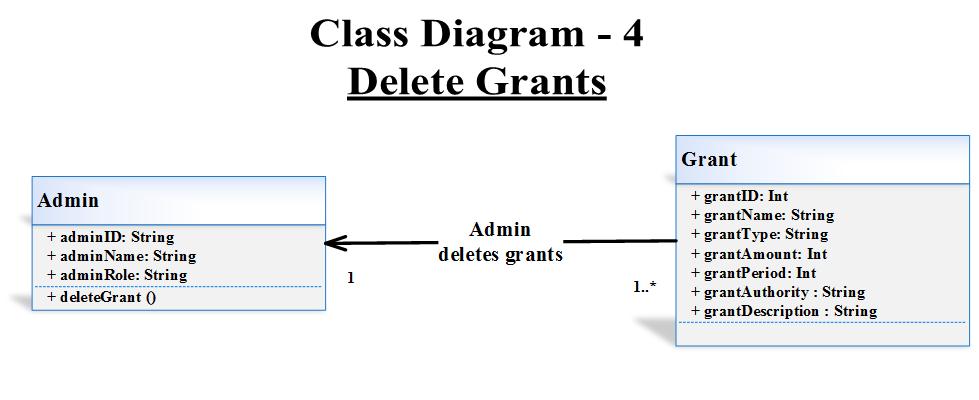


Url: <http://pictureintext.net/pro/use/Design-4/>

Or

CLICK HERE TO SEE THE PROTOTYPE - 4 IN ACTION

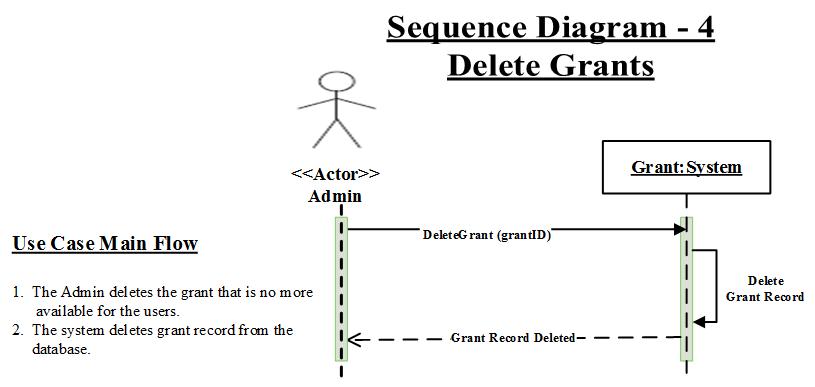
## Class Diagram – 4



## Class Diagram Narrative – 4

The delete grants class diagram explains the client two classes: Admin and Grant. The admin class holds admin’s attributes and the grant class holds grants attributes. The client once looks at this class model will receive impression of the thorough attributes and the associated method regarding delete grants class. The admin class interacts with grants class to delete a pre-existing grant devising one to many multiplicity relations. Both classes have association relationship with each other.

## Sequence Diagram -4



# Use Case 5 - Create Student Application

## Use Case Description – 5

**Use Case Specification: Create Student Application**

**Use-Case Name: Create Student Application**

**Brief Description**

This use case allows the student to create student application to apply for a grant through the Grants Management System.

**Flow of Events**

**Basic Flow**

1. The student selects "Create Student Application" option from the menu options.
2. The system displays "Create Student Application" interface.
3. The student enters all the required information of application.
4. The student presses “Submit” button.
5. The system saves student application record in the database.

**Alternative Flows**

**First Alternative Flow**

4.1. The student presses “Cancel” button.

4.2. The system clears all information from the interface.

**An Alternative Subflow**

N.A.

**Second Alternative Flow**

N.A.

**Special Requirements**

N.A.

**First Special Requirement**

N.A.

**Pre-conditions**

**Pre-condition One**

The student has browsed the website from internet.

**Post-conditions**

**Post-condition One**

The system successfully displays the student’s application, he has created.

**Extension Points**

N.A.

**Name of Extension Point**

N.A.

**System Requirements**

**Operational**

1. The system operates on Windows 7, 8 & 10, Linux 2005 Minimum, or MAC OS 10.0.

2. The system has a 2.8 GHz processor with 4 GB of free disk space.

4. The system has an internet connection for software activation.

5. The system is integrated with the existing University of Louisville Research database.

6. The system also works on mobile devices.

**Performance**

1. The system runs 24 hours per day, 365 days per year.

2. The system supports the capacity of the University of Louisville Research Database.

3. The system handles 100 transactions per hour.

**Security**

1. The system encrypts all student applications transactions.

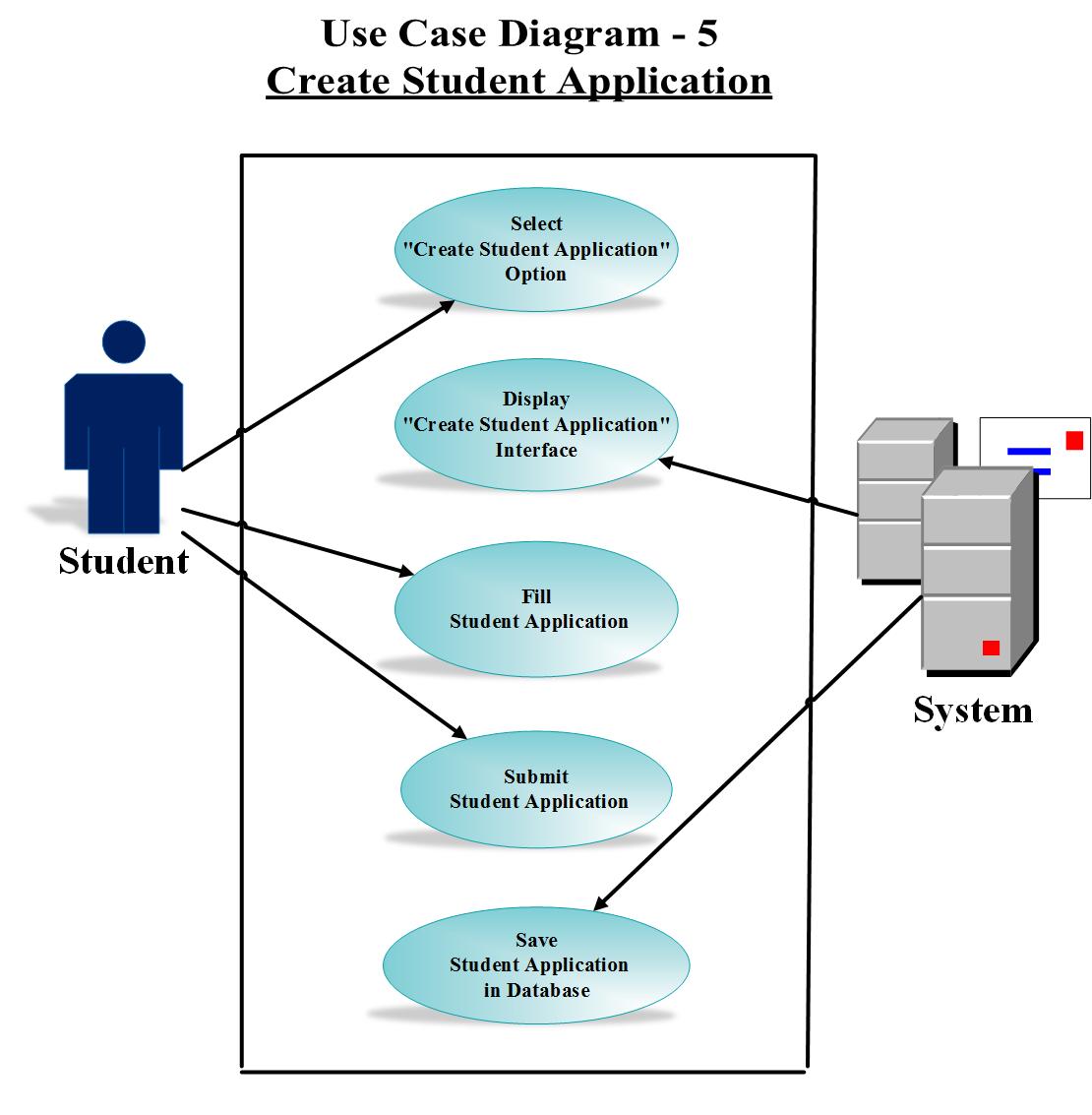
**Functional**

1. The system displays “Create Student Application” interface to the user.
2. The system allows all required information of application to be filled by the student.

**Trace Matrix**

The system is user friendly for all the users using it specially the student who is the single actor of our use case “Create Student Application”. The system allows student to create a new grant application by filling the required information. The student submits application form and the system saves it.

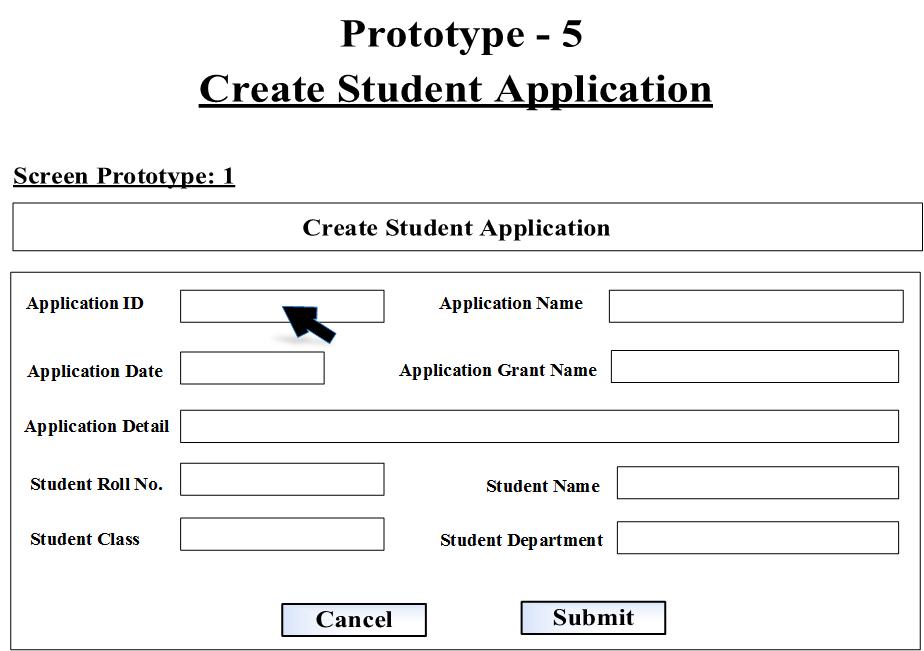
## Use Case Diagram – 5



## Use Case Narrative – 5

This use case delivers the client a stream of actions through which the actor student fills application to get a grant. The student chooses create student application interface and the system displays the interface. The use case entails the student to fill all the required information and submit the application. The system then saves the student application record in the database.

## Prototype – 5

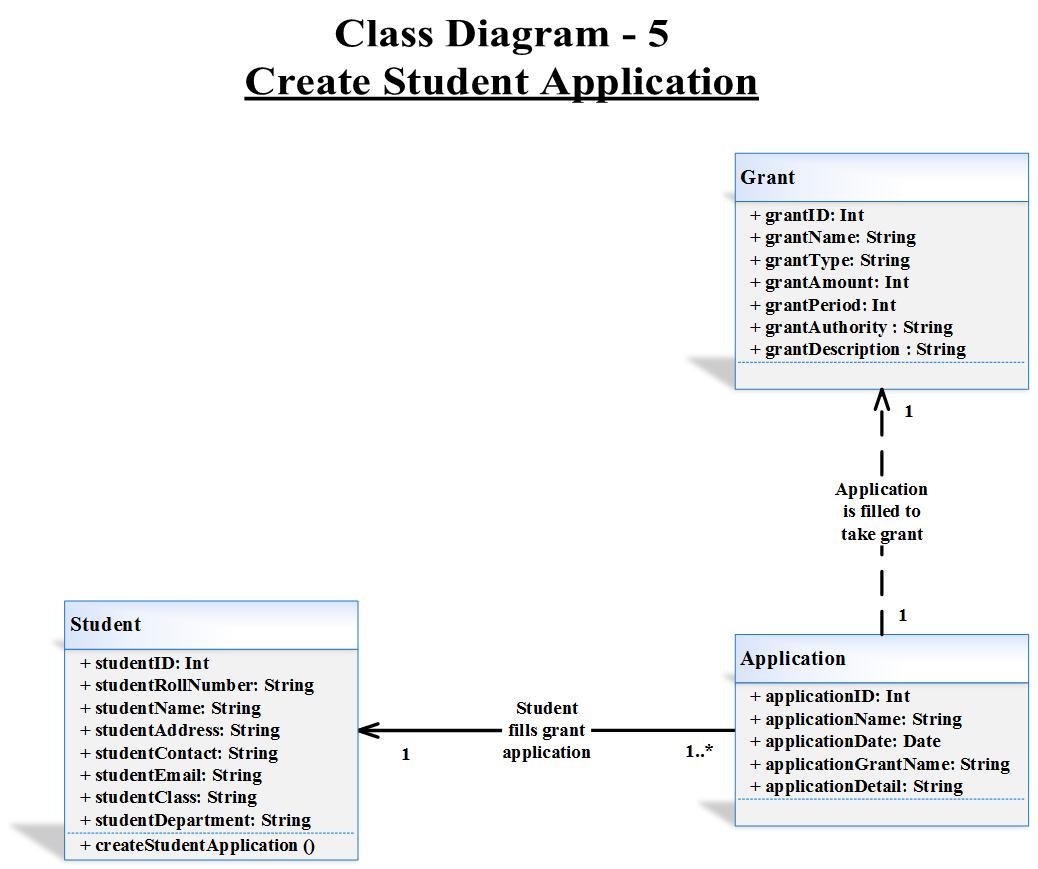


Url: <http://pictureintext.net/pro/use/Design-5/>

Or

CLICK HERE TO SEE THE PROTOTYPE - 5 IN ACTION

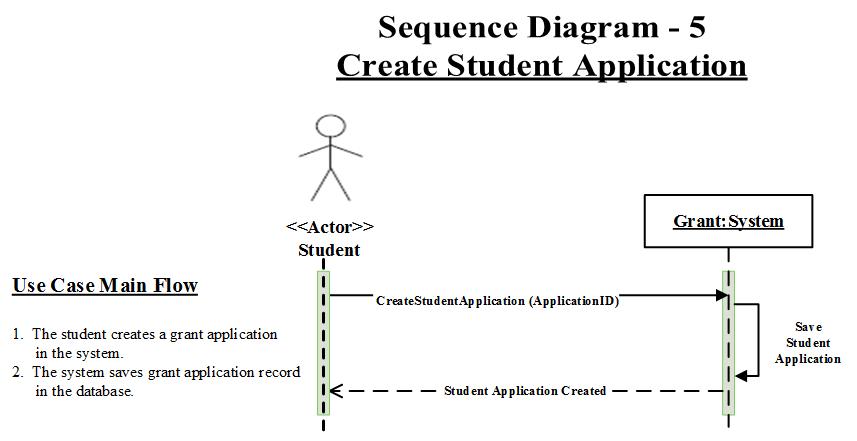
## Class Diagram – 5



## Class Diagram Narrative – 5

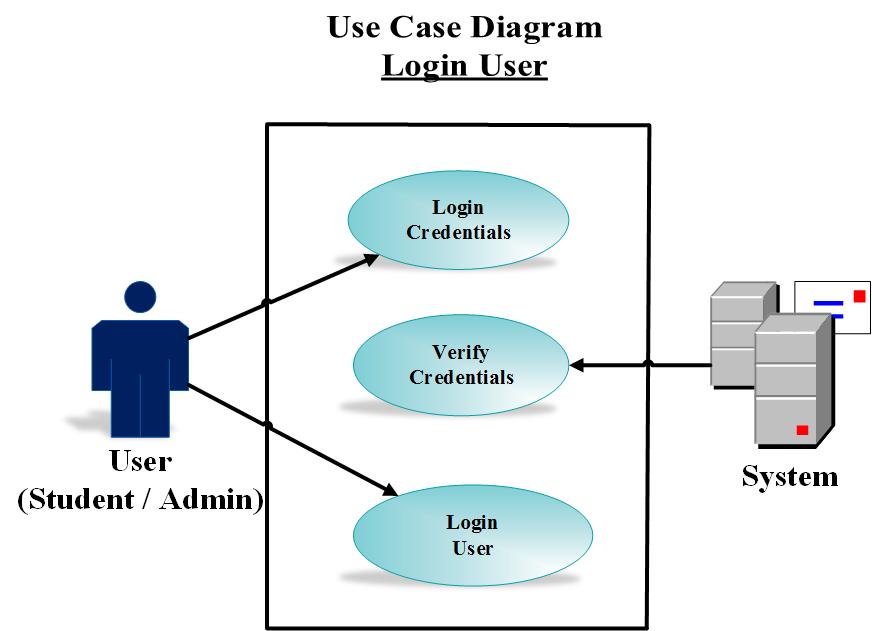
The create student application class diagram explains the client three classes: Student, Grant, and Application. The student class holds student’s attributes, the grant class clasps grants attributes, and the application class grips application attributes. The client when look at this class model, will have imprints of all the attributes and the associated method regarding create student application class diagram. The student class interacts with the application class to fill the grant application whereas application class is connected with grant class for taking grant. The application class is connected with the grant class using dependency relationship showing the one-to-one multiplicity relationship. On the other hand, student and application classes have association relationship with each other, multiplicity one-to-many.

## Sequence Diagram - 5



# Use Case 6 – Login User

## Use Case Diagram – 6



## Use Case Narrative - 6

This use case diagram describes the client a flow of actions through which the actor (student / admin) logs into the system. The system checks the credentials and allows the user to login.