**SYSTEM DESIGN**

**Data Flow Diagrams**

A graphical tool used to describe and analyze the moment of data through a system manual or automated including the process, stores of data, and delays in the system. Data Flow Diagrams are the central tool and the basis from which other components are developed. The transformation of data from input to output, through processes, may be described logically and independently of the physical components associated with the system. The DFD is also know as a data flow graph or a bubble chart.

DFDs are the model of the proposed system. They clearly should show the requirements on which the new system should be built. Later during design activity this is taken as the basis for drawing the system’s structure charts.

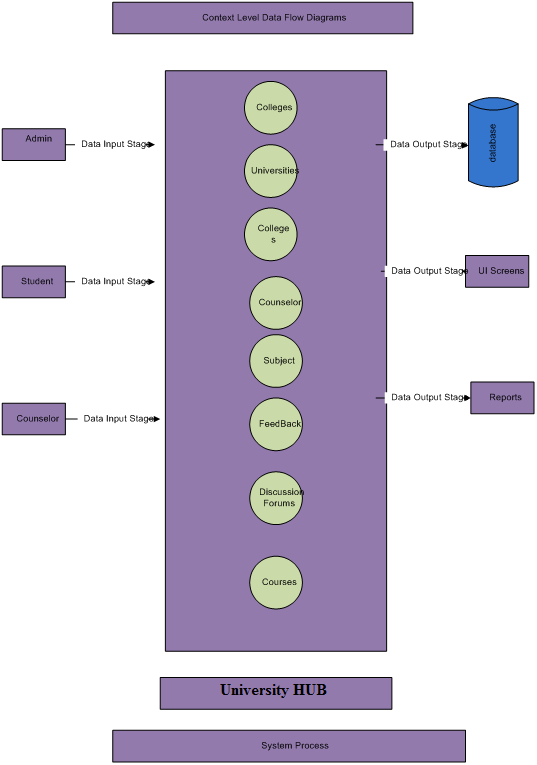
The Basic Notation used to create a DFD’s are as follows:

**1. Dataflow:** Data move in a specific direction from an origin to a destination.

**2. Process:** People, procedures, or devices that use or produce (Transform) Data. The physical component is not identified.

**3. Source:** External sources or destination of data, which may be People, programs, organizations or other entities.

**4. Data Store:** Here data are stored or referenced by a process in the System.

**Context Level Data Flow Diagram**

**Authentication Data Flow Diagram:**



**Level1 Data Flow Diagram for Admin (Institute Management):**

**Level -1 Data Flow Diagram:**



**Level -2 Data Flow Diagram:**



**Level-1 Data Flow Diagram:**



**Level -2 Diagram: Level -2 Diagram**





**Level1 Data Flow Diagram:** 

Level-2

**Level -2 Diagram**





Student User :

Level3 Data Flow Diagram:



Level3 Data Flow Diagram:



Counselor User:

Level3 Data Flow Diagram:



Leve3 Data Flow Diagram:



**UML DIAGRAMS**

**UNIFIED MODELING LANGUAGE DIAGRAMS**

The unified modeling language allows the software engineer to express an analysis model using the modeling notation that is governed by a set of syntactic semantic and pragmatic rules.

A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagram, which is as follows.

**USER MODEL VIEW**

This view represents the system from the users perspective.

The analysis representation describes a usage scenario from the end-users perspective.

**STRUCTURAL MODEL VIEW**

In this model the data and functionality are arrived from inside the system.

This model view models the static structures.

**BEHAVIORAL MODEL VIEW**

It represents the dynamic of behavioral as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

**IMPLEMENTATION MODEL VIEW**

In this the structural and behavioral as parts of the system are represented as they are to be built.

**ENVIRONMENTAL MODEL VIEW**

In this the structural and behavioral aspects of the environment in which the system is to be implemented are represented.

UML is specifically constructed through two different domains they are:

UML Analysis modeling, which focuses on the user model and structural model views of the system.

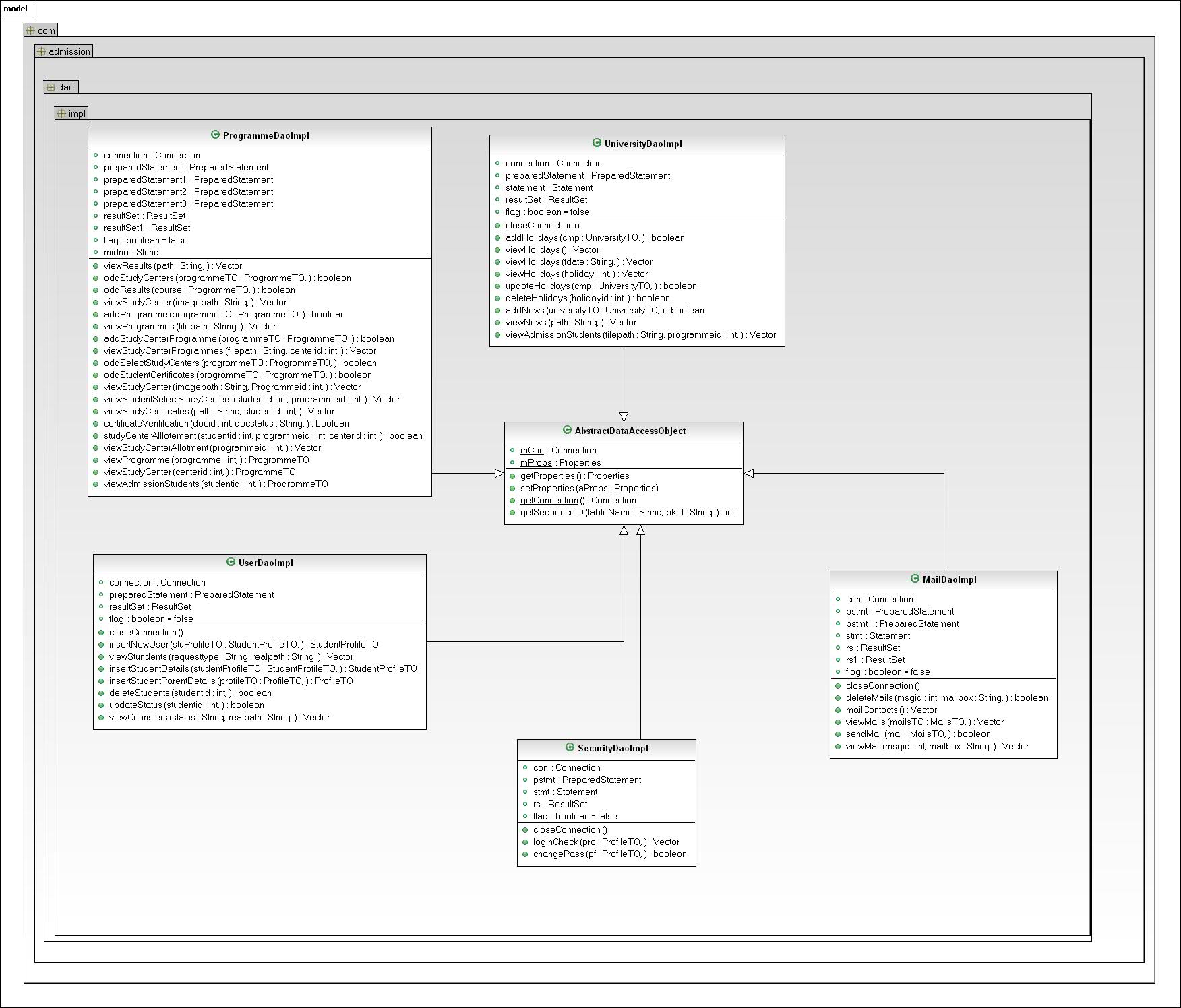
UML design modeling, which focuses on the behavioral modeling, implementation modeling and environmental model views.

Use case Diagrams represent the functionality of the system from a user’s point of view. Use cases are used during requirements elicitation and analysis to represent the functionality of the system. Use cases focus on the behavior of the system from external point of view.

Actors are external entities that interact with the system. Examples of actors include users like administrator, bank customer …etc., or another system like central database.

**CLASS DIAGRAM**

Class diagrams describe the structure of the system in terms of classes and objects. The servlet api class diagram will be as follows.

****

**Use Case Diagrams**

**UML Diagrams**

**Unified Modeling Language**:

The Unified Modeling Language allows the software engineer to express an analysis model using the modeling notation that is governed by a set of syntactic semantic and pragmatic rules.

A UML system is represented using five different views that describe the system from distinctly different perspective. Each view is defined by a set of diagram, which is as follows.

* + User Model View
    1. This view represents the system from the users perspective.
    2. The analysis representation describes a usage scenario from the end-users perspective.
  + Structural model view
    1. In this model the data and functionality are arrived from inside the system.
    2. This model view models the static structures.
* Behavioral Model View

It represents the dynamic of behavioral as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

* Implementation Model View

In this the structural and behavioral as parts of the system are represented as they are to be built.

* Environmental Model View

In this the structural and behavioral aspects of the environment in which the system is to be implemented are represented.

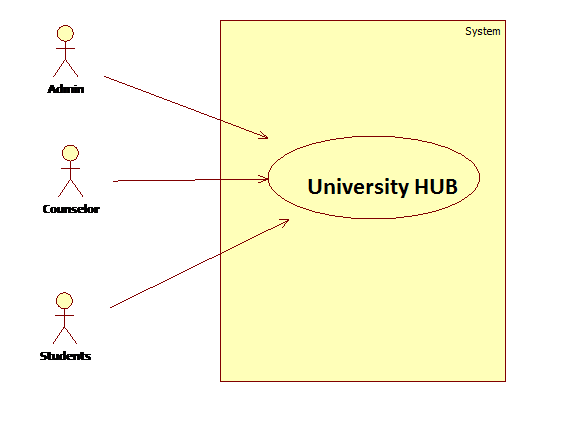
UML is specifically constructed through two different domains they are:

* UML Analysis modeling, this focuses on the user model and structural model views of the system.
* UML design modeling, which focuses on the behavioral modeling, implementation modeling and environmental model views.

Use case Diagrams represent the functionality of the system from a user’s point of view. Use cases are used during requirements elicitation and analysis to represent the functionality of the system. Use cases focus on the behavior of the system from external point of view.

Actors are external entities that interact with the system. Examples of actors include users like administrator, bank customer …etc., or another system like central database.

1. **system Use Case Diagram**

****

1. **Administrator Use Case Diagram**

****

1. **Counselors Use Case Diagram**



1. **Student Use Case Diagram :**

**Student**

**StudyCenter**

**Programme**

**ViewStudycenters**

**ViewStudyCenter**

**ViewResults**

**Results**

**News**

**ViewNews**

**ViewHolidays**

**Holidays**

**Magzine**

**Mail**

**inbox**

**Compose**

**chat**

**outbox**

**Security**

**changePassword**

**logout**

**FillAdmissionForm**

**ViewStudentAllotementListletters**

**Sequence Diagrams**

**User-Level Sequence Diagrams**

**Login Sequence Diagram :**

****

1 : Execute()

**ChangePassword Sequence Diagram :**

ChangePasswordAction

Securitydelegate

Securityserviceimpl

Securitydaoimpl

dbutil

database

1 :changePass ()

2 : :changePass ()

3 : :changePass ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

1 : Execute()

**ForgetPassword:**

ForgetPasswordAction

Securitydelegate

Securityserviceimpl

Securitydaoimpl

dbutil

database

1 :forgetPass ()

2 : :forgetPass ()

3 : : forgetPass ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

1 : Execute()

Registration Secquence Diagram :

RegistrationAction

Userdelegate

Userserviceimpl

UserDaoImpl

dbutil

database

1 : insertNewUser ()

2 : : insertNewUser ()

3 : : insertNewUser ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : boolean

10 : boolean

11 : boolean /failure

1 : Execute()

**viewMail Sequence Diagram :**

ViewMailAction

Maildelegate

Mailserviceimpl

MailDaoImpl

dbutil

database

1 : viewMails ()

2 : : viewMails ()

3 : : viewMails ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

1 : Execute()

**sendMail Sequence Diagram :**

SendMailAction

Maildelegate

Mailserviceimpl

MailDaoImpl

dbutil

database

1 : sendMail ()

2 : : sendMail ()

3 : : sendMail ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

1 : Execute()

**deleteMail Sequence Diagram :**

DeleteMailAction

Maildelegate

Mailserviceimpl

MailDaoImpl

dbutil

database

1 : deleteMails ()

2 : : deleteMails ()

3 : : deleteMails ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

1 : Execute()

**AddHolidays Sequence Diagram:**

AddHolodaysAction

Companydelegate

Eompanyserviceimpl

CompanyDaoImpl

dbutil

database

1 : addHoliday()

()

2 : : addHoliday ()

3 : : addHoliday ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

1 : Execute()

**Add Programme Sequence Diagram:**

AddProgrammeAction

ProgrammeDelegate

ProgrammeServiceImpl

ProgrammeDaoImpl

dbutil

database

1 : addProgrammes

()

2 : : addProgrammes ()

3 : : addProgrammes ()

4 : addProgrammes ()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

1 : Execute()

**Add Results Sequence Diagram:**

AddResultAction

ProgrammeDelegate

ProgrammeServiceImpl

ProgrammeDaoImpl

dbutil

database

1 : addResultAction

()

2 : : addResultAction ()

3 : : addResultAction ()

4 : addResultAction ()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

1 : Execute()

**Add Study Centers Sequence Diagram:**

AddStudyCenterAction

ProgrammeDelegate

ProgrammeServiceImpl

ProgrammeDaoImpl

dbutil

database

1 : addStudyCenterAction

()

2 : : addStudyCenterAction ()

3 : : addStudyCenterAction ()

4 : addStudyCenterAction ()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

1 : Execute()

**View Results Sequence Diagram:**

ViewResultAction

ProgrammeDelegate

ProgrammeServiceImpl

ProgrammeDaoImpl

dbutil

database

1 : viewResultAction

()

2 : : viewResultAction ()

3 : : viewResultAction ()

4 : viewResultAction ()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

1 : Execute()

ViewStudentAllotmentOrder

ProgrammeDelegate

ProgrammeServiceImpl

ProgrammeDaoImpl

dbutil

database

1 : viewStudentAllotmentOrder

()

2 : : viewStudentAllotmentOrder ()

3 : : viewStudentAllotmentOrder ()

4 : viewStudentAllotmentOrder ()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

1 : Execute()

**View Student Certificates Sequence Diagram:**

ViewStudentCertificates

ProgrammeDelegate

ProgrammeServiceImpl

ProgrammeDaoImpl

dbutil

database

1 : viewStudentCertificates

()

2 : : viewStudentCertificates ()

3 : : viewStudentCertificates ()

4 : viewStudentCertificates ()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

1 : Execute()

**View Studycenter Sequence Diagram:**

ViewStudyCenter

ProgrammeDelegate

ProgrammeServiceImpl

ProgrammeDaoImpl

dbutil

database

1 : viewStudyCenter

()

2 : : viewStudyCenter ()

3 : : viewStudyCenter ()

4 : viewStudyCenter ()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

1 : Execute()

**Collaboration Diagrams**

**Login Sequence Collabration Diagram :**



**Changepassword SequenceCollbration diagram :**

ChangepasswordAction

Securitydelegate

Securityserviceimpl

Securitydaoimpl

dbutil

database

1 : changePass()

2 :changePass()

3 : changePass()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

ForgetpasswordAction

Securitydelegate

Securityserviceimpl

Securitydaoimpl

dbutil

database

1 : forgetPass ()

2 :forgetPass()

3 : forgetPass ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

**Registration Sequence CollbrationDiagram:**

RegistrationAction

Userdelegate

Userserviceimpl

Userdaoimpl

dbutil

database

1 : insertNewUser()

2 : insertNewUser()

3 : insertNewUser ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

**SendMail Sequence Collabration Diagram**

SendMailAction

Maildelegate

Mailserviceimpl

Maildaoimpl

dbutil

database

1 : sendMail()

2 : sendMail ()

3 : sendMail ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

**ViewMail Sequences Collabration Diagram:**

ViewMailAction

Maildelegate

Mailserviceimpl

Maildaoimpl

dbutil

database

1 : viewMails()

2 : viewMails ()

3 : viewMails ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

**deleteMail Sequences Collabration Diagram:**

DeleteMailAction

Maildelegate

Mailserviceimpl

Maildaoimpl

dbutil

database

1 : deleteMails()

2 : deleteMails ()

3 : deleteMails ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

AddHolidays SequenceCollbration Diagram:

AddHolodaysAction

Companydelegate

Eompanyserviceimpl

CompanyDaoImpl

dbutil

database

1 : addHoliday ()

2 : addHoliday ()

3 : addHoliday ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

Add Admission Sequence Collaboration Diagram:

AddAdmissionDetailsAction

ProgrammeDelegate

ProgrammeServiceImpl

ProgrammeDaoImpl

dbutil

database

1 : addAdmissionDetailsAction

()

2 : addAdmissionDetailsAction ()

3 : addAdmissionDetailsAction ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

Add Programme Sequence Collaboration Diagram:

AddProgrammeAction

ProgrammeDelegate

addProgrammeAction

addProgrammeAction

dbutil

database

1 : addProgrammeAction

()

2 : addAdmissionDetailsAction ()

3 : addAdmissionDetailsAction ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

Add Results Sequence Collaboration Diagram:

AddResultAction

ProgrammeDelegate

ProgrammeServiceImpl

ProgrammeDaoImpl

dbutil

database

1 : addResultAction

()

2 : addResultAction ()

3 : addResultAction ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

Add StudyCenter Sequence Collaboration Diagram:

AddStudyCenterAction

ProgrammeDelegate

ProgrammeServiceImpl

ProgrammeDaoImpl

dbutil

database

1 : addStudyCenterAction

()

2 : addStudyCenterAction ()

3 : addStudyCenterAction ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

view StudyCenter Sequence Collaboration Diagram:

ViewStudyCenter

ProgrammeDelegate

ProgrammeServiceImpl

ProgrammeDaoImpl

dbutil

database

1 : viewStudyCenter

()

2 : viewStudyCenter ()

3 : viewStudyCenter ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

View StudyCenters Sequence Collaboration Diagram:

ViewStudyCenterAllotements

ProgrammeDelegate

ProgrammeServiceImpl

ProgrammeDaoImpl

dbutil

database

1 : viewStudyCenterAllotements

()

2 : viewStudyCenterAllotements ()

3 : viewStudyCenterAllotements ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

View University Sequence Collaboration Diagram:

ViewUniversityNewsAction

ProgrammeDelegate

ProgrammeServiceImpl

ProgrammeDaoImpl

dbutil

database

1 : viewUniversityNewsAction

()

2 : viewUniversityNewsAction ()

3 : viewUniversityNewsAction ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

**ACTIVITY DIAGRAMS**

**ACTIVITY DIAGRAMS**

1. **Activity Diagram for Admin :**

****

1. **Activity Diagram for Counselor :**

Counsler

StudyCenter

Programmme

StudyCenterProgramme

mails

Holidays

News

Magzine

Results

view

view

view

view

view

view

Compose

Security

inbox

sentitems

chat

changepass

logout

false

true

ViewStudentAdmission

VerifyCertificates

AllotSeats

AllotmentOrderLetters

viewStudentAllotments

1. **Activity Diagram for Student**

Studentt

StudyCenter

Programmme

StudyCenterProgramme

mails

Holidays

News

Magzine

Results

view

view

view

view

view

view

Compose

Security

inbox

sentitems

chat

changepass

logout

false

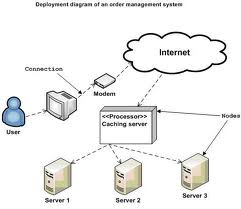
true

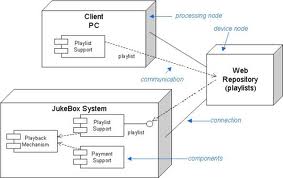
Fill the admission form

viewStudentAllotments

**Component Diagram**

**Component Diagram:**





**Deployment Diagram**

**Deployment Diagram:**

****