Complain Box

Submitted To

Kishan Kumar Ganguly
Lecturer
Institute of Information Technology

Submitted By

Saif Kamal Chowdhury (BSSE0924)
Anika Tabassum (BSSE0925)
A.T.M. Fazlay Rabbi (BSSE0926)
Niraj Chaudhary (BSSE0942)

Submission Date

November 6, 2019

IIT
University of Dhaka

Table of Contents:

Chapter One: Architectural Design **3**
1.1 Representing Complain Box in Context 3
1.2 Archetypes 4
1.3 Refining Architecture into Components 6
Chapter Two: Component Level Design **8**
2.1 Analysis Classes 8
2.2 Elaboration of Analysis Classes 10
2.2.1 Admin Class 10
2.2.2 General User Class 11
2.2.3 Complain Submission Class 12
2.2.4 Complain Supervision Class 13
2.3 Collaboration between Classes or Components 14
2.4 Component Based Development 15
2.4.1 Complain Submission 15
2.4.2 Complain Supervision 16
2.4.3 General User 17
2.4.4 Notice 18
2.4.5 Emergency Institution 19
2.5 State Chart Fragment for Classes 20
2.5.1 Admin Class : 20
2.5.2 GeneralUser Class : 21
2.5.3 ComplainSubmission Class : 22
2.5.4 ComplainSupervision Class : 23
2.5.5 Notice Class : 24
2.5.4 EmergencyInstitution Class : 25

List of Figures:

Figure 1: Architectural Context Diagram for Complain Box 5
Figure 2:UML relationship for Complain Box archetypes 6
Figure 3:overall architectural structure for Complain Box with top-level components 7
Figure 4: An instantiation of the Complain Box with component elaboration 8
Figure 5: Analysis Classes of Complain Box 10
Figure 6:Elaboration of Design Component Admin 11
Figure 7:Elaboration of Design Component General User 12
Figure 8:Elaboration of Design Component Complain Submission 13
Figure 9:Elaboration of Design Component Complain Supervision 14
Figure 10:Collaboration Diagram with messaging 15
Figure 12: Pseudo Code of Component addComplainDescription from class complainSubmission 16
Figure 13: Pseudo Code of Component updateComplain from class complainSupervision 17
Figure 14: Flowchart of Component createAccount from class generalUser 18
Figure 15: Pseudo code of Component createNotice from class Notice 19
Figure 16: Pseudo code of Component callEmergencyInstituion from class EmergencyInstitution 20
Figure 17: Pseudo code of Component callEmergencyInstituion from class Admin 21
Figure 18: Statechart fragment for Admin Class 22
Figure 19: Statechart fragment for General User Class 23
Figure 20: Statechart fragment for ComplainSubmission Class 24
Figure 21: Statechart fragment for ComplainSupervision Class 25
Figure 22: Statechart fragment for Notice Class 26
Figure 23: Statechart fragment for EmergencyInstitution Class 27

Chapter One: Architectural Design

1.1 Representing Complain Box in Context

To define which software interacts with entities of the system, we use context diagram, For our system, which is “Complain Box” the main Context details is given below:
Superordinate System: No other system uses complain box as their part. So Complain Box does not have any superordinate system.

Subordinate System: Complain box uses No other system as it’s part. So Complain Box does not have any subordinate system.

Peer-level System: No other system interact on a peer level with complain box.

Actors: Admin and **User** use the complain box to submit and supervise problems. So the actors are actors.

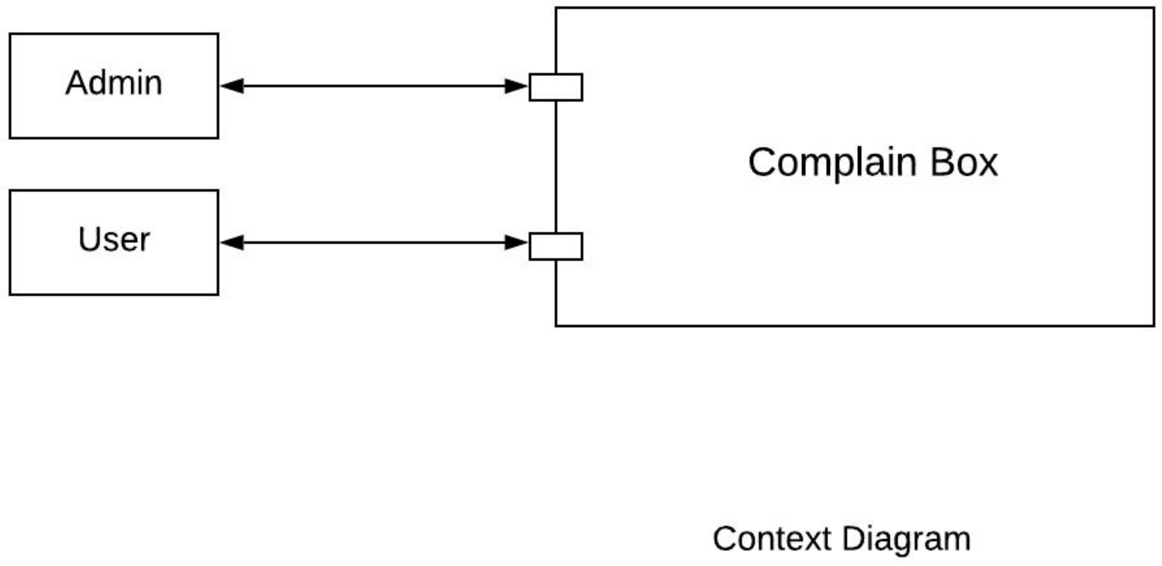


Figure 1: Architectural Context Diagram for Complain Box

1.2 Archetypes

An archetype is a class or pattern that represents a core abstraction that is critical to the design of an architecture for the target system. In general, a relatively small set of archetypes is required to design even relatively complex systems.

Complain Box is composed of four archetypes which represent stable elements of the architecture but may be instantiated many different ways based on the behavior of the system. The following four archetypes have been defined for the architectural design.

- **User :** This archetype represents all kinds of users for the Complain Box System. For example: Admin and Client can be the users of this system.

- **Complain :** An abstraction that encompasses all kinds of management required for maintaining complains. This is the core of the system. End users can submit complaints and Admin has to monitor them. All these tasks is represented by this archetype.

- **Notice :** An abstraction for notice management for this system.

- **Emergency Institution :** People can call for nearby hospitals, police stations, fire stations etc from this system. This archetype is the abstraction for this communication process.

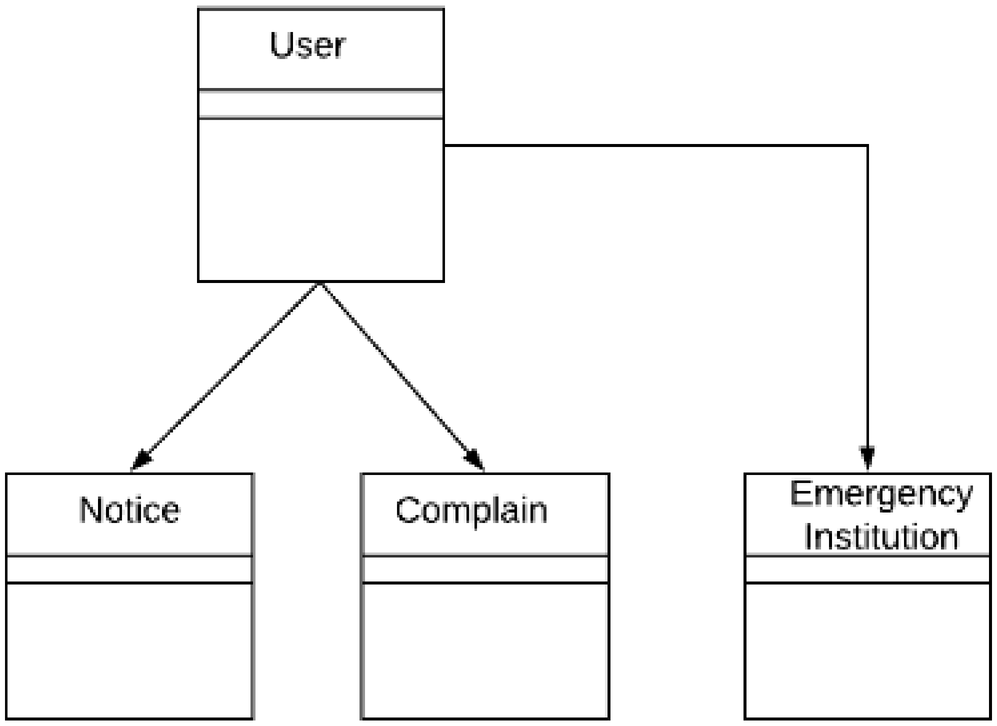


Figure 2:UML relationship for *Complain Box* archetypes

1.3 Refining Architecture into Components

To select the architectural component design of a system, we have to look at the components of various types. For complain box, we have selected some top level components that addressed the following functionalities:

Complain: Manages all complains into the complain box system.
Notice: Coordinates with all the notices of the system.
Emergency Institution: Connects with emergency institute through system.

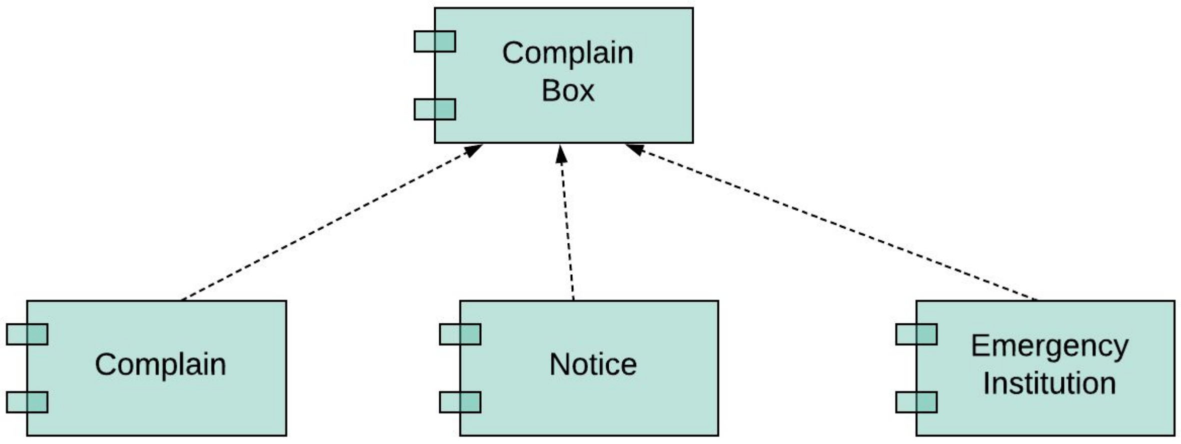


Figure 3:overall architectural structure for *Complain Box* with top-level components

The top level components have identified the major components of the system. Further refinement is necessary. To do that an actual instantiation of the architecture is developed. Components showed in figure 3 is elaborated to show additional detail. Such as-
Complain: This component interacts with complain submission and complain supervision.
Emergency Institution: This component manages emergency institution.
Notice: This component manages all the notice generation, updation and display.

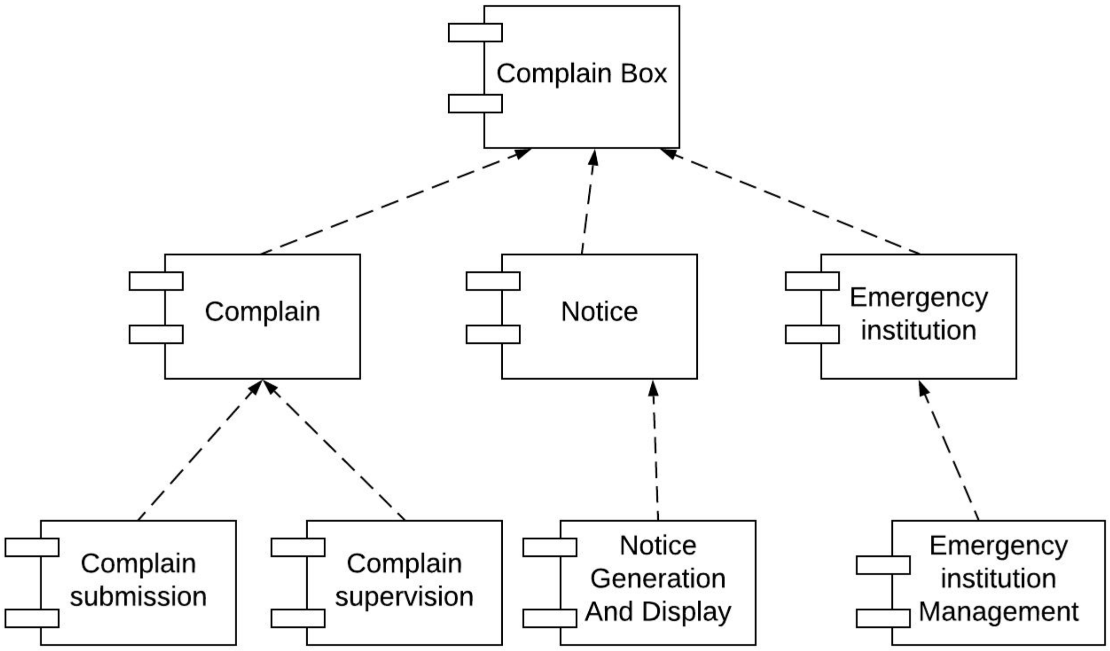


Figure 4: An instantiation of the *Complain Box* with component elaboration

Chapter Two: Component Level Design

2.1 Analysis Classes

As we have followed object oriented context, so the components are collaborating classes. For *Complain Box* we have these classes:

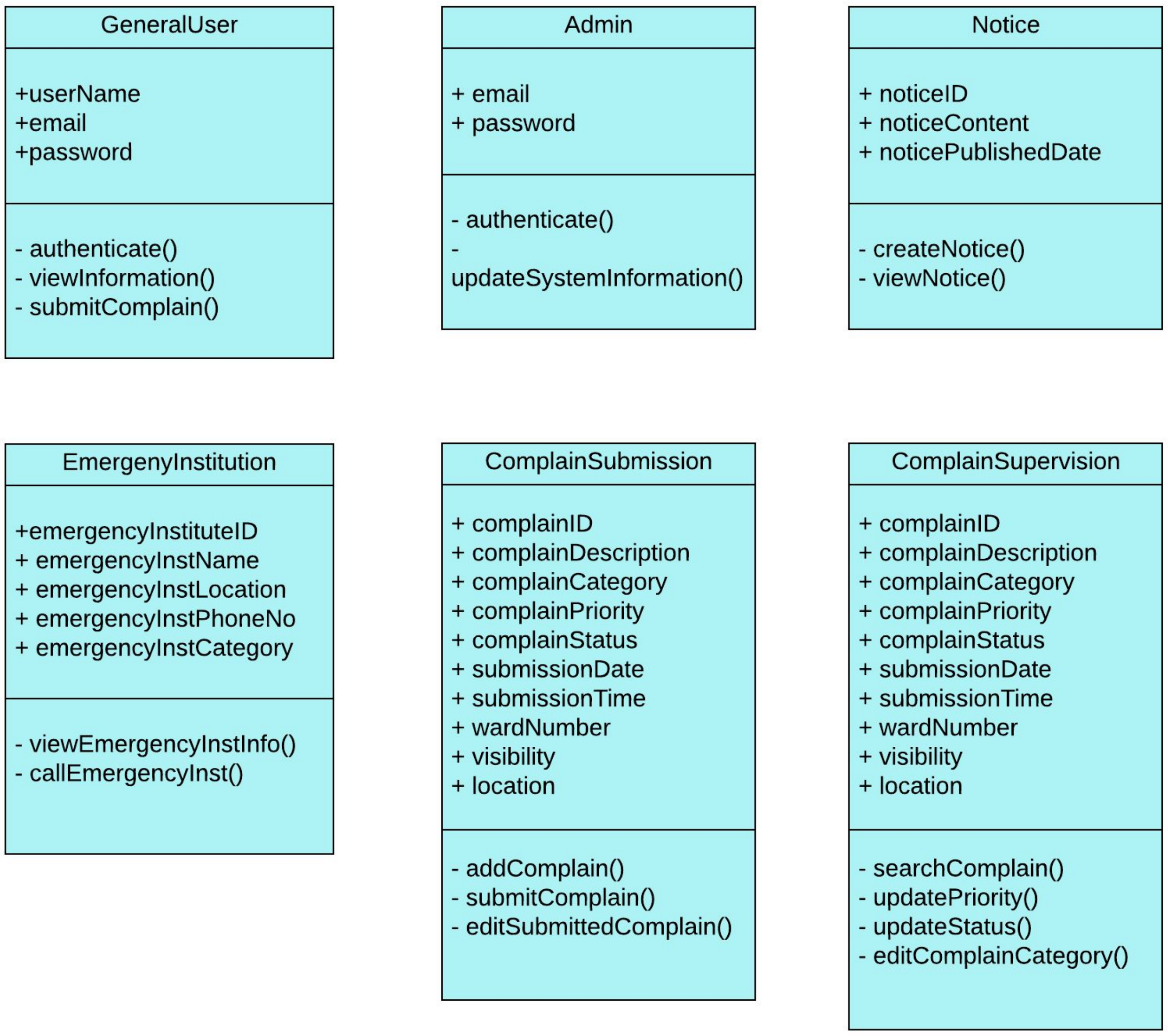


Figure 5: Analysis Classes of *Complain Box*

2.2 Elaboration of Analysis Classes

2.2.1 Admin Class

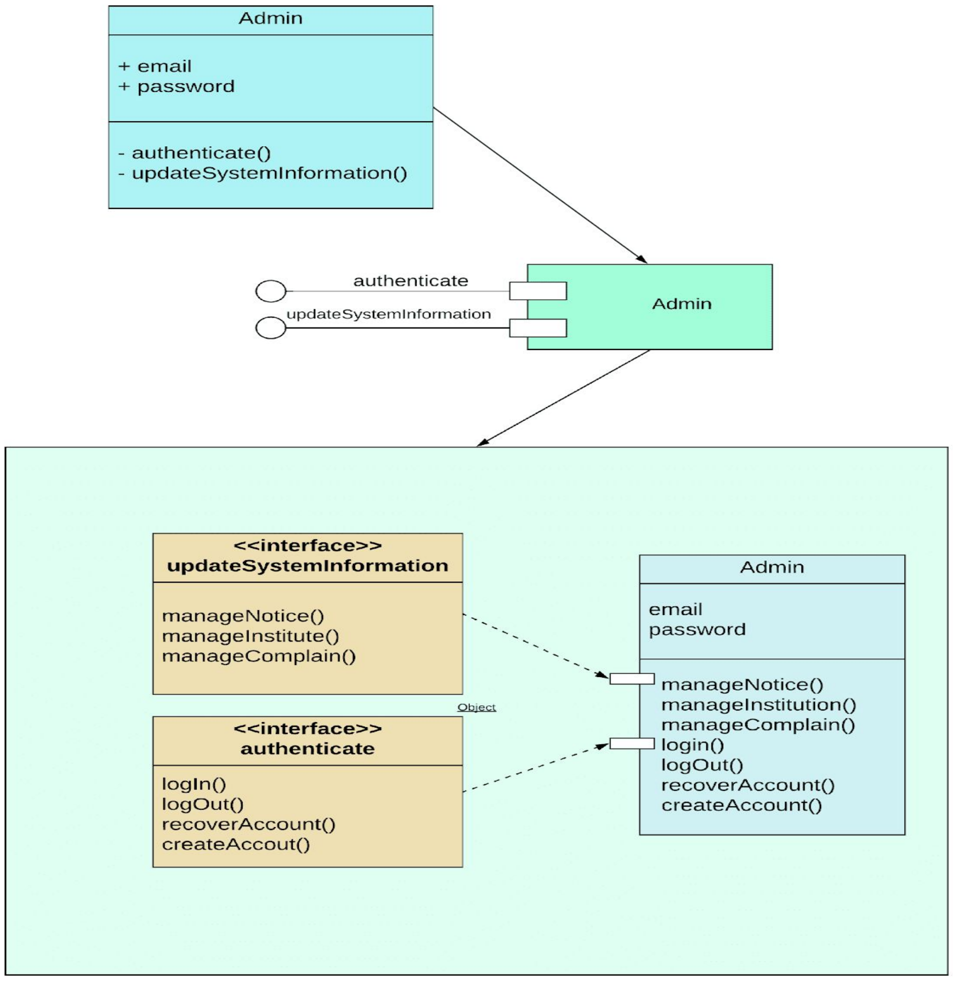


Figure 6:Elaboration of Design Component **Admin**

Admin class can be considered under **User** archetype.
Admin has two interfaces, *authenticate,* Which provides admin's authentication to the system by signing in and logging in and *updateSystemInformation,* which enables admin to manage notice, institution and complain by updating them when necessary. These are represented using "lollipop" symbols shown to the left of the component box.

2.2.2 General User Class

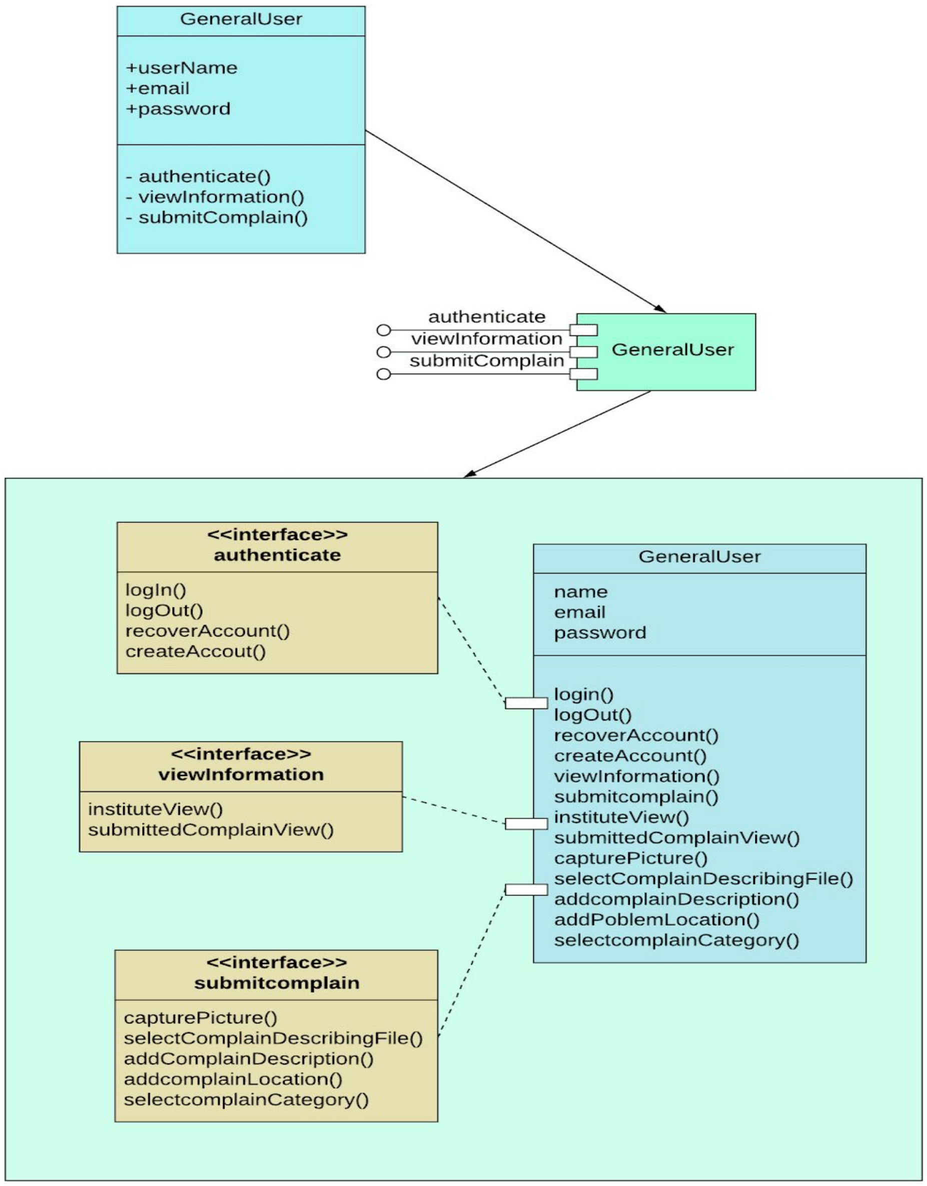


Figure 7:Elaboration of Design Component **General User**

GeneralUser class can be considered under **User** archetype.
General User has three interfaces, *authenticate,* Which provides user's authentication to the system by signing in and logging in , *viewInformation,* which enables admin to view emergency institution information and submitted complains and *submitComplain,* which enables user to capture picture, upload complain describing file as pdf, select complain location and category. These are represented using "lollipop" symbols.

2.2.3 Complain Submission Class

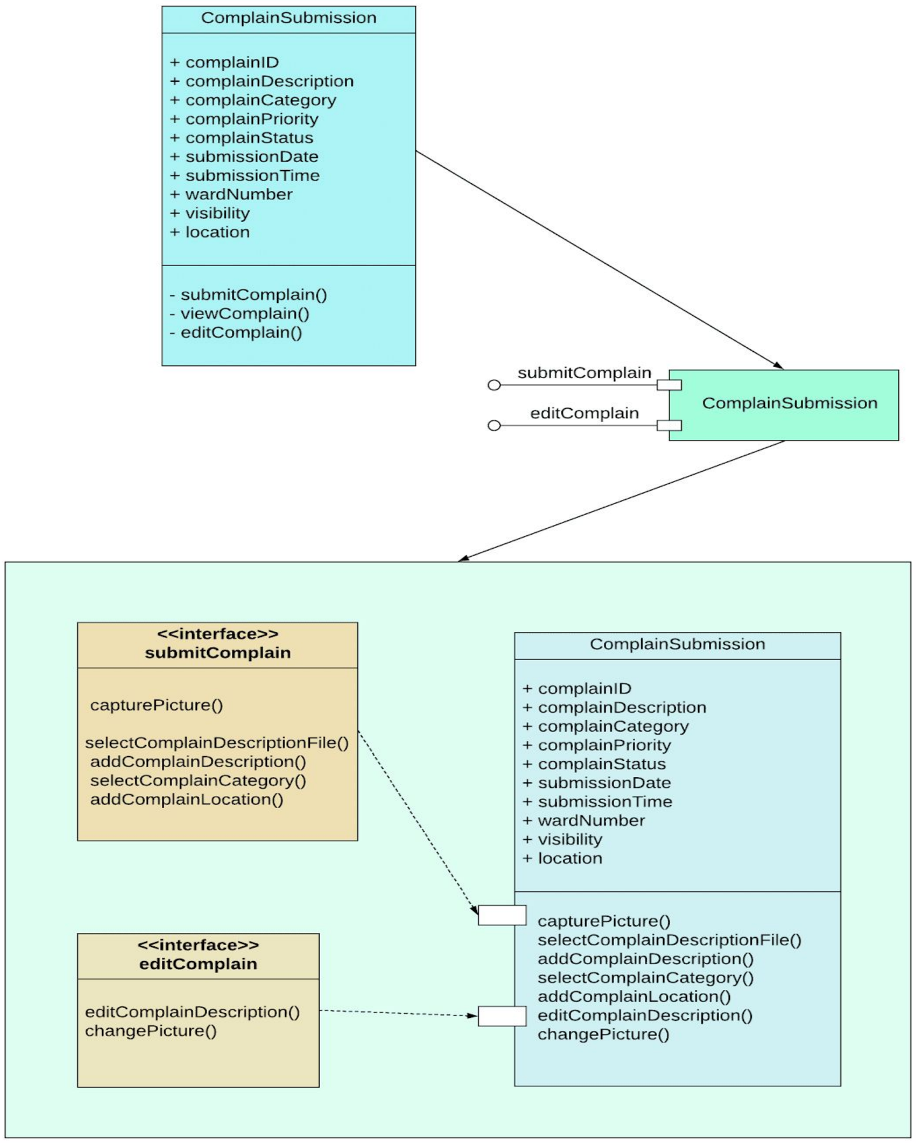


Figure 8:Elaboration of Design Component **Complain Submission**

ComplainSubmission class can be considered under **Complain** archetype.
Complain Submission has two interfaces, *submitComplain,* which enables user to capture picture, select complain file as pdf, add complain description, location and category, and *editComplain,* which gives user interface to edit any submitted complains description and change picture of the submitted complain. These are represented using “lollipop” symbols.

2.2.4 Complain Supervision Class

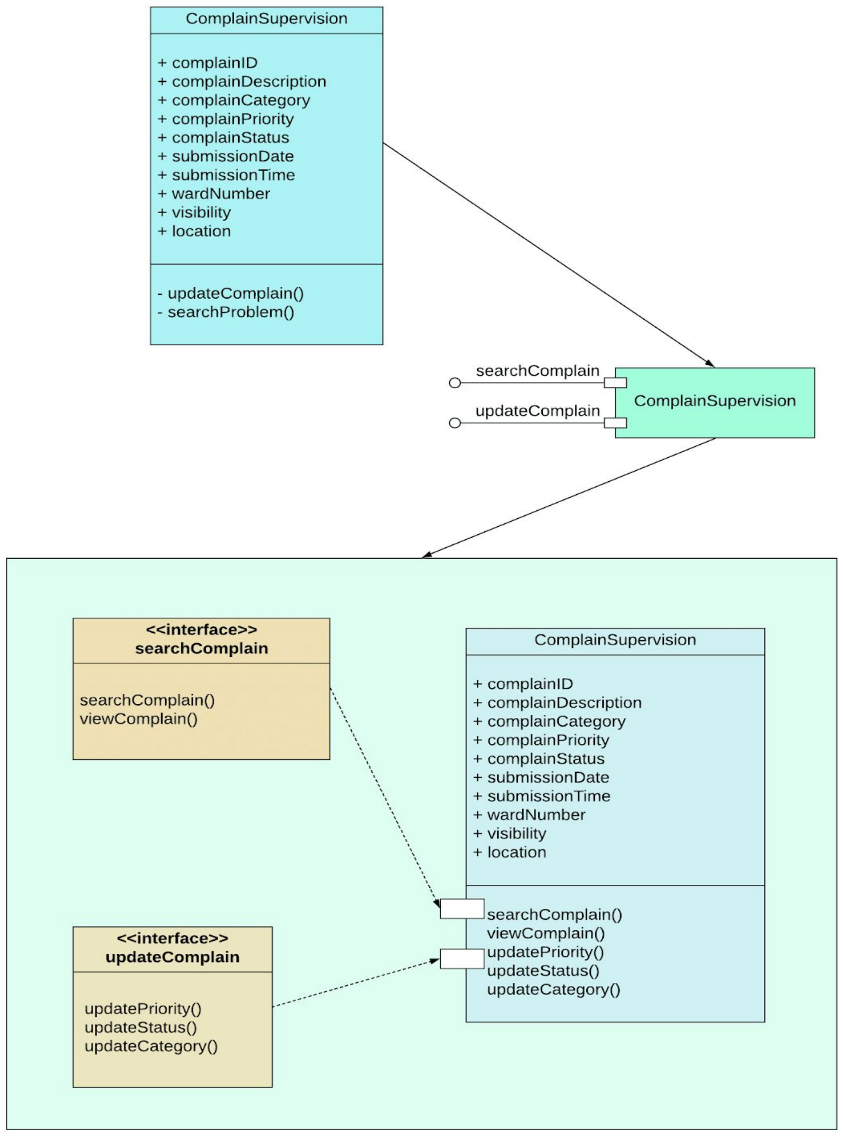


Figure 9:Elaboration of Design Component **Complain Supervision**

ComplainSupervision class can be considered under **Complain** archetype.
Complain Supervision has two interfaces, *searchComplain,* which enables search and view complains in various category such as- solved complains, complain in progress and complain under construction, and *updateComplain,* which gives user interface to update priority, category and status of the complains. These are represented using “lollipop” symbols.

2.3 Collaboration between Classes or Components

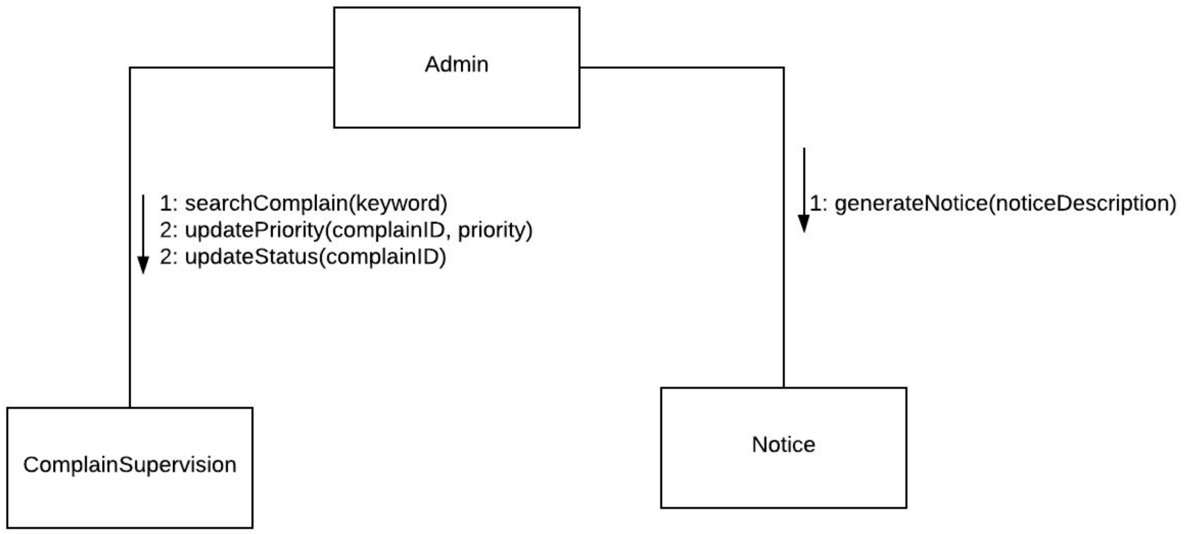

Admin
1: searchComplain(keyword)
2: updatePriority(complainID, priority)
2: updateStatus(complainID)
1: generateNotice(noticeDescription)
ComplainSupervision
Notice

Figure 10:Collaboration Diagram with messaging

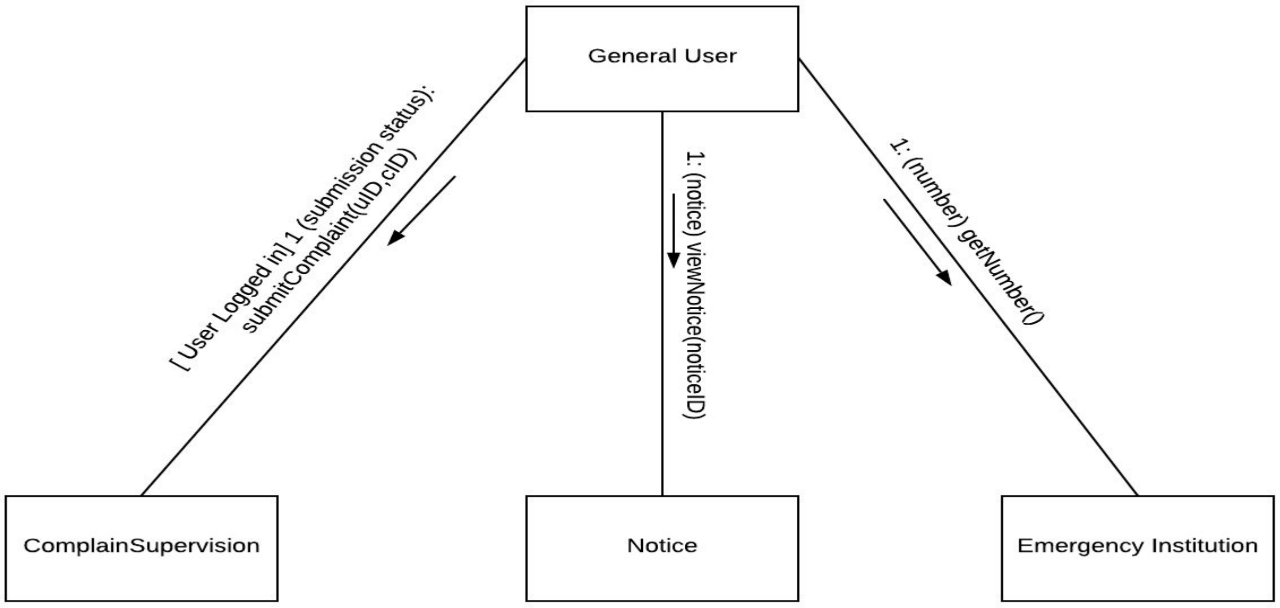
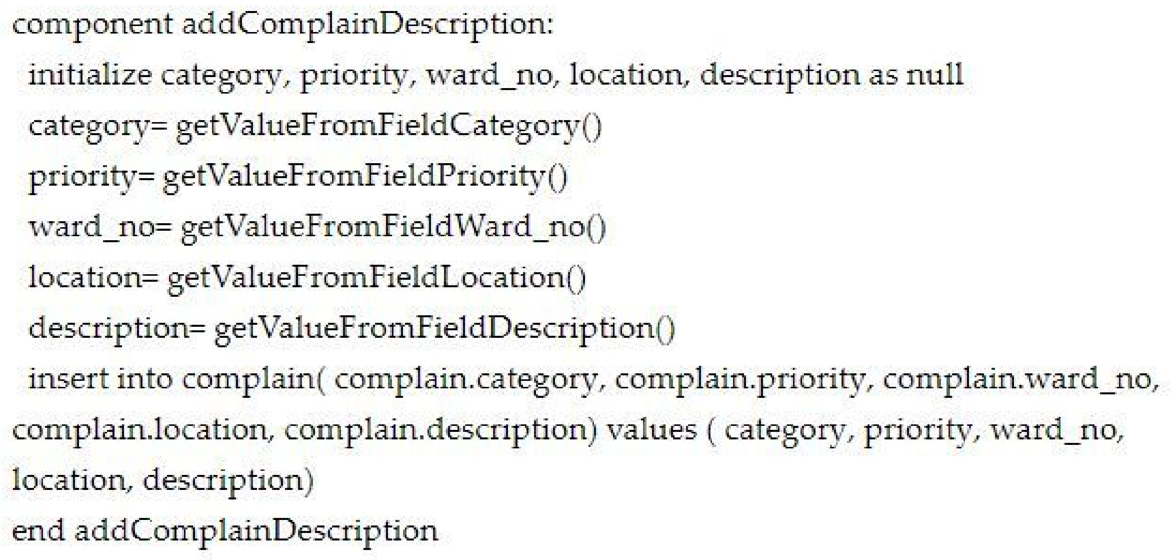


Figure 11:Collaboration Diagram with messaging

2.4 Component Based Development

2.4.1 Complain Submission

For class ComplainSubmission, a user can add complain description and submit the complain. So addComplainDescription is an important component. Where user is given a complain form and the values from the fields are fetched. Then the values are stored in the database.

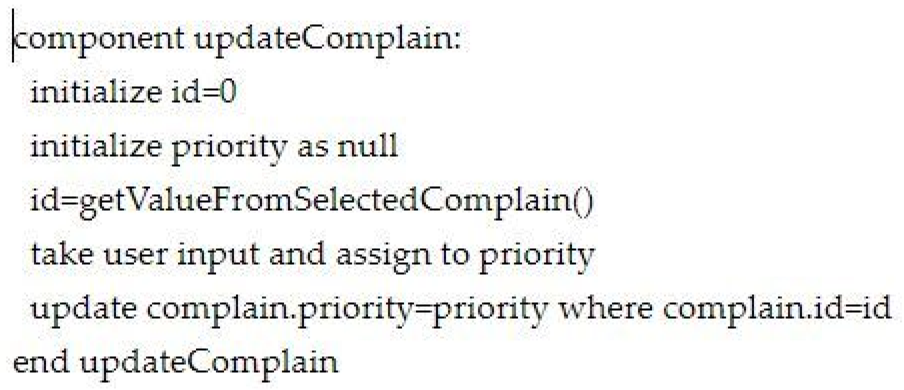


```
component addComplainDescription:
  initialize category, priority, ward_no, location, description as null
  category= getValueFromFieldCategory()
  priority= getValueFromFieldPriority()
  ward_no= getValueFromFieldWard_no()
  location= getValueFromFieldLocation()
  description= getValueFromFieldDescription()
  insert into complain( complain.category, complain.priority, complain.ward_no,
complain.location, complain.description) values ( category, priority, ward_no,
location, description)
end addComplainDescription
```

Figure 12: Pseudo Code of Component *addComplainDescription* from class *complainSubmission*

2.4.2 Complain Supervision

Under complain supervision, component updateComplain exists. In this component admin selects a complain. The id of the selected component is fetched. Then admin gives priority as input and the new priority is updated and stored in database.



```
component updateComplain:
  initialize id=0
  initialize priority as null
  id=getValueFromSelectedComplain()
  take user input and assign to priority
  update complain.priority=priority where complain.id=id
end updateComplain
```

Figure 13: Pseudo Code of Component *updateComplain* from class *complainSupervision*

2.4.3 General User

The general user class enables user to create account and then login and submit complain. We have selected the component create account here. The procedure is to take name, phone no and password as user input and check validity and sif valid then store them into database.

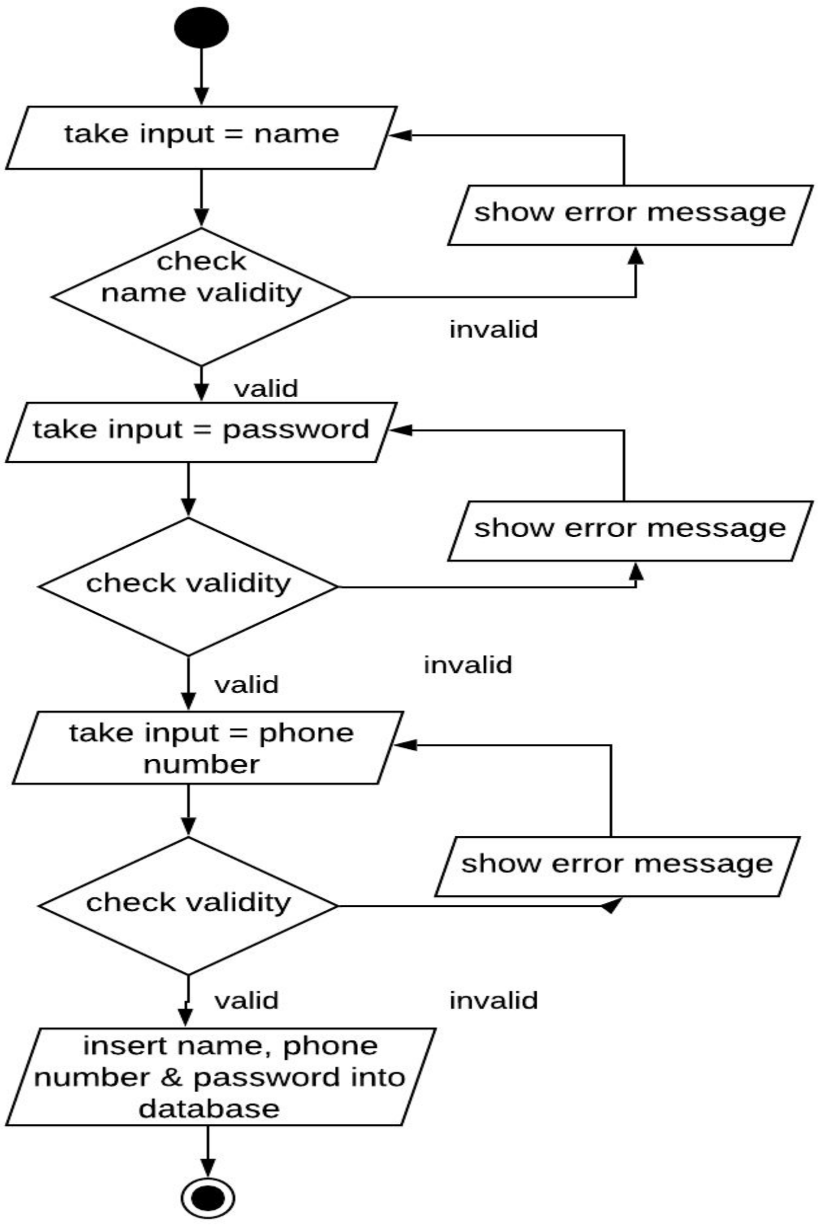
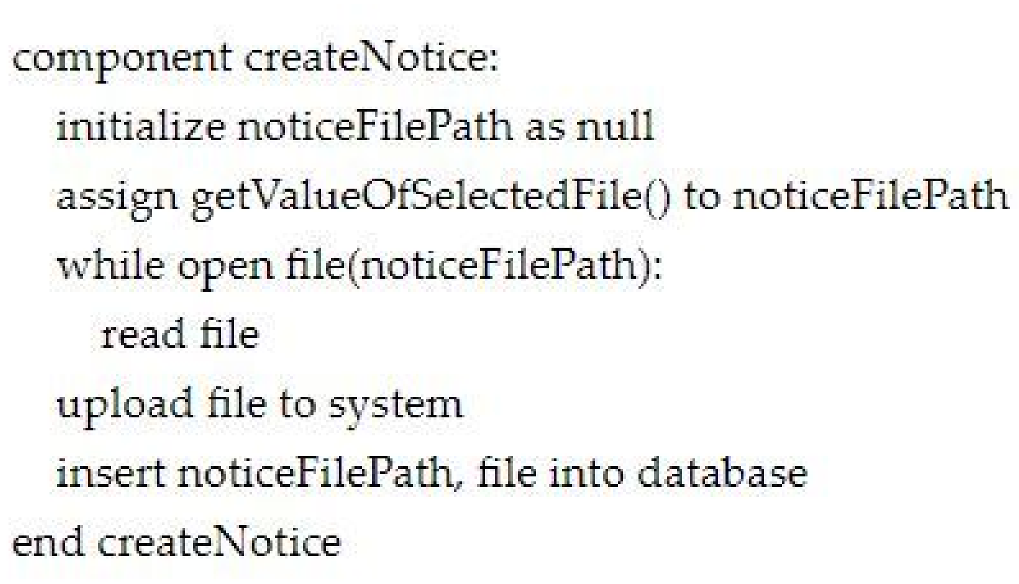


Figure 14: Flowchart of Component *createAccount* from class *generalUser*

2.4.4 Notice

Class Notice enables admin to create notice. So in this component, admin can upload a file from computer as notice file. So the after selecting the file, the file is read and uploaded to the system and stored in database

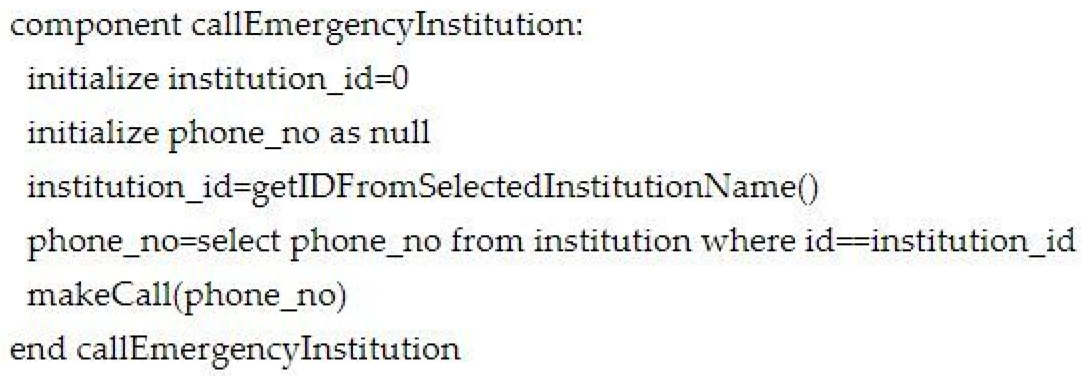


```
component createNotice:
    initialize noticeFilePath as null
    assign getValueOfSelectedFile() to noticeFilePath
    while open file(noticeFilePath):
        read file
    upload file to system
    insert noticeFilePath, file into database
end createNotice
```

Figure 15: Pseudo code of Component *createNotice* from class *Notice*

2.4.5 Emergency Institution

Class Emergency Institution enables user to call to the institution and see information (location) of the institute. Here pseudo code of component callEmergencyInstitution is created. Where, the user selects an emergency institute and the system finds the phone no of the institute and initialize a calling interface to make a call.

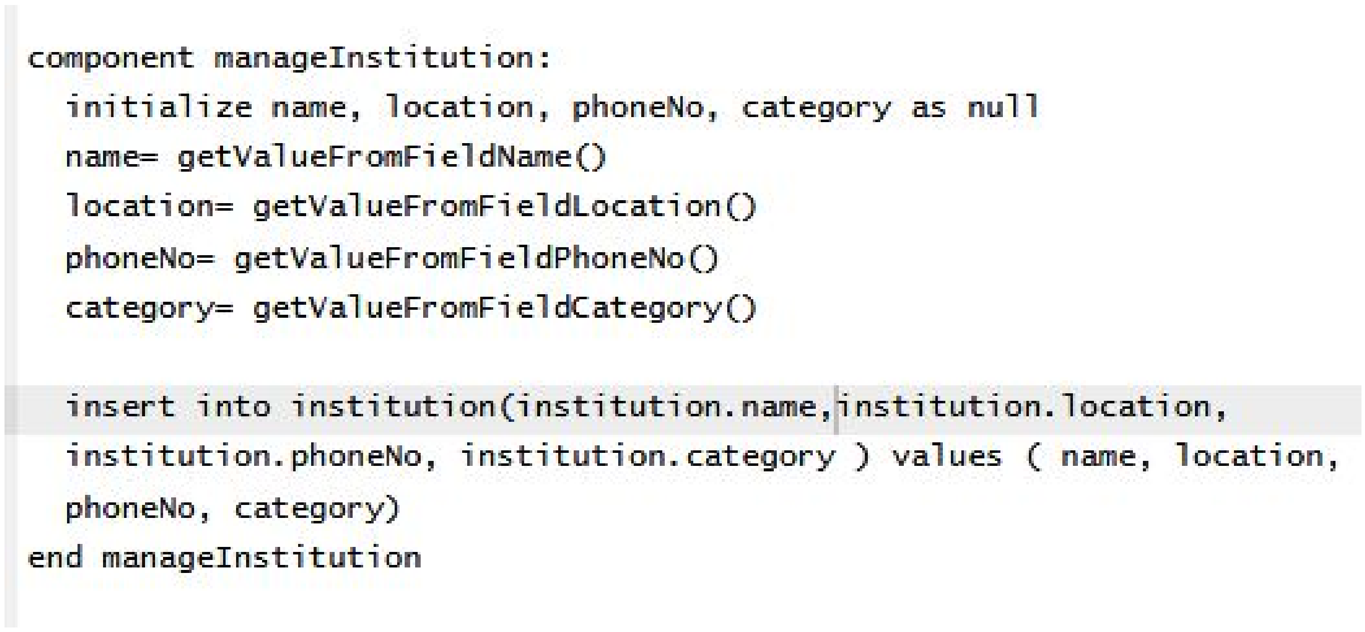


```
component callEmergencyInstitution:
  initialize institution_id=0
  initialize phone_no as null
  institution_id=getIDFromSelectedInstitutionName()
  phone_no=select phone_no from institution where id==institution_id
  makeCall(phone_no)
end callEmergencyInstitution
```

Figure 16: Pseudo code of Component *callEmergencyInstituion* from class *EmergencyInstitution*

2.4.5 Admin

Admin can manageInstitution by adding an emergency institute. To add an emergency institute , admin has to fillUp form with name, location, phoneNo and category. So the values of the fields are get through the getter and store the values into database.



```
component manageInstitution:
  initialize name, location, phoneNo, category as null
  name= getValueFromFieldName()
  location= getValueFromFieldLocation()
  phoneNo= getValueFromFieldPhoneNo()
  category= getValueFromFieldCategory()

  insert into institution(institution.name,institution.location,
  institution.phoneNo, institution.category ) values ( name, location,
  phoneNo, category)
end manageInstitution
```

Figure 17: Pseudo code of Component *manageInstituion* from class *Admin*

2.5 State Chart Fragment for Classes

2.5.1 Admin Class :

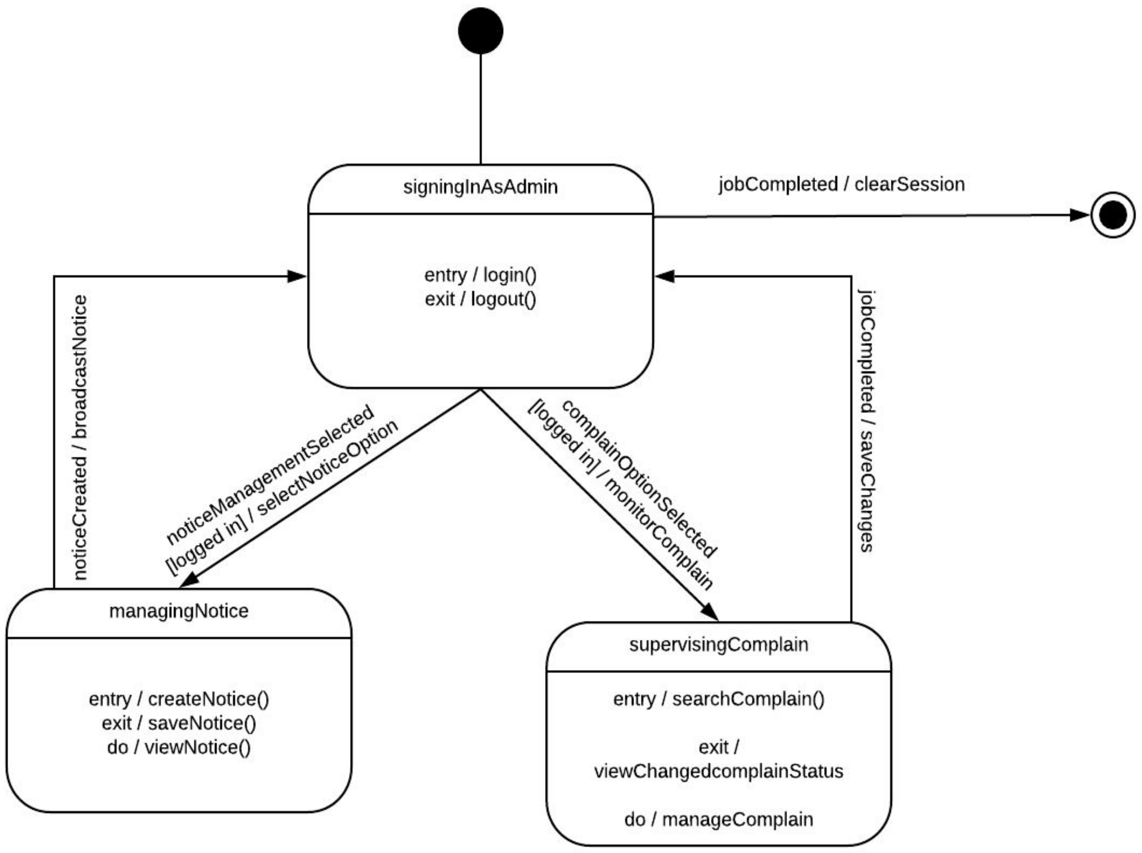

signingInAsAdmin
entry / login()
exit / logout()
jobCompleted / clearSession
noticeCreated / broadcastNotice
jobCompleted / saveChanges
noticeManagementSelected [logged in] / selectNoticeOption
complainOptionSelected [logged in] / monitorComplain
managingNotice
entry / createNotice()
exit / saveNotice()
do / viewNotice()
supervisingComplain
entry / searchComplain()
exit / viewChangedcomplainStatus
do / manageComplain

Figure 18: Statechart fragment for Admin Class

2.5.2 GeneralUser Class :

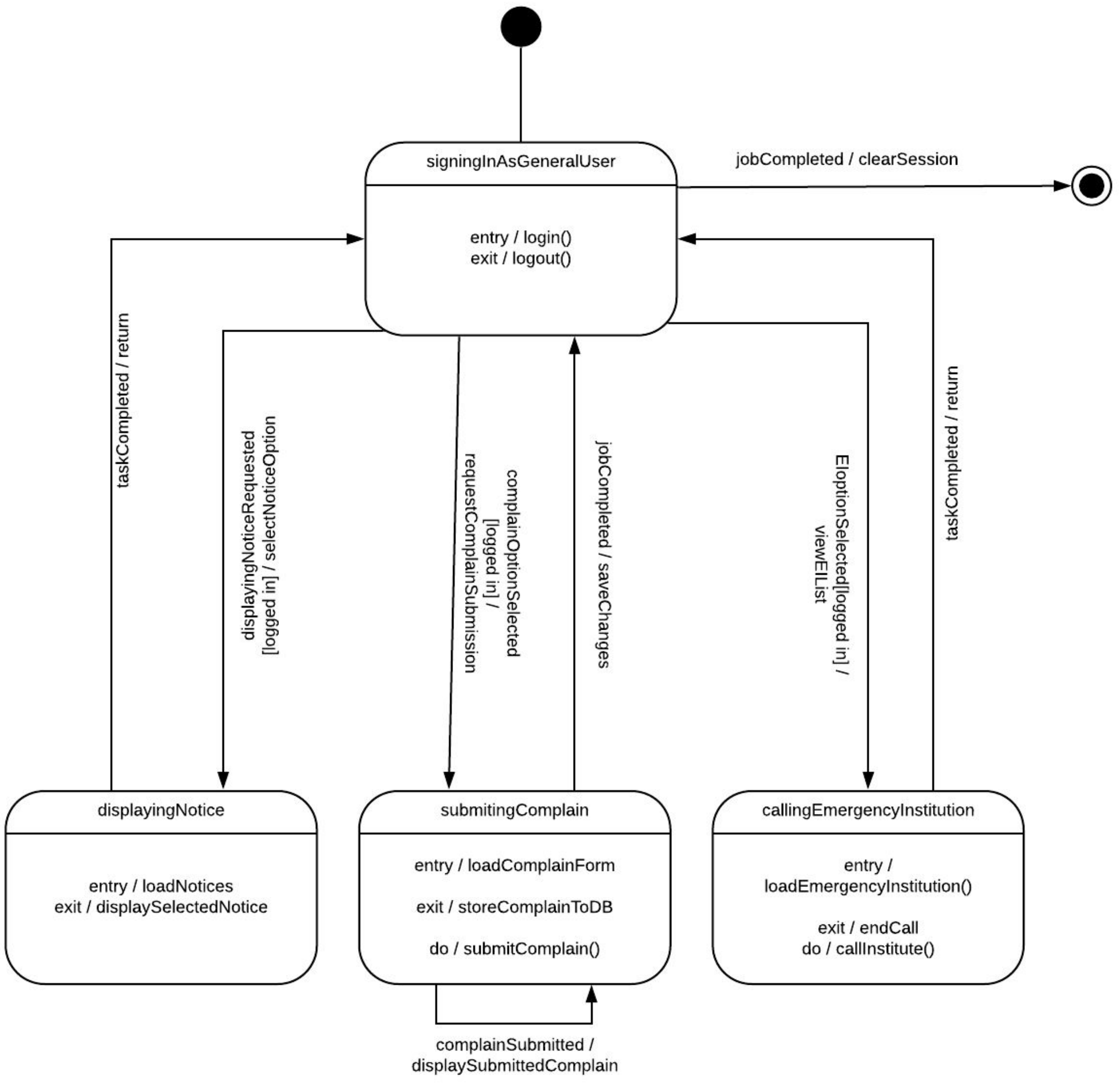

signingInAsGeneralUser
entry / login()
exit / logout()
jobCompleted / clearSession
taskCompleted / return
displayingNoticeRequested [logged in] / selectNoticeOption
complainOptionSelected [logged in] / requestComplainSubmission
jobCompleted / saveChanges
EIoptionSelected[logged in] / viewEIList
taskCompleted / return
displayingNotice
entry / loadNotices
exit / displaySelectedNotice
submitingComplain
entry / loadComplainForm
exit / storeComplainToDB
do / submitComplain()
callingEmergencyInstitution
entry / loadEmergencyInstitution()
exit / endCall
do / callInstitute()
complainSubmitted / displaySubmittedComplain

Figure 19: Statechart fragment for General User Class

2.5.3 ComplainSubmission Class :

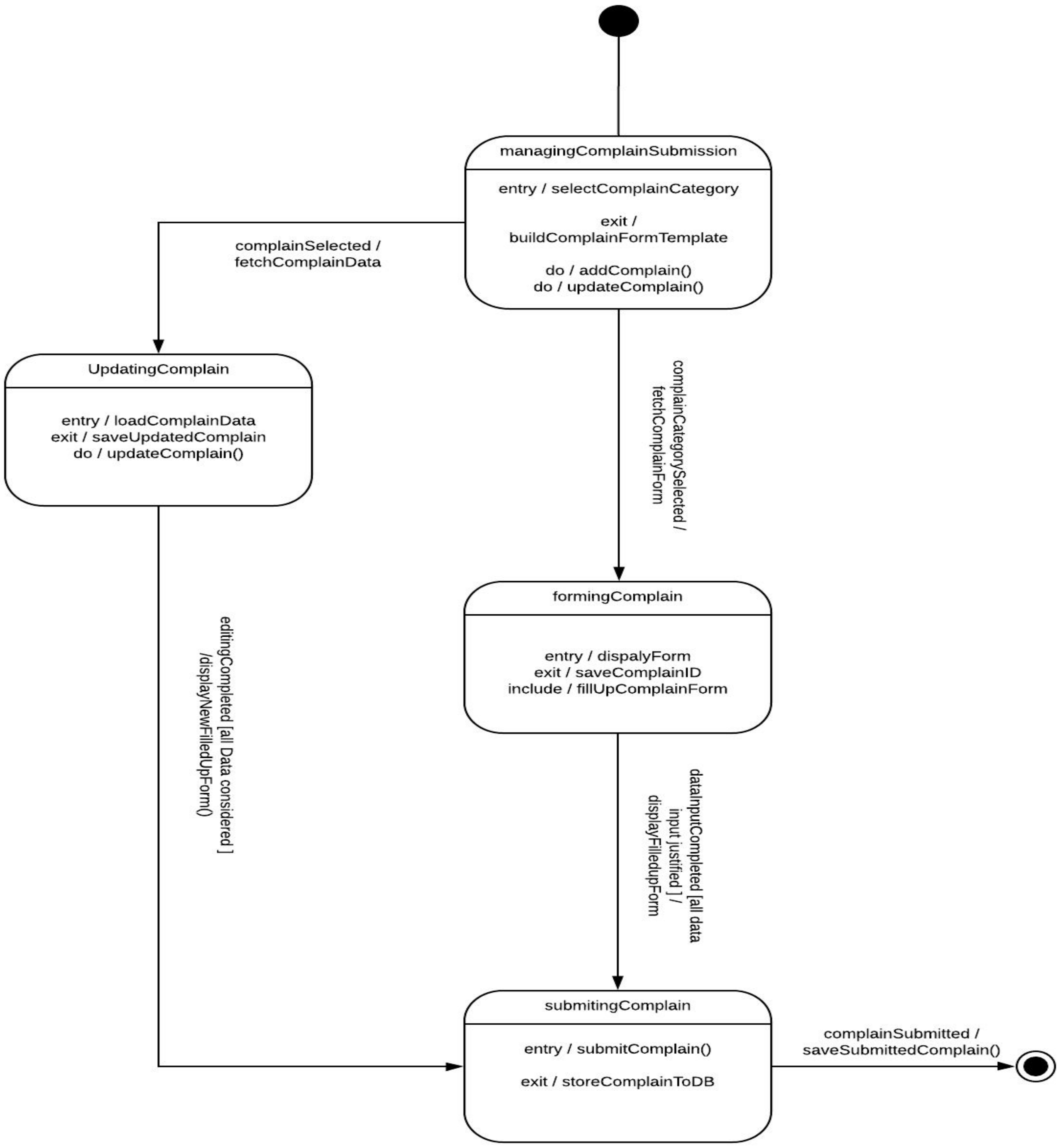

managingComplainSubmission
entry / selectComplainCategory
exit / buildComplainFormTemplate
do / addComplain()
do / updateComplain()
complainSelected / fetchComplainData
UpdatingComplain
entry / loadComplainData
exit / saveUpdatedComplain
do / updateComplain()
complainCategorySelected / fetchComplainForm
formingComplain
entry / dispalyForm
exit / saveComplainID
include / fillUpComplainForm
editingCompleted [all Data considered] /displayNewFilledUpForm()
dataInputCompleted [all data input justified] / displayFilledupForm
submitingComplain
entry / submitComplain()
exit / storeComplainToDB
complainSubmitted / saveSubmittedComplain()

Figure 20: Statechart fragment for ComplainSubmission Class

2.5.4 ComplainSupervision Class :

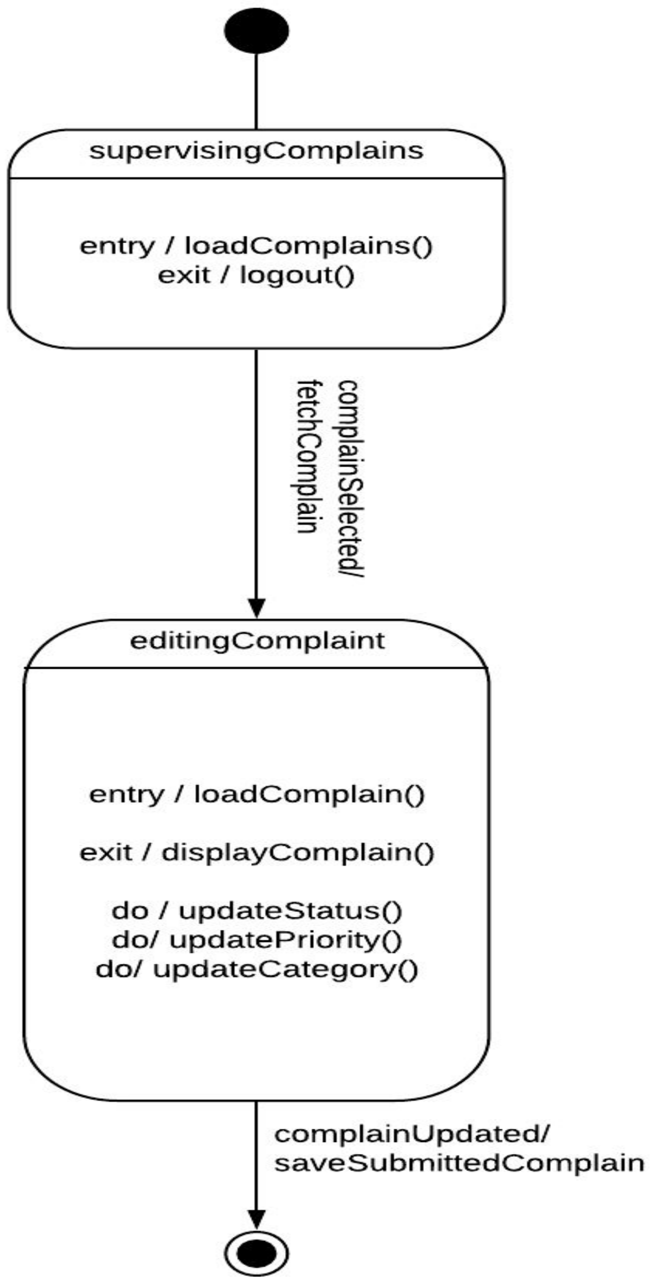


Figure 21: Statechart fragment for ComplainSupervision Class

2.5.5 Notice Class :

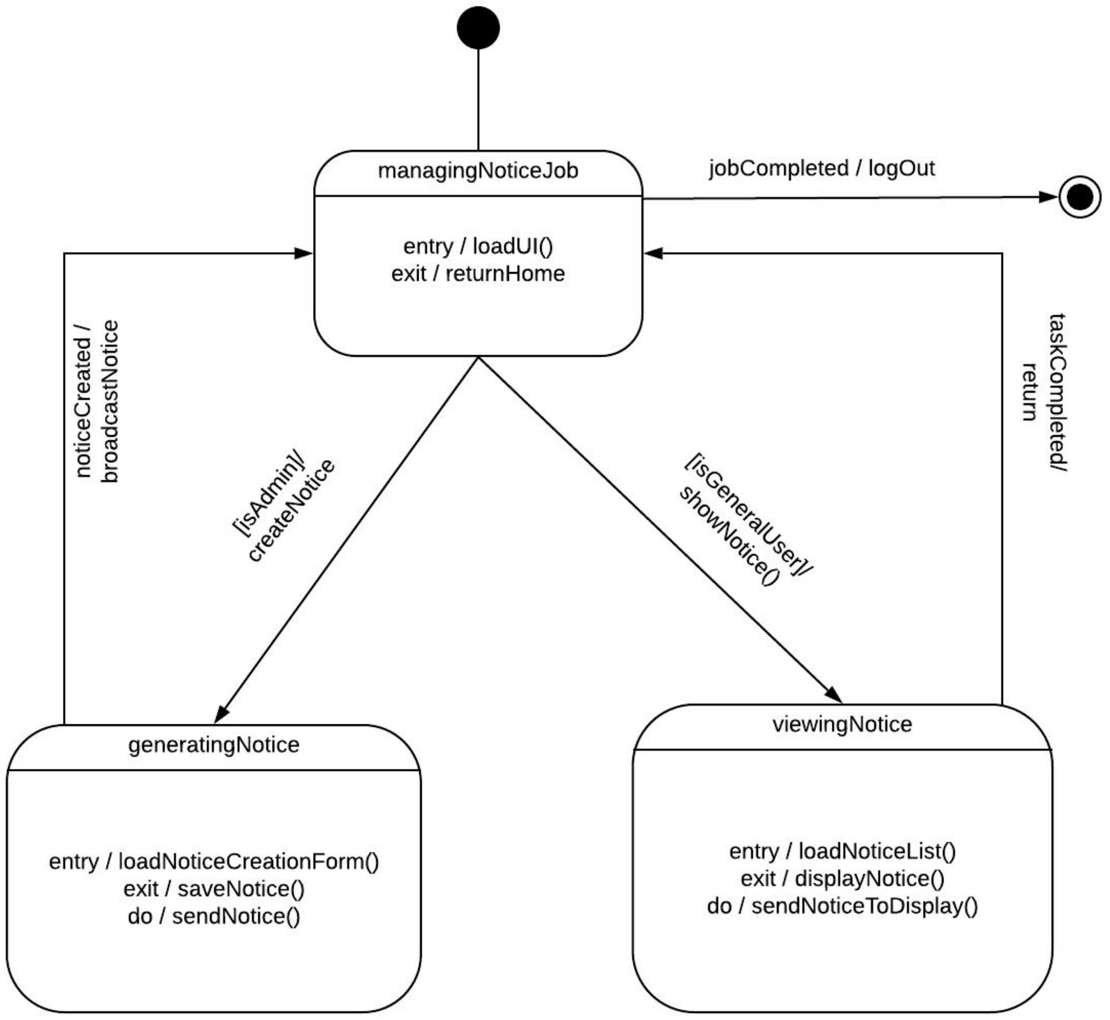


Figure 22: Statechart fragment for Notice Class

2.5.4 EmergencyInstitution Class :

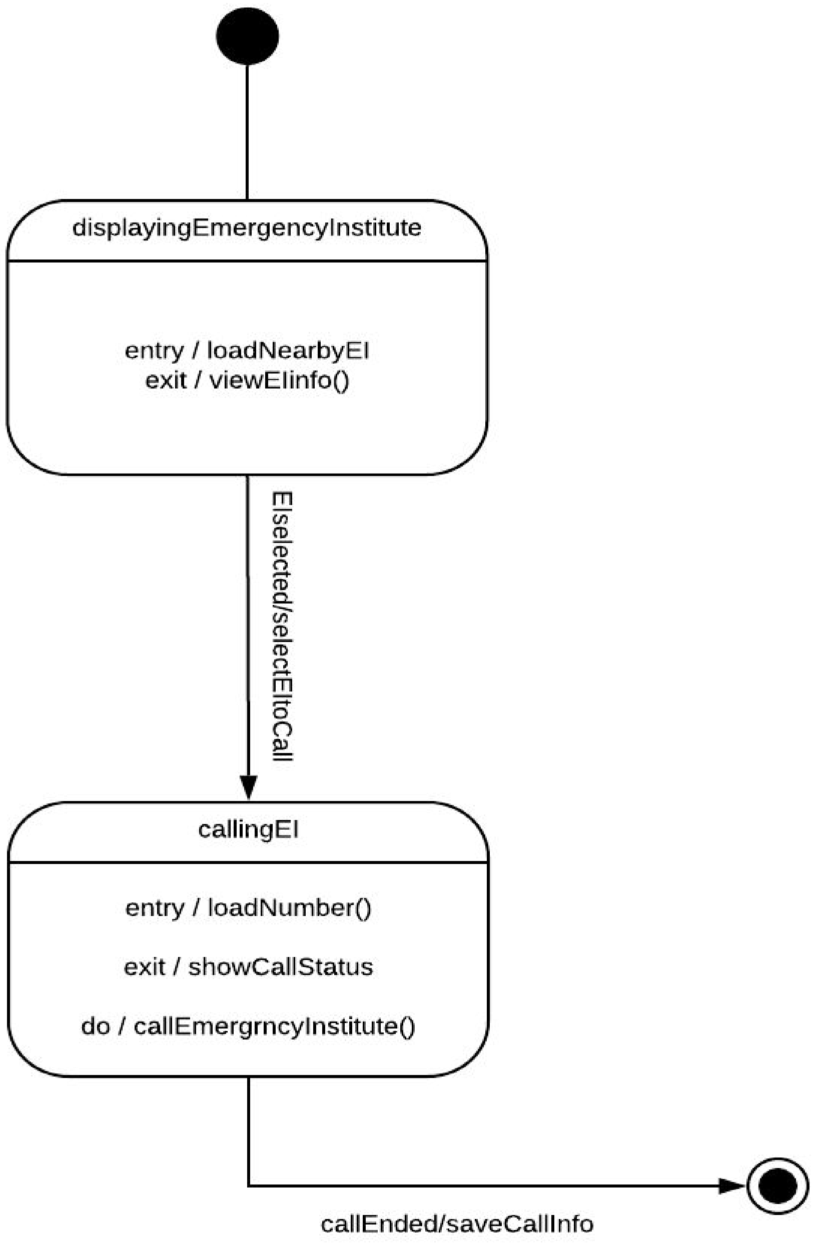


Figure 23: Statechart fragment for EmergencyInstitution Class

2.6 Deployment Diagram

Deployment-level design elements indicate how software functionality and subsystems
will be allocated within the physical computing environment that will support the software.
The elements of the *Complain Box* are configured to operate within three primary computing environments- A server for complain box, A personal computer for Admin, A general user pc.
The General User PC subsystem implements Complain Submission, Notice and Emergency Institution. In addition, an external access subsystem has been designed to manage all attempts to access the *Complain Box* from an external source.
The Personal Computer for Admin subsystem implements Complain Supervision and Notice. An external access subsystem has been designed to access Server from Admin.

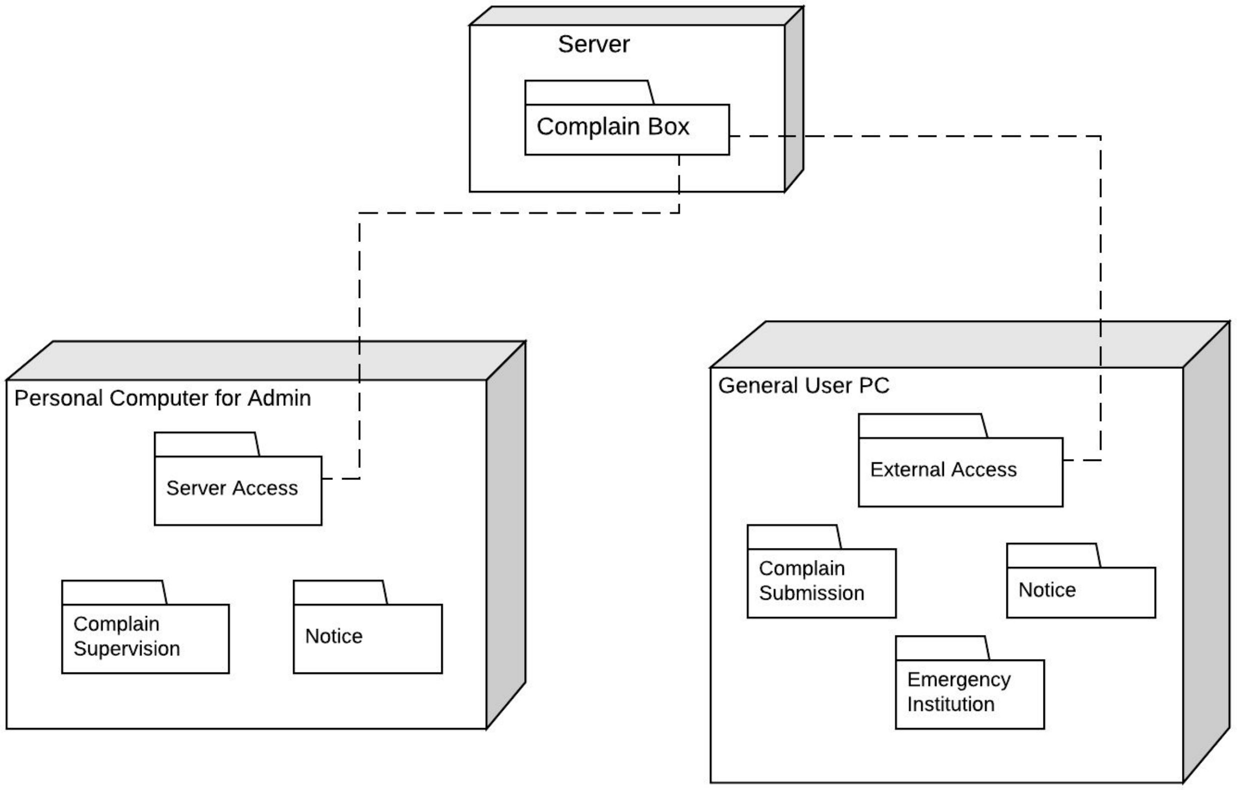


Figure 24: A UML Deployment Diagram

Chapter Three

Interface Analysis

User Analysis:

To design user interface first of all, we have to select the users. From requirement analysis, we have two types of user- General User and admin.
To understand the user of complain box, we have understood the following things:

- Users can be both educated and uneducated
- For General user, it can not be guaranteed if they will have technical knowledge or not. Admins can have some technical knowledge.
- The users are capable of learning from written materials.
- The users are not keyboard phobic
- The age range of the users is greater than 18
- The users can be of any gender
- The primary spoken language for the user is Bangla
- If user makes any mistake using *Complain Box,*they can undo the complain
- The admins are moderate in the subject matters that is addressed by the *Complain Box*
- User does not want to know about the technology that sits behind the interface

Task Analysis:

- The admin type user will supervise complains , manage notice and manage institutions. The general user will submit complain
- For admin to Supervise the complain these subtasks will be performed-
  - update complain Category
  - Update complain status
  - Update complain priority
  - View solved complains
  - View complain under construction
- For general user to Submit a complain these subtasks will be performed-
  - Add complain details
  - Capture picture
  - Edit submitted complain
- The user will manipulate as work is performed - a complain form by adding data into database

Workflow:

From the perspective of Admin

1. Authentication
2. Complain Supervision
    a. Update complain status
    b. Update complain priority
    c. Update complain category
3. Notice Management
    a. Add notice
4. Emergency Institute Management
    a. Add emergency institution
5. Logout

From the perspective of General User

1. Authentication
2. Complain submission
    a. Add complain description
    b. View submitted complain
    c. Edit submitted complain
3. Emergency institution
    a. Call emergency institution
    b. View location
4. Logout