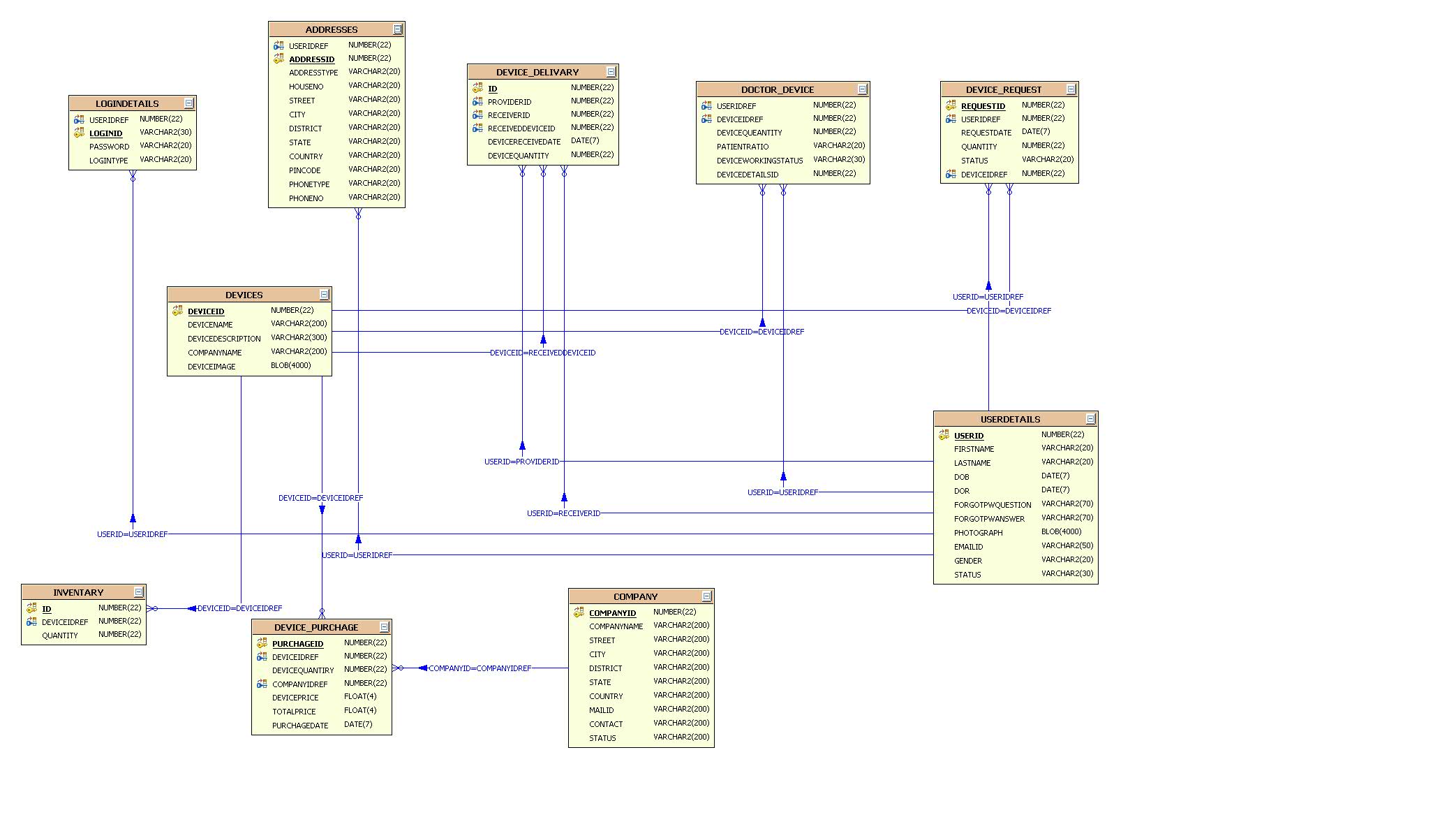
**SYSTEM DESIGN**

**E - R Diagrams**



**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.

UML Diagrams in Visual Case :

Use Case Diagram

The Use Case Diagram describes the system functionality

as a set of Use Cases which represent discrete tasks.

Actors interact with the system to complete the tasks.

Class Diagram

The Class Diagram describes the structure of the software system.

This is the core diagram for object-oriented design.

Sequence Diagram:

The Sequence Diagram describes messages exchanged between

classes to accomplish tasks.

Collaboration Diagram

The Collaboration Diagram describes interactions between classes and associations.

State Diagram

State Diagrams model the dynamic behaviour of a system by showing the various

states that an object can get into and the transitions that occur between the states.

Activity Diagram

The Activity Diagram describes the activities of a class in response to internal events.

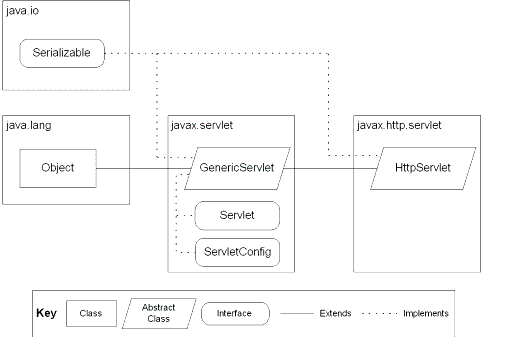
Component Diagram

The Component Diagram describes the structure and dependencies among software components.

Deployment Diagram

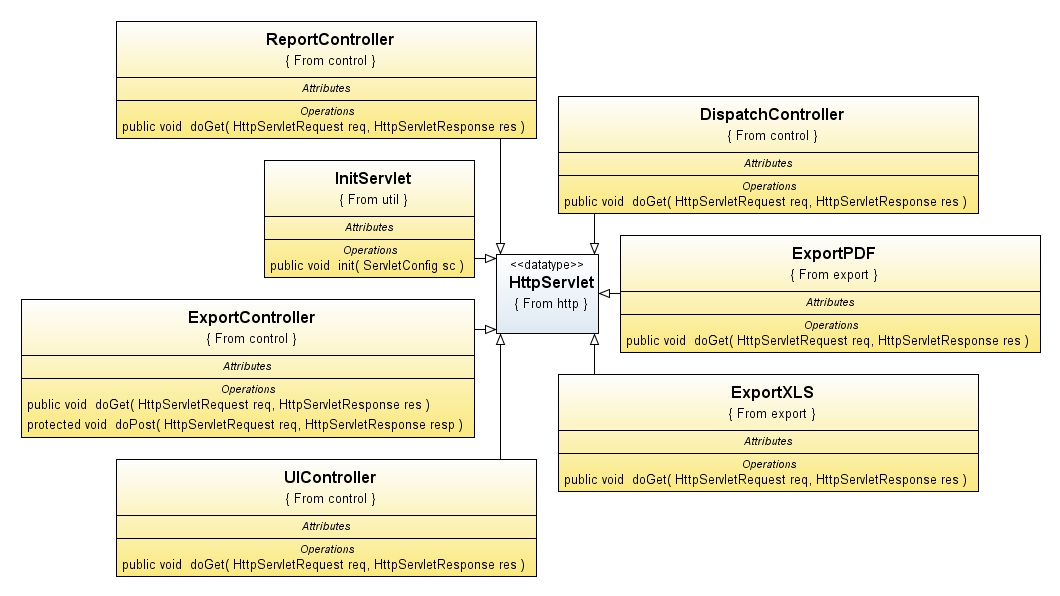
The Deployment Diagram describes the physical layout of software components.

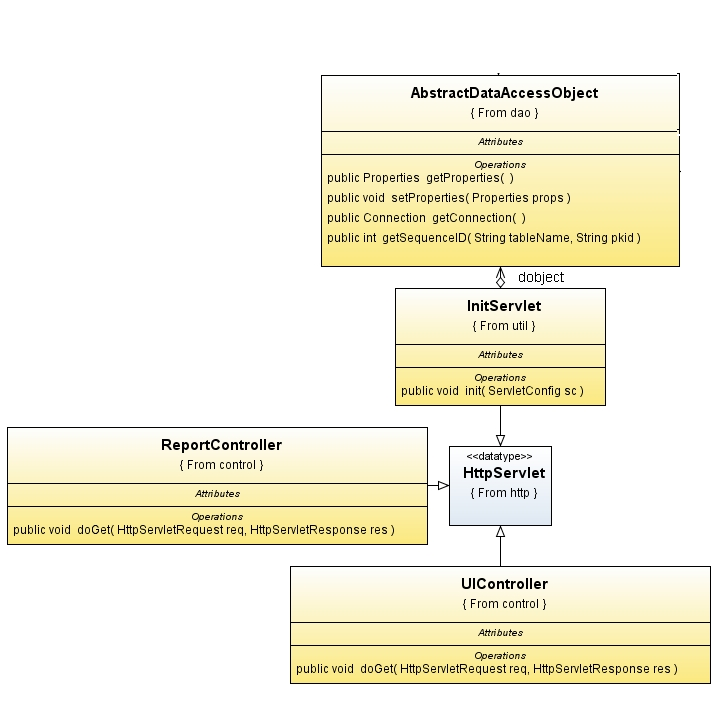
**Class Diagram**



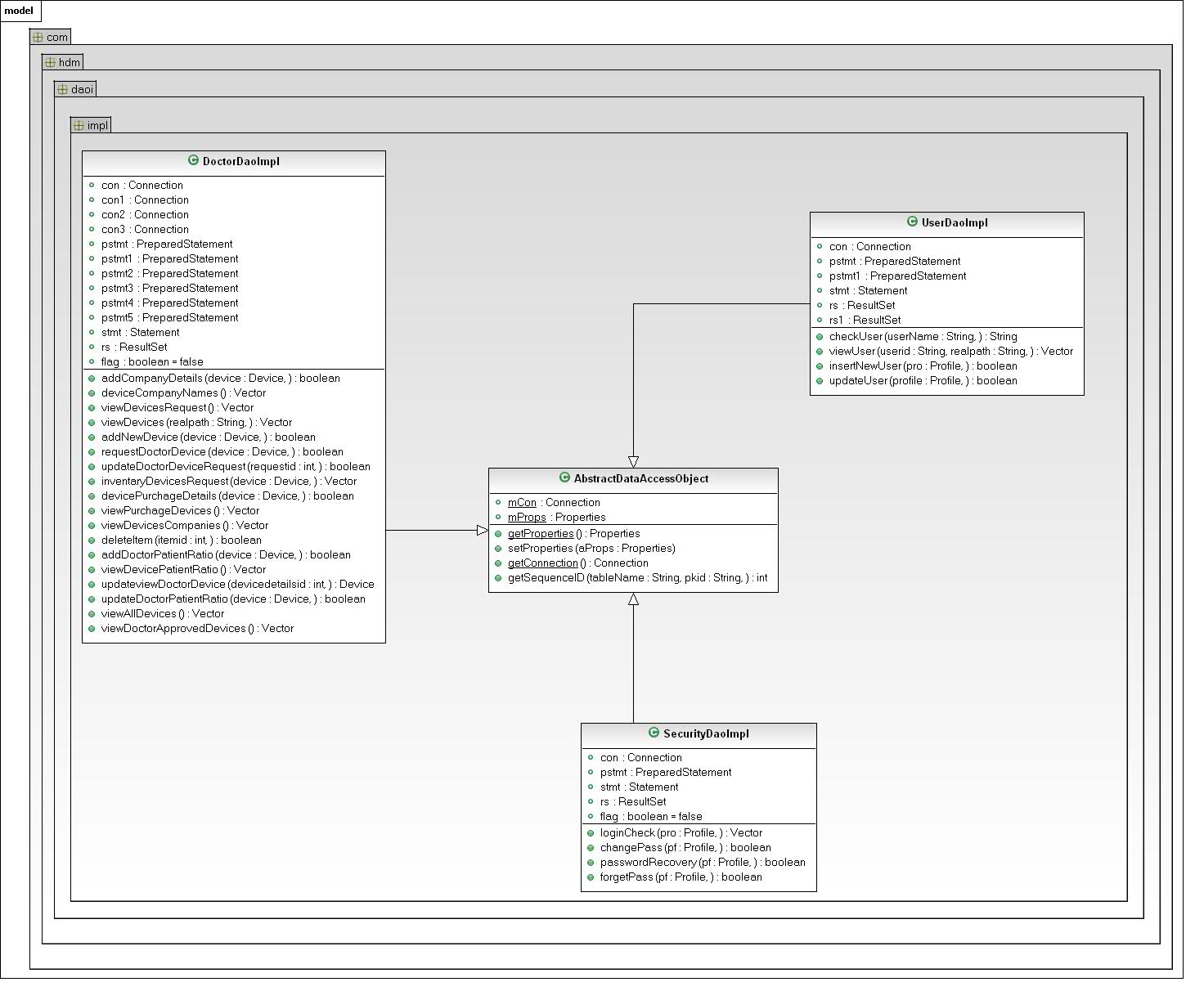
JSP: Implicit Objects

**Class Collaboration Diagrams**

**Class Collaboration Diagram**



**CLASS COLLABRATION DIAGRAM :**



**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. **Admin Use Case Diagram**

****

1. **Doctor Use Case Diagram**

****

1. **Financer Use Case Diagram**

****

**Sequence Diagrams**

**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()

Update UserProfile Sequence Diagram :

UpdateUserProfileAction

Userdelegate

Userserviceimpl

UserDaoImpl

dbutil

database

1 : updateUser ()

2 : : updateUser ()

3 : : updateUser ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

1 : Execute()

view UserProfile Sequence Diagram :

ViewProfileAction

Userdelegate

Userserviceimpl

UserDaoImpl

dbutil

database

1 : viewUser ()

2 : : viewUser ()

3 : : viewUser ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

1 : Execute()

AddNewDevice Sequence Diagram :

AddNewDeviceAction

Doctordelegate

Doctorserviceimpl

DoctorDaoImpl

dbutil

database

1 : addNewDevice ()

2 : : addNewDevice ()

3 : : addNewDevice ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

1 : Execute()

AddCompanyAction Sequence Diagram:

AddCompanyAction

Doctordelegate

Doctorserviceimpl

DoctorDaoImpl

dbutil

database

1 : addCompanyDetails ()

2 : : addCompanyDetails ()

3 : : addCompanyDetails ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

1 : Execute()

AddDoctorPatientRatio Sequence Diagram:

AddDoctorPatientRatioAction

Doctordelegate

Doctorserviceimpl

DoctorDaoImpl

dbutil

database

1 : addDoctorPatientRatio ()

2 : : addDoctorPatientRatio ()

3 : : addDoctorPatientRatio ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

1 : Execute()

ViewDevicePatientRatio Sequence Diagram:

viewDevicerPatientRatioAction

Doctordelegate

Doctorserviceimpl

DoctorDaoImpl

dbutil

database

1 : viewDoctorPatientRatio ()

2 : : viewDoctorPatientRatio ()

3 : : viewDoctorPatientRatio ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

1 : Execute()

**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()

Update UserProfile SequenceCollbration Diagram :

UpdateUserProfileAction

Userdelegate

Userserviceimpl

Userdaoimpl

dbutil

database

1 : updateUser ()

2 : updateUser()

3 : updateUser ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

View UserProfile Sequence Collabration Diagram :

ViewUserProfileAction

Userdelegate

Userserviceimpl

Userdaoimpl

dbutil

database

1 : viewUser ()

2 : viewUser ()

3 : viewUser ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

AddNewDevice Sequence Collabration Diagram :

AddNewDeviceAction

Doctordelegate

Doctorserviceimpl

DoctorDaoImpl

dbutil

database

1 : addNewDevice ()

2 : addNewDevice ()

3 : addNewDevice ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

AddCompanyAction Sequence Collabration Diagram:

AddCompanyAction

Doctordelegate

Doctorserviceimpl

DoctorDaoImpl

dbutil

database

1 : addCompanyDetails ()

2 : addCompanyDetails ()

3 : addCompanyDetails ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

AddDoctorPatientRatio Sequence Collabration Diagram:

AddDoctorPatientRatioAction

Doctordelegate

Doctorserviceimpl

DoctorDaoImpl

dbutil

database

1 : addDoctorPatientRatio ()

2 : addDoctorPatientRatio ()

3 : addDoctorPatientRatio ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

ViewDevicePatientRatio Sequence Collabration Diagram:

viewDevicerPatientRatioAction

Doctordelegate

Doctorserviceimpl

DoctorDaoImpl

dbutil

database

1 : viewDoctorPatientRatio ()

2 : viewDoctorPatientRatio ()

3 : viewDoctorPatientRatio ()

4 : getConnection()

5 : getConnection()

6 : getConnection

7 : exexutequery()

8 : queryResult

9 : returnStatus()

10 : returnStatus

11 : Success/failure()

**ACTIVITY DIAGRAMS**

**ACTIVITY DIAGRAMS**

##### Administrator Activity Diagram



##### Doctor Activity Diagram



##### Financer Activity Diagram

****

\

**Component Diagram :**

****

**Deployment Diagram:**

****