IIT
University of Dhaka

Component Level Design-”Srenikokkho”

Submitted To

Md. Nurul Ahad Tawhid

Assistant Professor

Submitted By

Group Members

Tahlil-803

Saara Sheneen-833

Institute of Information Technology

Submission Date:4-10-18

Table Of Contents

Steps for conducting component level design are given below:

1. Identify all design classes that correspond to the problem domain as defined in the analysis model and architectural model

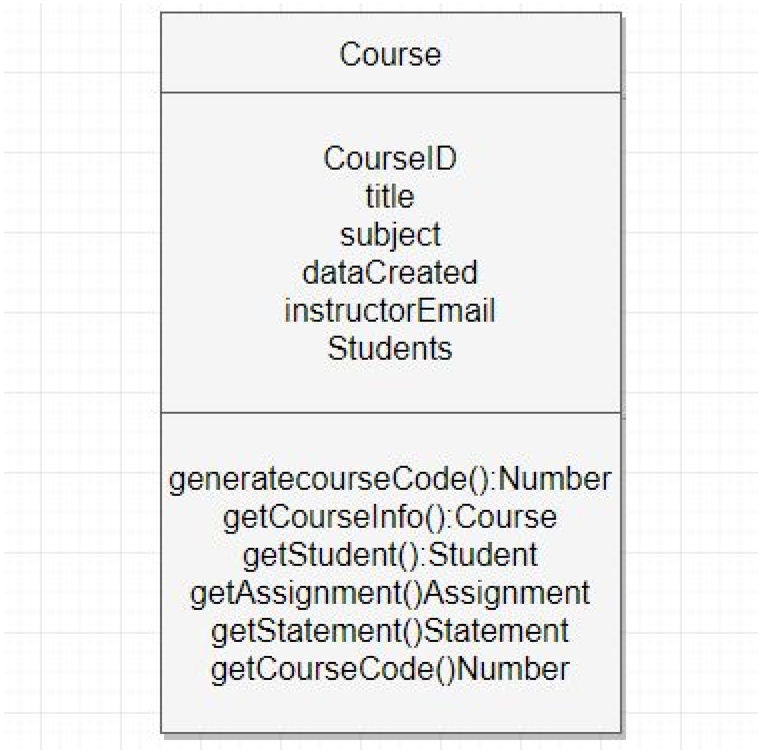


Fig:1

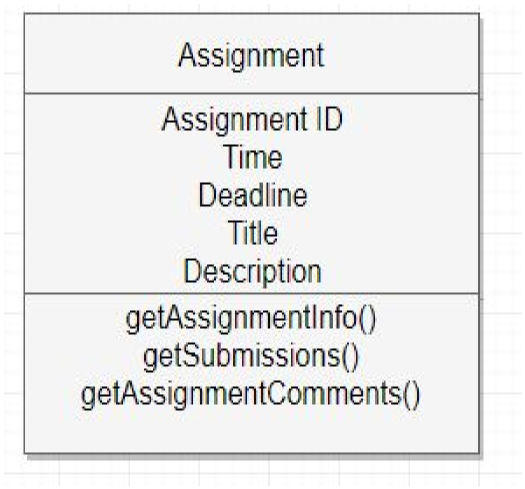


Fig:2

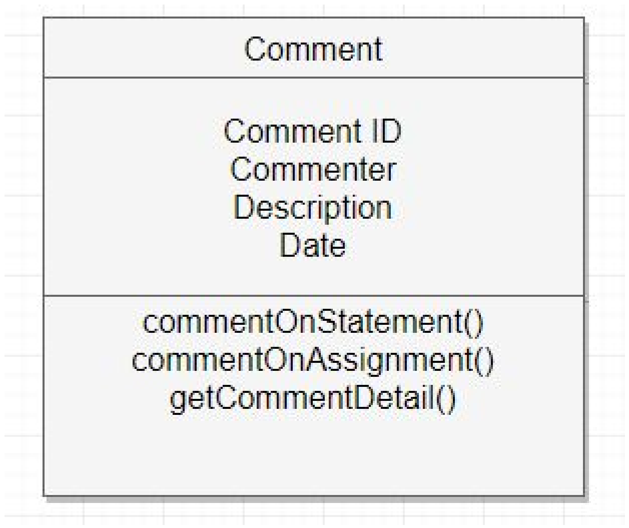


Fig:3

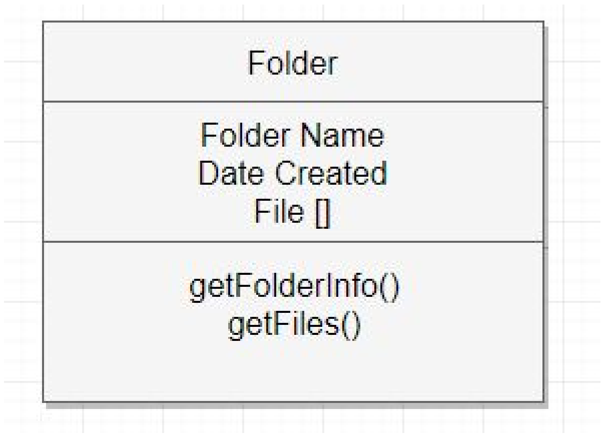


Fig:4

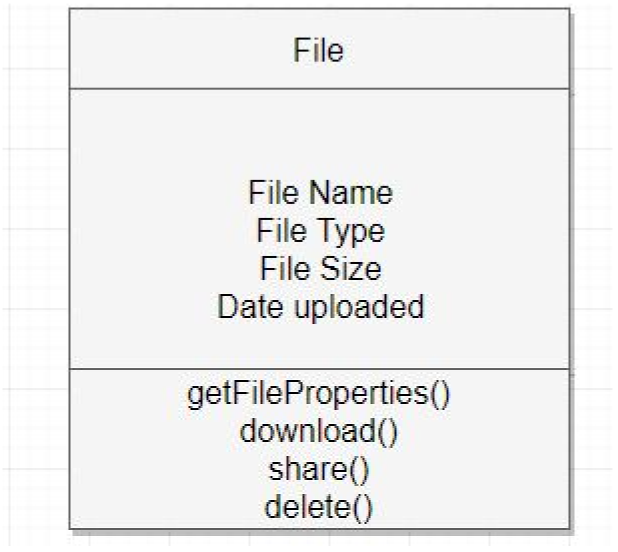

File
File Name
File Type
File Size
Date uploaded
getFileProperties()
download()
share()
delete()

Fig:5

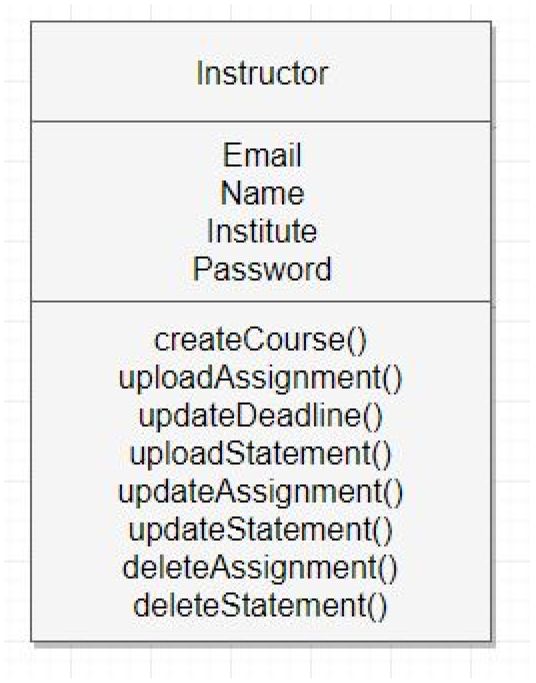

Instructor
Email
Name
Institute
Password
createCourse()
uploadAssignment()
updateDeadline()
uploadStatement()
updateAssignment()
updateStatement()
deleteAssignment()
deleteStatement()

Fig:6

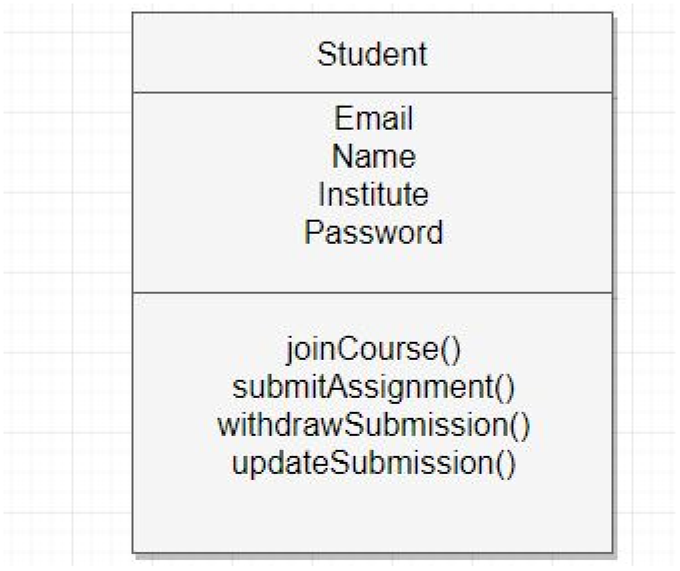

Student
Email
Name
Institute
Password
joinCourse()
submitAssignment()
withdrawSubmission()
updateSubmission()

Fig:7

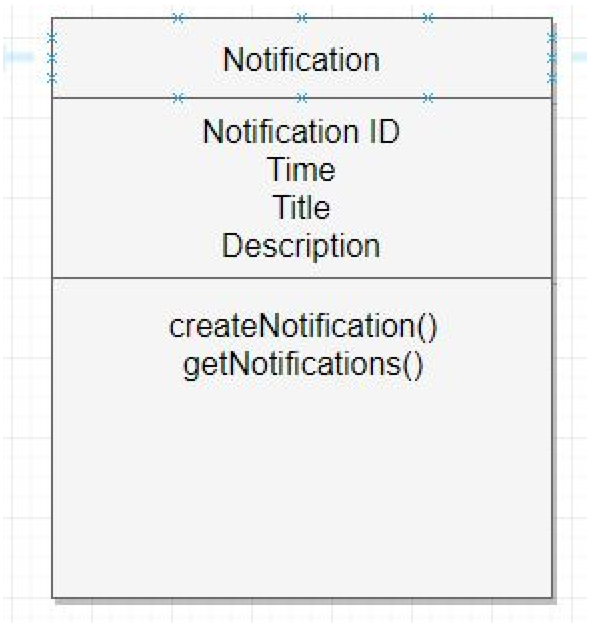

Notification
Notification ID
Time
Title
Description
createNotification()
getNotifications()

Fig:8

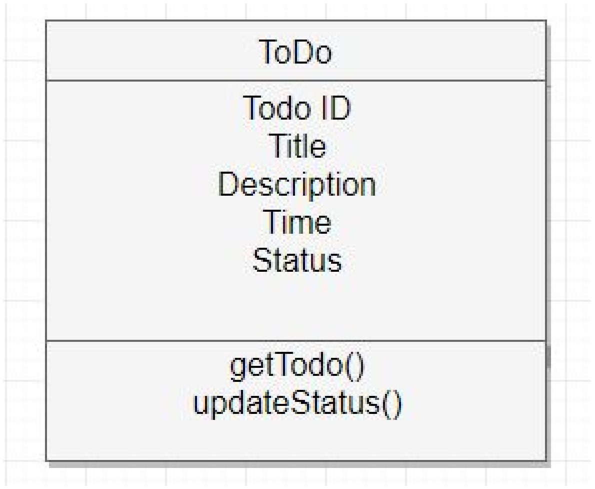

ToDo
Todo ID
Title
Description
Time
Status
getTodo()
updateStatus()

Fig: 9

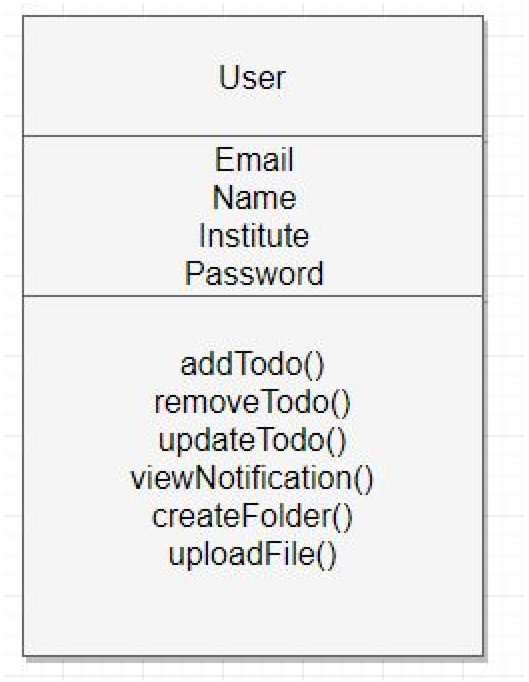

User
Email
Name
Institute
Password
addTodo()
removeTodo()
updateTodo()
viewNotification()
createFolder()
uploadFile()

Fig: 10

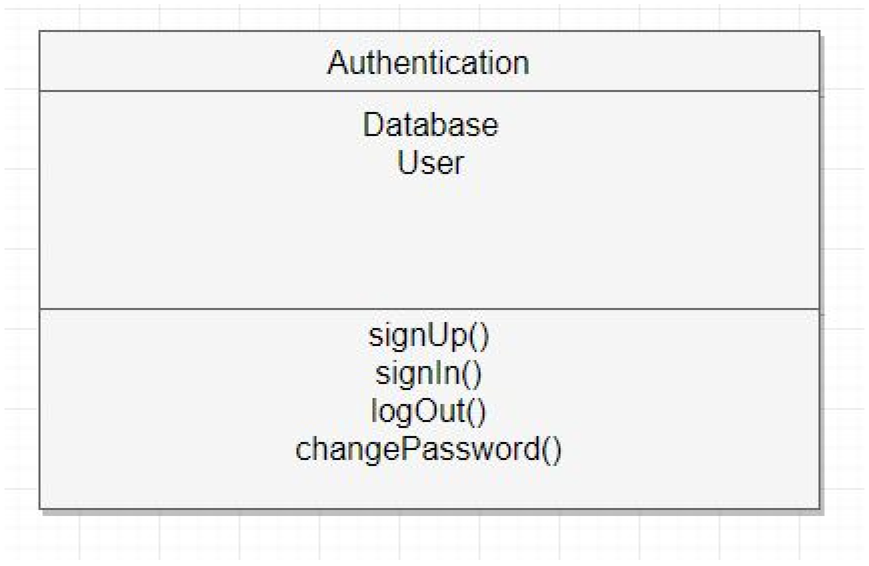


Fig:11

2.Identify all design classes that correspond to the infrastructure domain

- These classes are usually not present in the analysis or architectural models
- These classes include Gui components, operating system components, data management components, networking components etc.

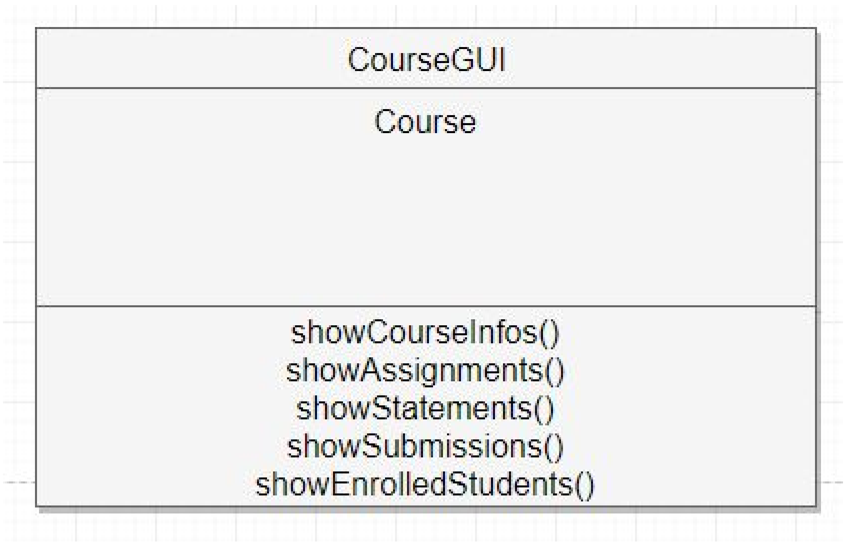

CourseGUI
Course
showCourseInfos()
showAssignments()
showStatements()
showSubmissions()
showEnrolledStudents()

Fig:12

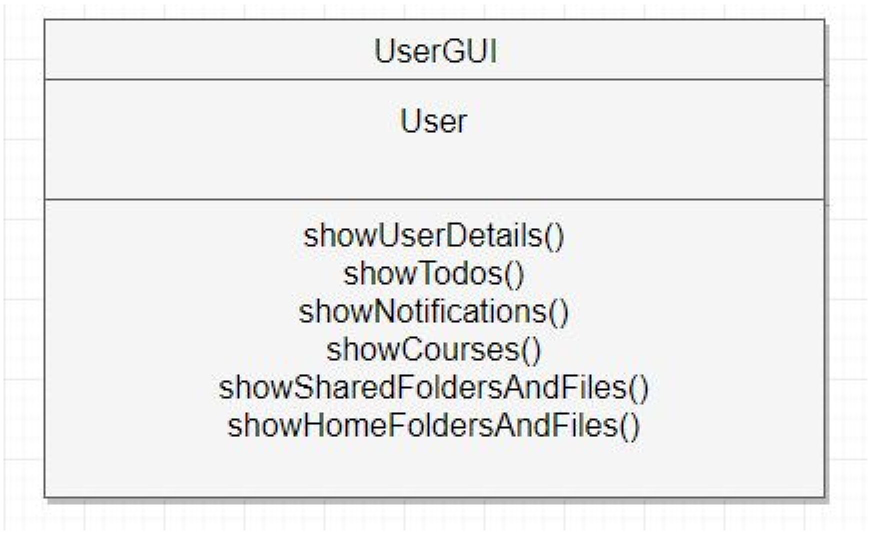

UserGUI
User
showUserDetails()
showTodos()
showNotifications()
showCourses()
showSharedFoldersAndFiles()
showHomeFoldersAndFiles()

Fig:13

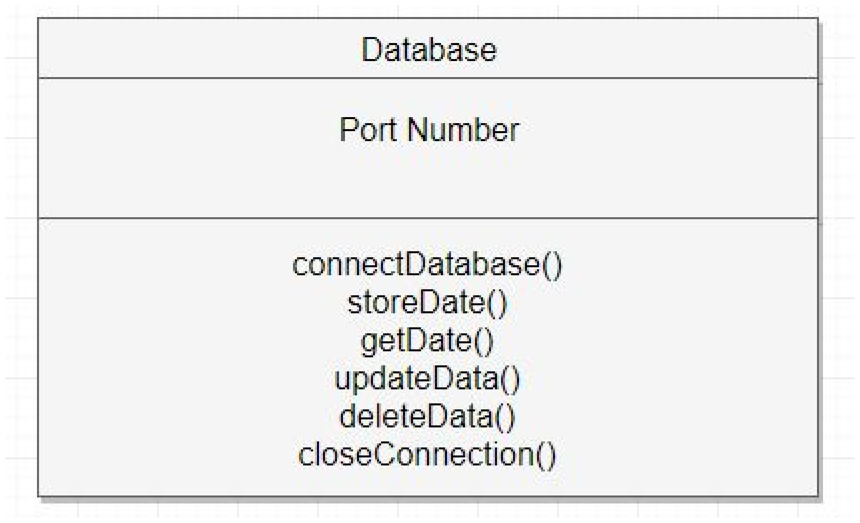

Database
Port Number
connectDatabase()
storeDate()
getDate()
updateData()
deleteData()
closeConnection()

Fig:14

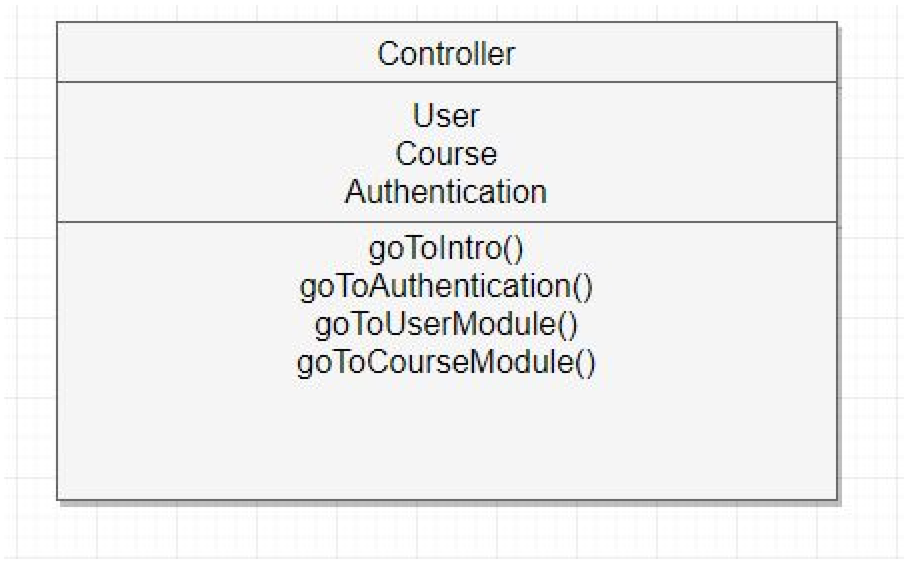

Controller
User
Course
Authentication
goToIntro()
goToAuthentication()
goToUserModule()
goToCourseModule()

Fig:15

3.Elaborate all design classes that are not acquired as reusable components

- **Class collaboration details**

Messages can be elaborated by expanding their syntax in the following manner:

[guard condition] sequence expression (return value) :=message name (argument list)

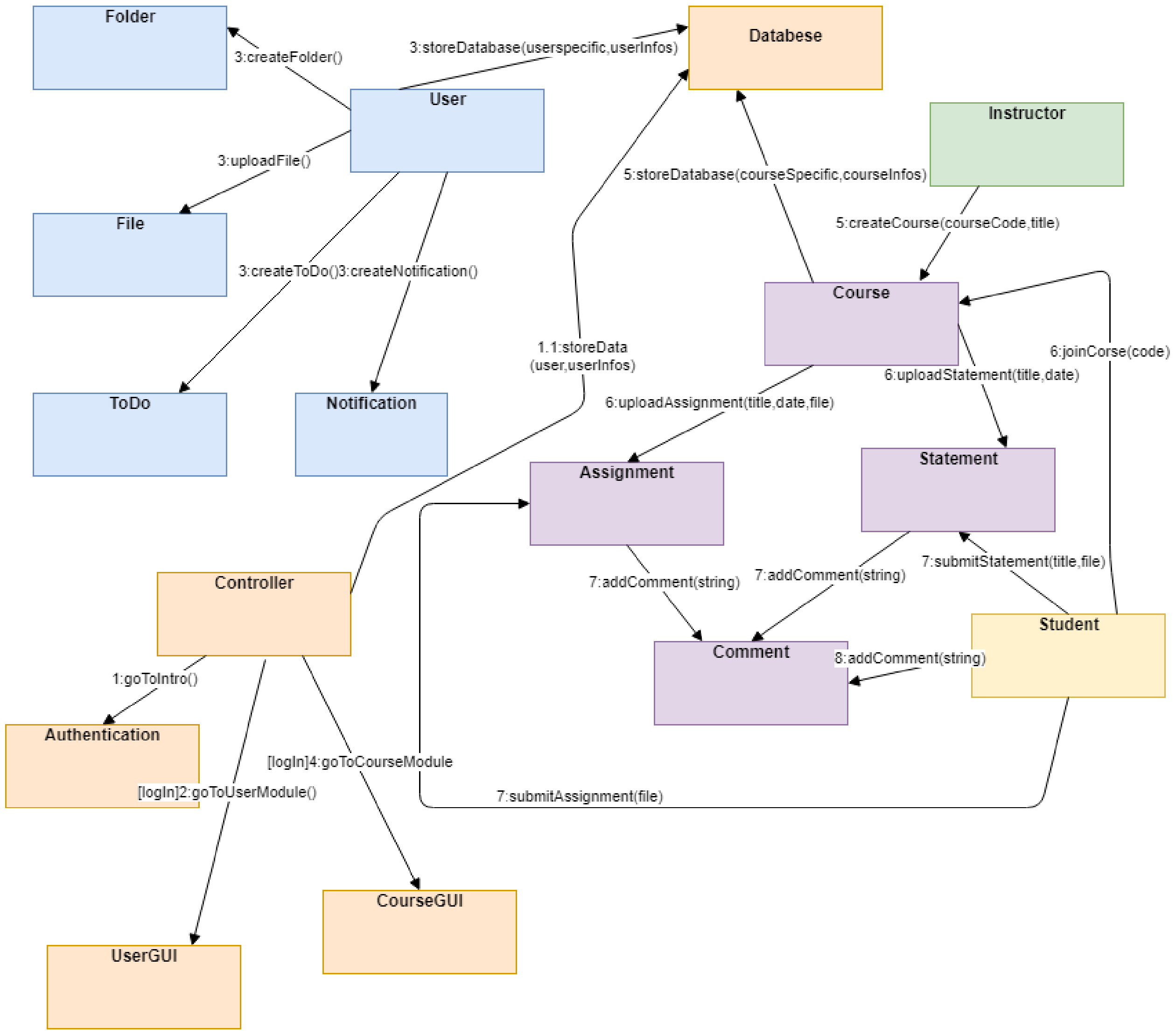


Fig 16:class collaboration detail of “Srenikokkho”

- **Appropriate Interfaces**

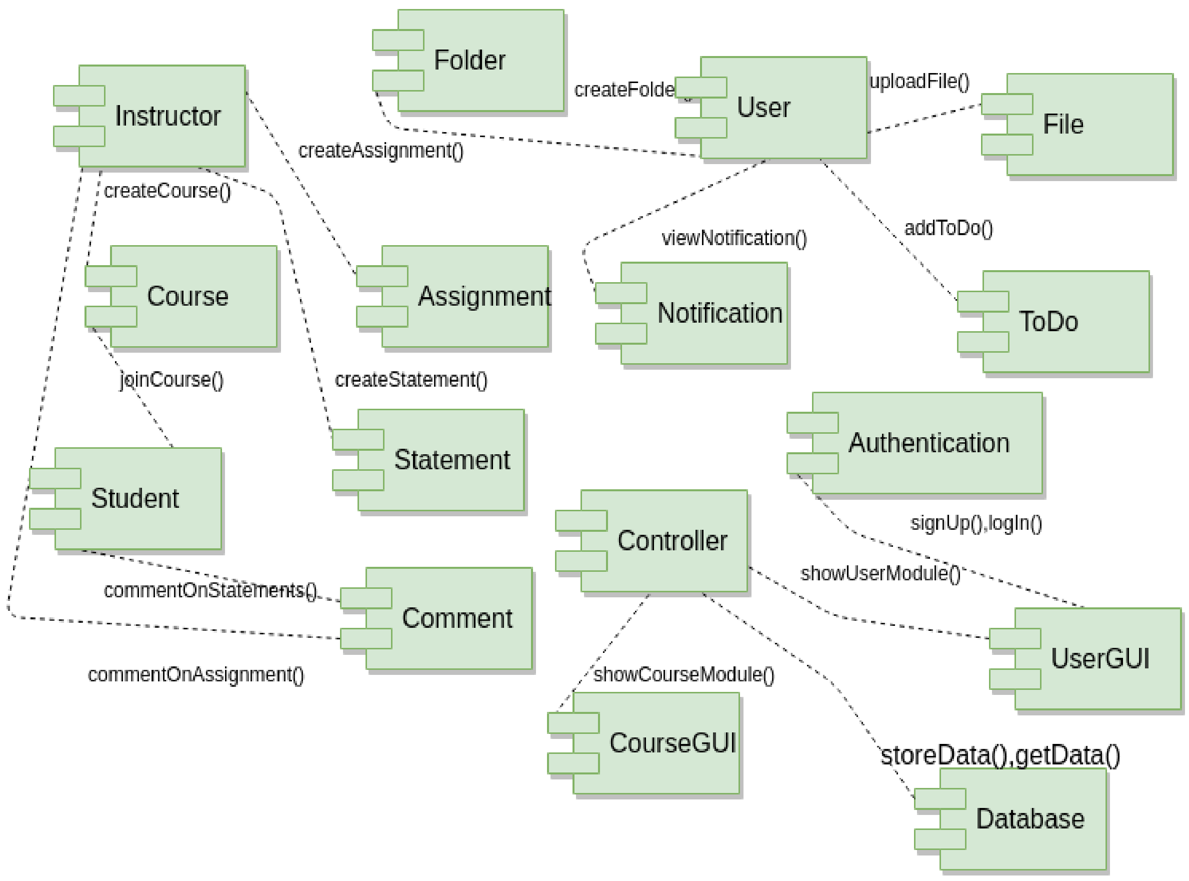


Fig 17: public methods of different classes

- **Elaborate attributes**
    1. Analysis classes will typically only list names of general attributes (ex. paperType).
    2. List all attributes during component design.
    3. UML syntax: name : type-expression = initial-value { property string}

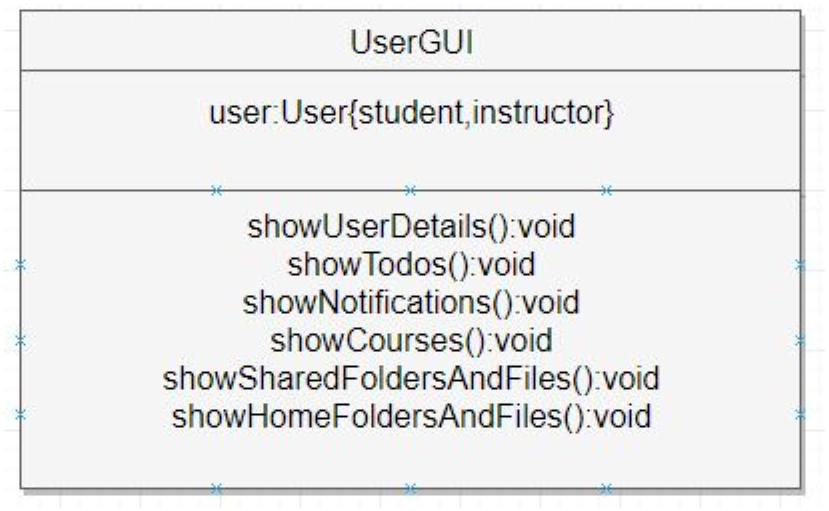

UserGUI
user:User{student,instructor}
showUserDetails():void
showTodos():void
showNotifications():void
showCourses():void
showSharedFoldersAndFiles():void
showHomeFoldersAndFiles():void

Fig:18

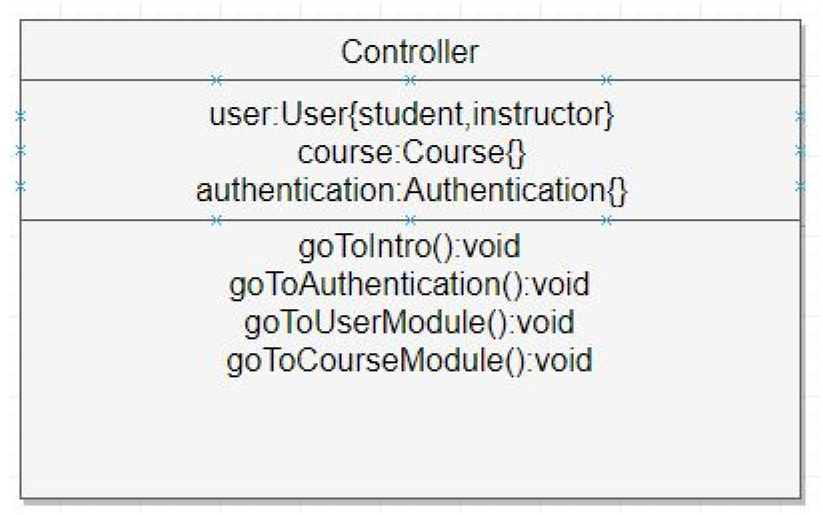

Controller
user:User{student,instructor}
course:Course{}
authentication:Authentication{}
goToIntro():void
goToAuthentication():void
goToUserModule():void
goToCourseModule():void

Fig:19

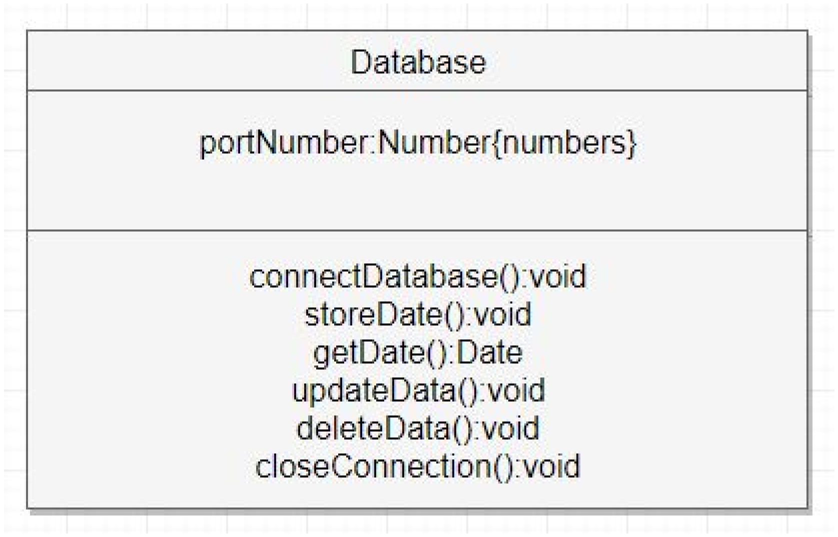

Database
portNumber:Number{numbers}
connectDatabase():void
storeDate():void
getDate():Date
updateData():void
deleteData():void
closeConnection():void

Fig:20

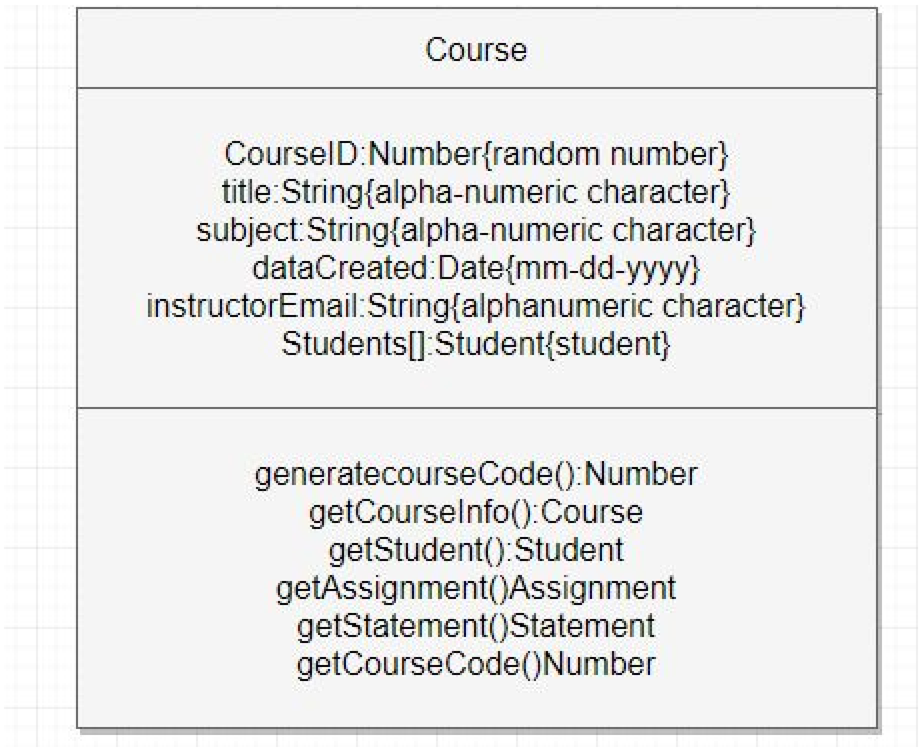

Course
CourseID:Number{random number}
title:String{alpha-numeric character}
subject:String{alpha-numeric character}
dataCreated:Date{mm-dd-yyyy}
instructorEmail:String{alphanumeric character}
Students[]:Student{student}
generatecourseCode():Number
getCourseInfo():Course
getStudent():Student
getAssignment()Assignment
getStatement()Statement
getCourseCode()Number

fig:21

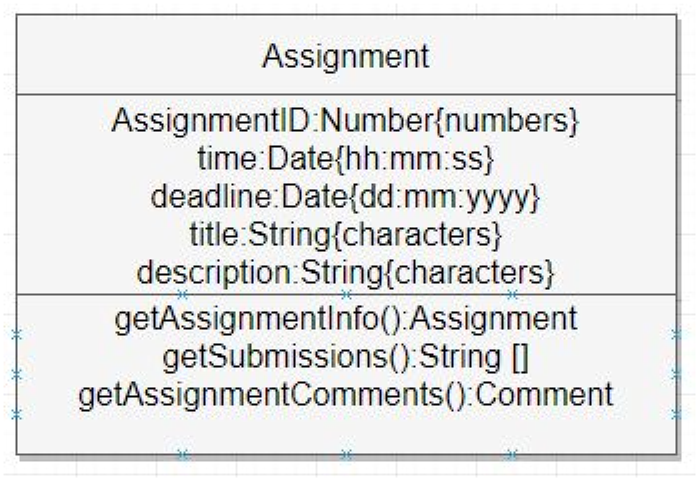

Assignment
AssignmentID:Number{numbers}
time:Date{hh:mm:ss}
deadline:Date{dd:mm:yyyy}
title:String{characters}
description:String{characters}
getAssignmentInfo():Assignment
getSubmissions():String []
getAssignmentComments():Comment

Fig:22

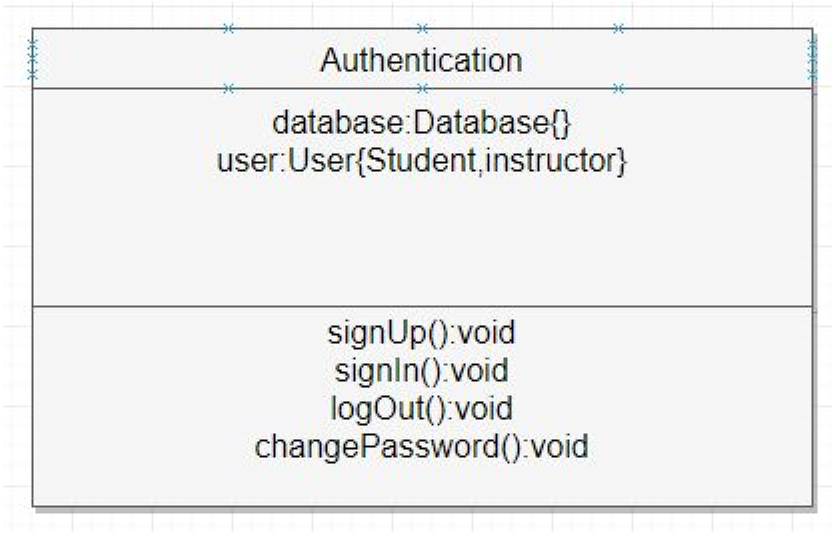

Authentication
database:Database{}
user:User{Student,instructor}
signUp():void
signIn():void
logOut():void
changePassword():void

Fig: 23

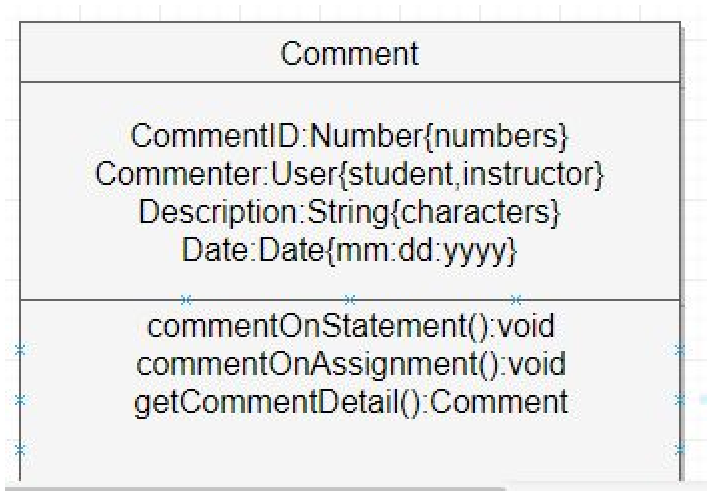

Comment
CommentID:Number{numbers}
Commenter:User{student,instructor}
Description:String{characters}
Date:Date{mm:dd:yyyy}
commentOnStatement():void
commentOnAssignment():void
getCommentDetail():Comment

Fig:24

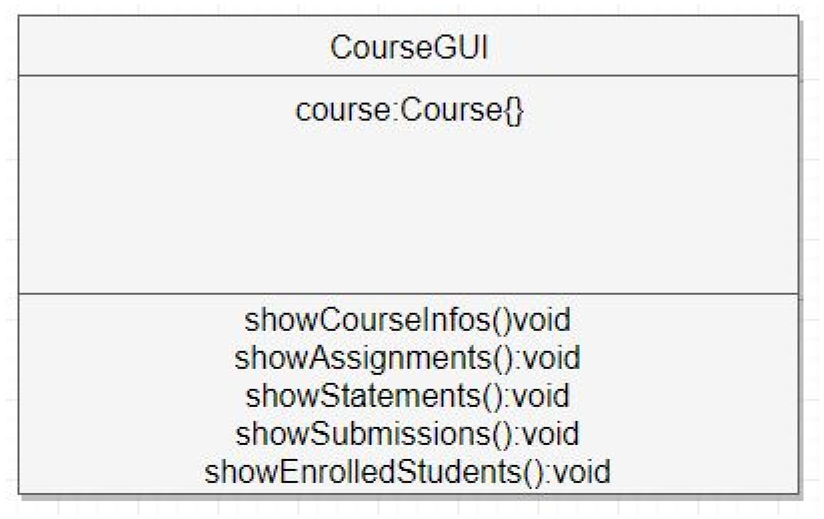

CourseGUI
course:Course{}
showCourseInfos()void
showAssignments():void
showStatements():void
showSubmissions():void
showEnrolledStudents():void

Fig:25

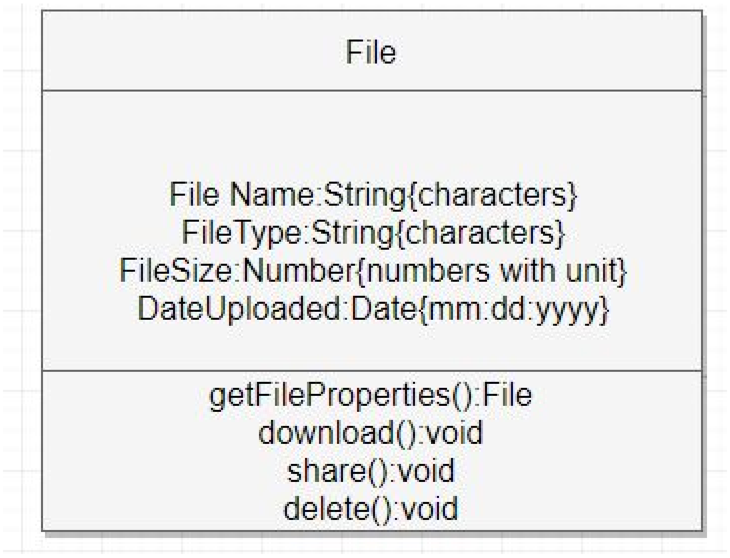

File
File Name:String{characters}
FileType:String{characters}
FileSize:Number{numbers with unit}
DateUploaded:Date{mm:dd:yyyy}
getFileProperties():File
download():void
share():void
delete():void

Fig:26

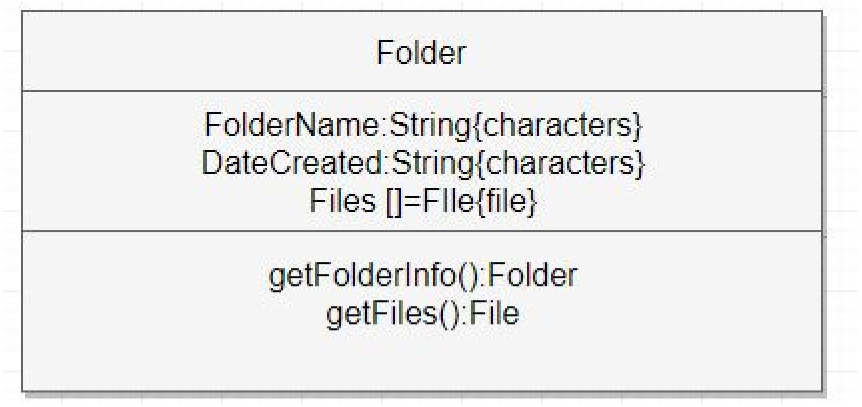

Folder
FolderName:String{characters}
DateCreated:String{characters}
Files []=FIle{file}
getFolderInfo():Folder
getFiles():File

Fig:27

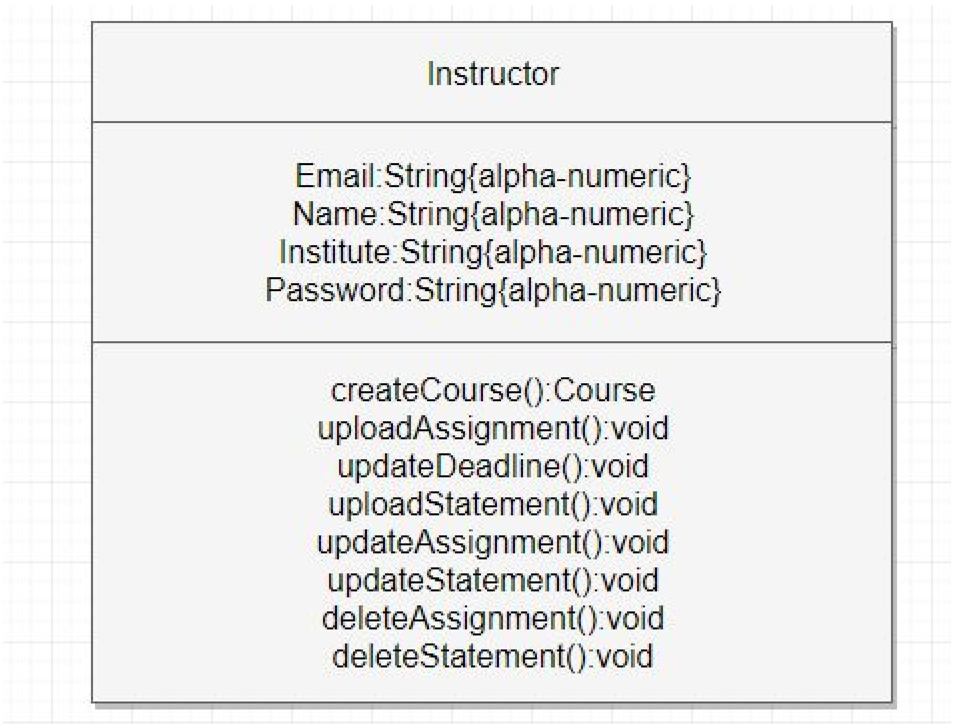

Instructor
Email:String{alpha-numeric}
Name:String{alpha-numeric}
Institute:String{alpha-numeric}
Password:String{alpha-numeric}
createCourse():Course
uploadAssignment():void
updateDeadline():void
uploadStatement():void
updateAssignment():void
updateStatement():void
deleteAssignment():void
deleteStatement():void

Fig:28

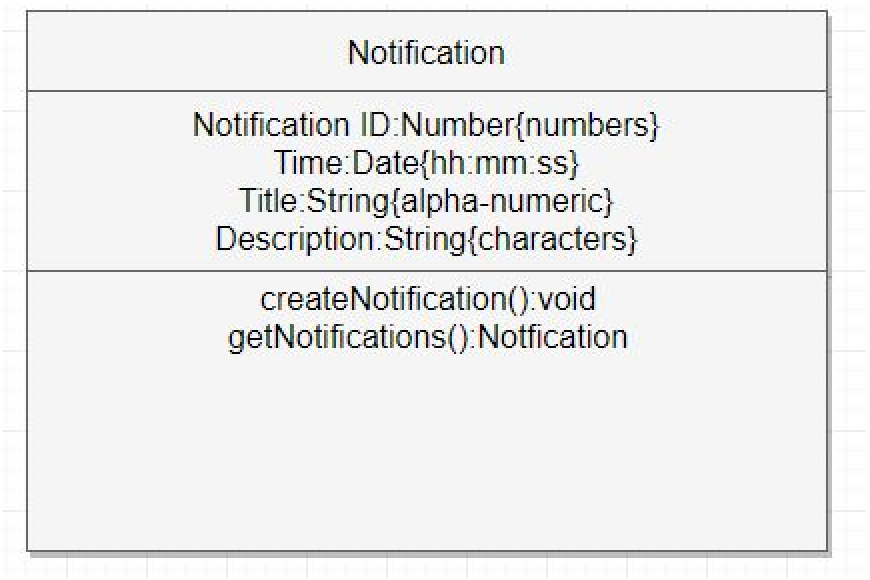

Notification
Notification ID:Number{numbers}
Time:Date{hh:mm:ss}
Title:String{alpha-numeric}
Description:String{characters}
createNotification():void
getNotifications():Notfication

Fig:29

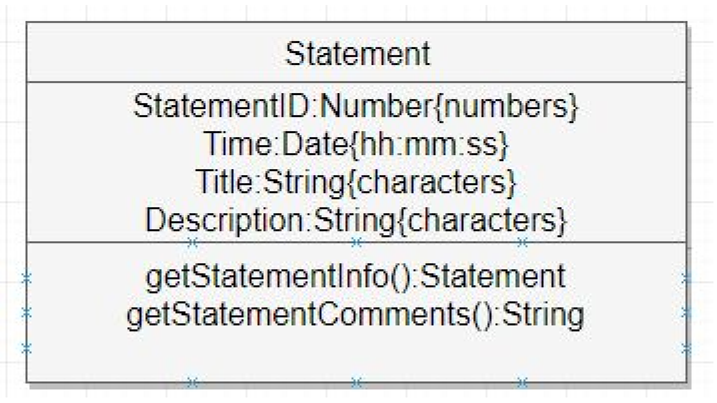

Statement
StatementID:Number{numbers}
Time:Date{hh:mm:ss}
Title:String{characters}
Description:String{characters}
getStatementInfo():Statement
getStatementComments():String

Fig:30

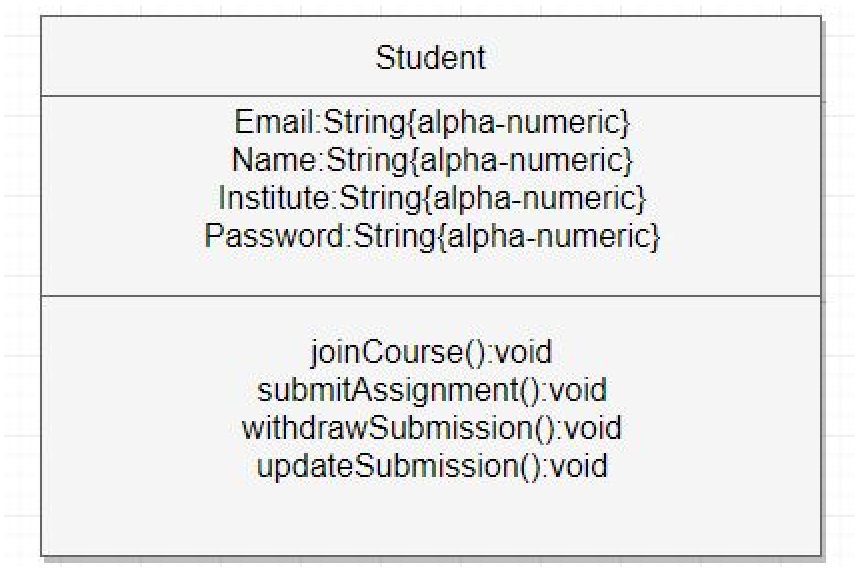

Student
Email:String{alpha-numeric}
Name:String{alpha-numeric}
Institute:String{alpha-numeric}
Password:String{alpha-numeric}
joinCourse():void
submitAssignment():void
withdrawSubmission():void
updateSubmission():void

Fig:31

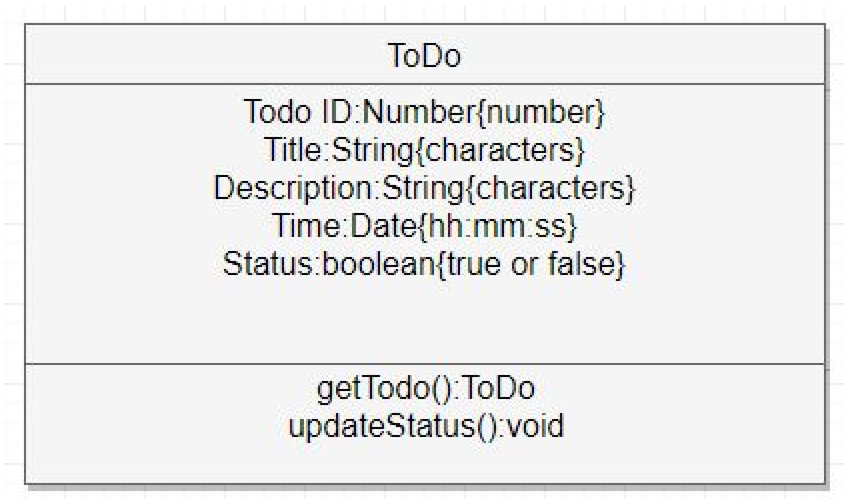

ToDo
Todo ID:Number{number}
Title:String{characters}
Description:String{characters}
Time:Date{hh:mm:ss}
Status:boolean{true or false}
getTodo():ToDo
updateStatus():void

Fig:32

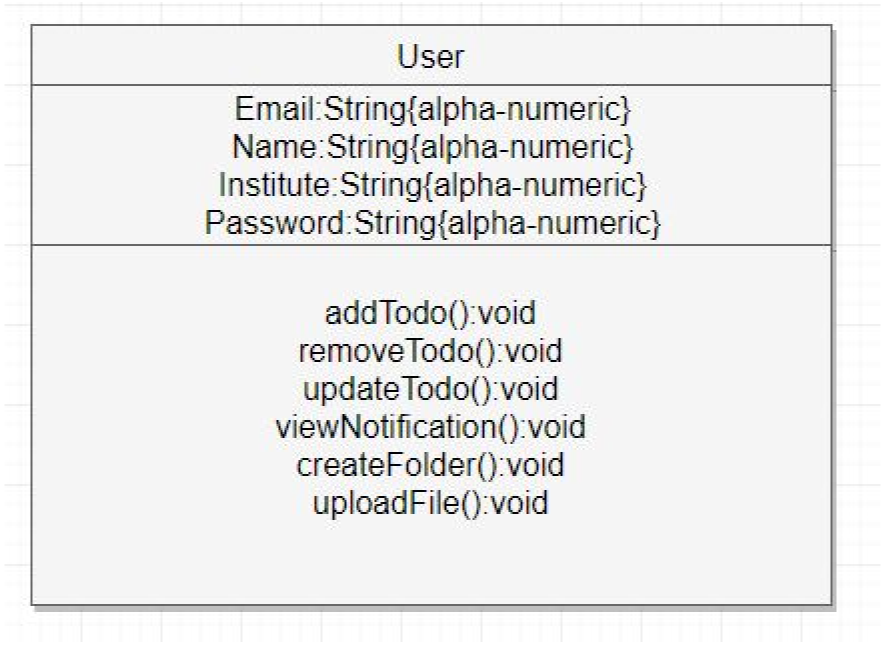

User
Email:String{alpha-numeric}
Name:String{alpha-numeric}
Institute:String{alpha-numeric}
Password:String{alpha-numeric}
addTodo():void
removeTodo():void
updateTodo():void
viewNotification():void
createFolder():void
uploadFile():void

Fig:33

- **Describe processing flow**

Here activity diagram for each methods of the classes are designed.

generateCourseCode()

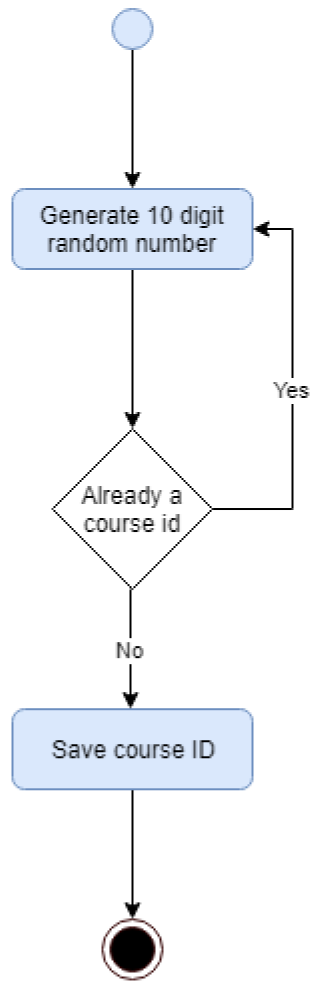


Fig:34

addToDo()

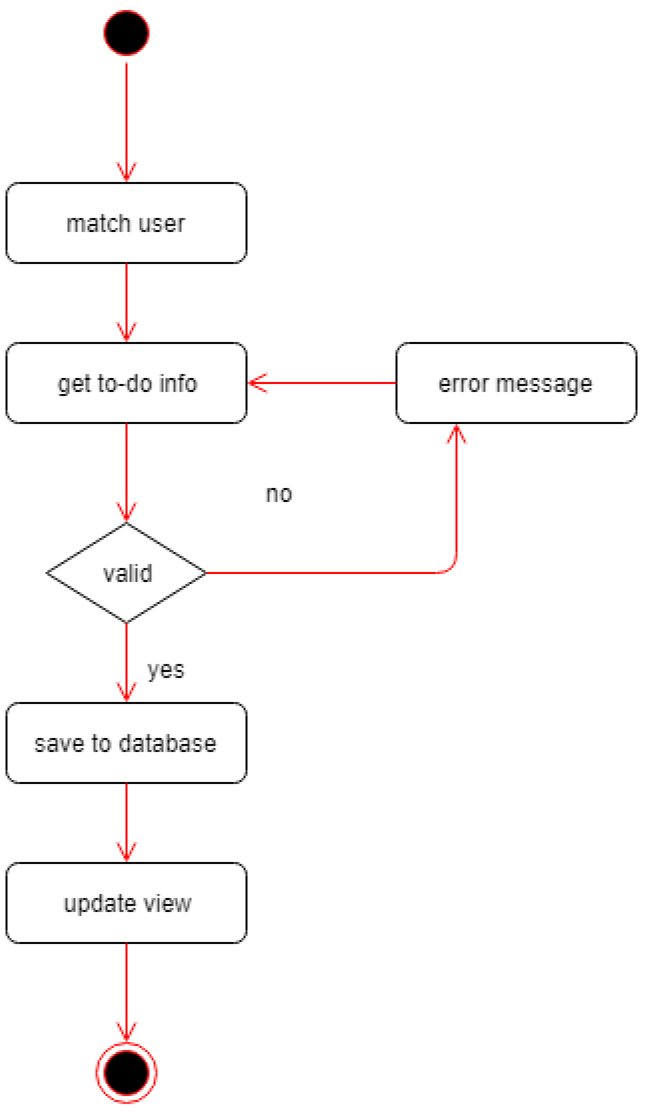


Fig:35

removeTodo()

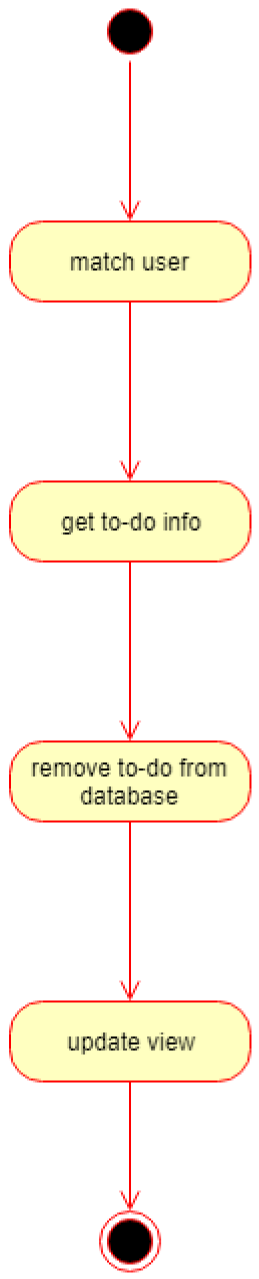


Fig:36

updateTodo()

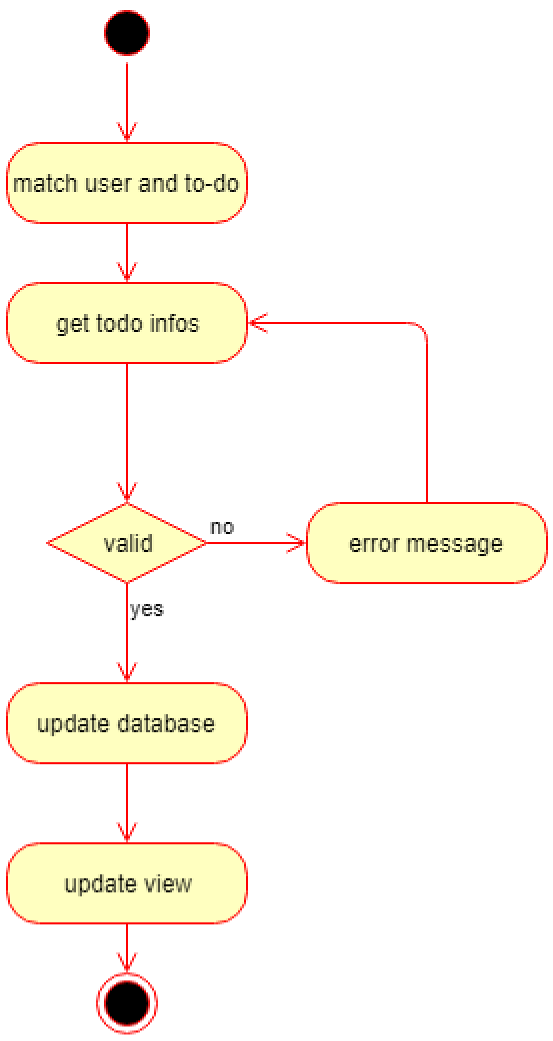


Fig:37

viewNotification()

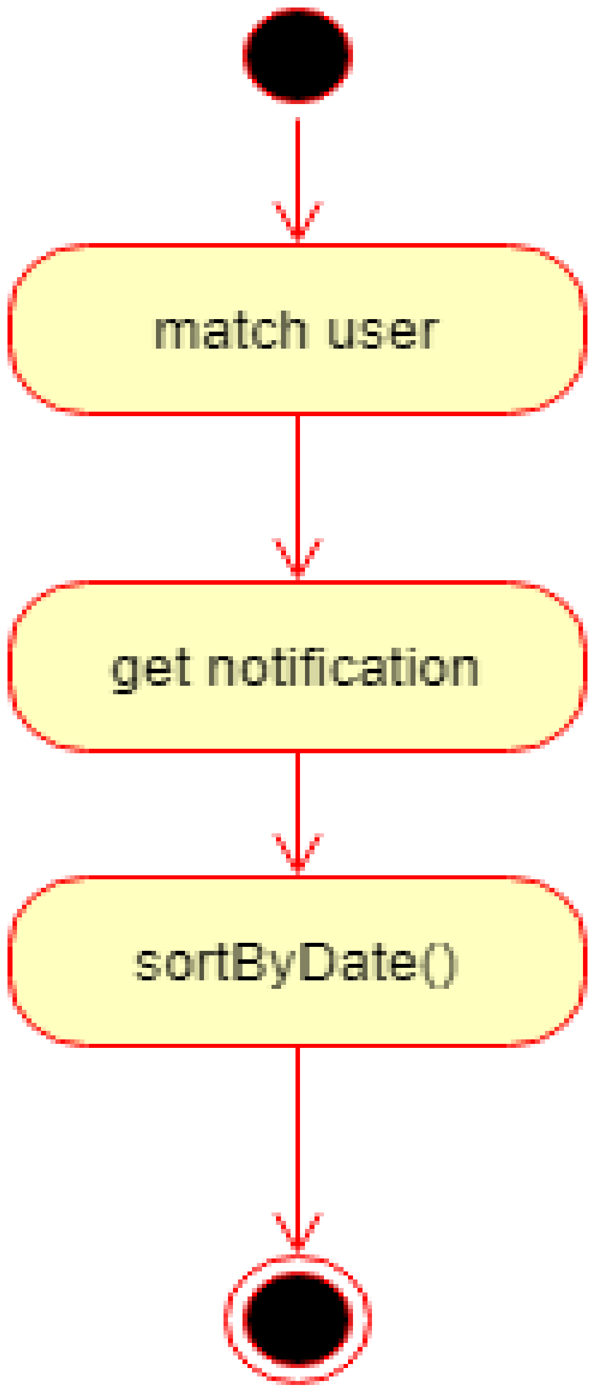


Fig38

uploadFile()

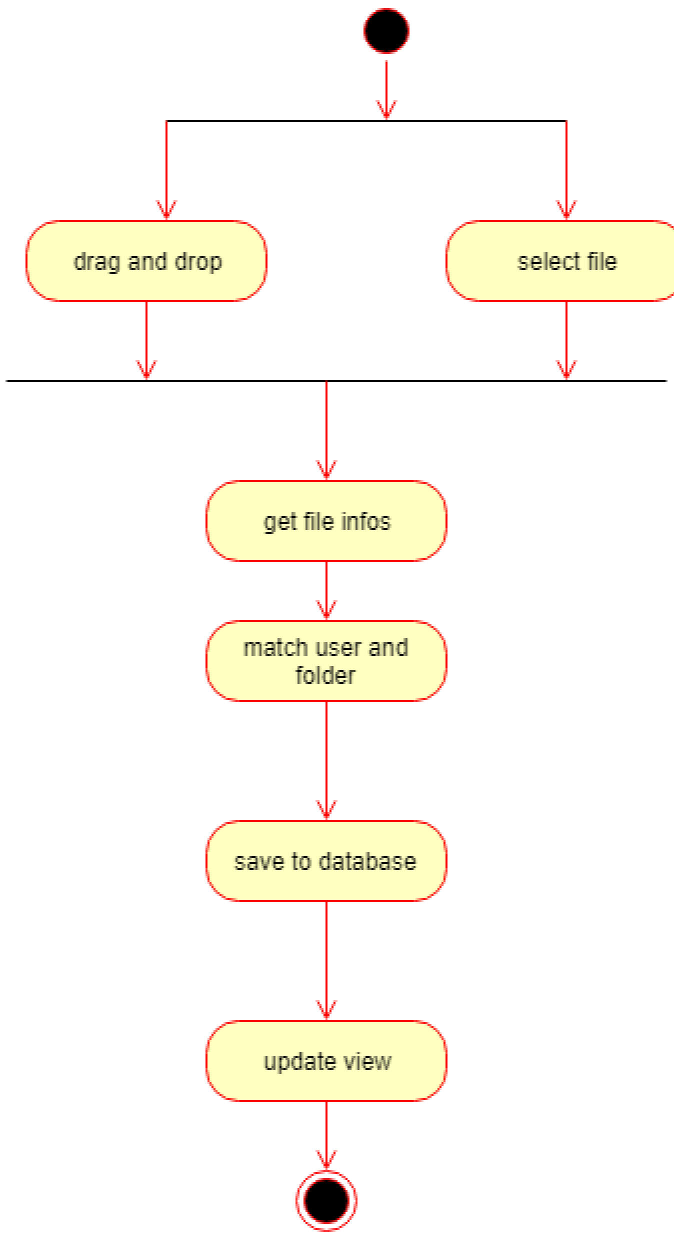


Fig:39

createFolder()

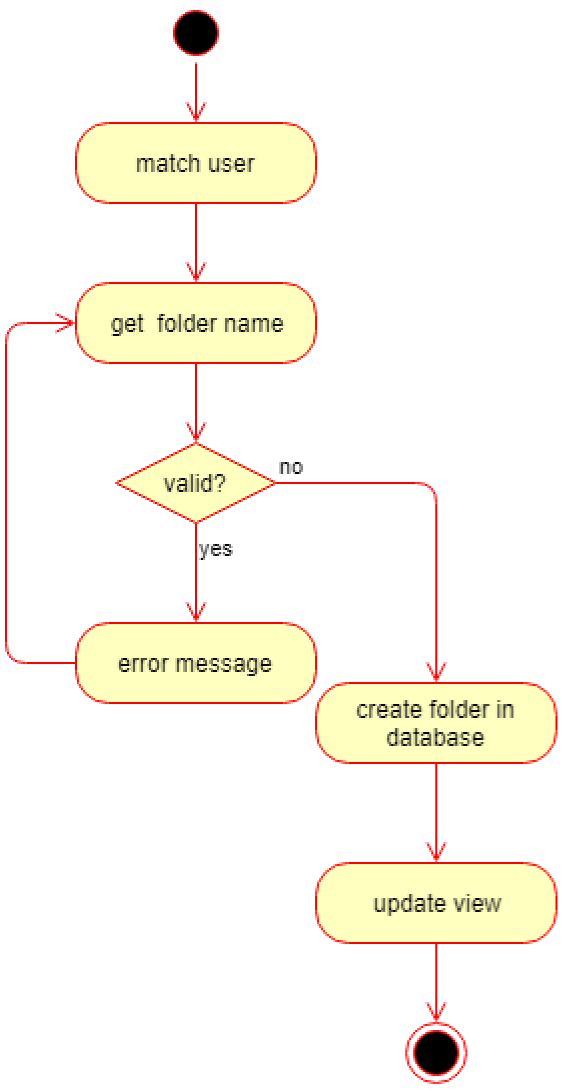


Fig:40

createCourse()

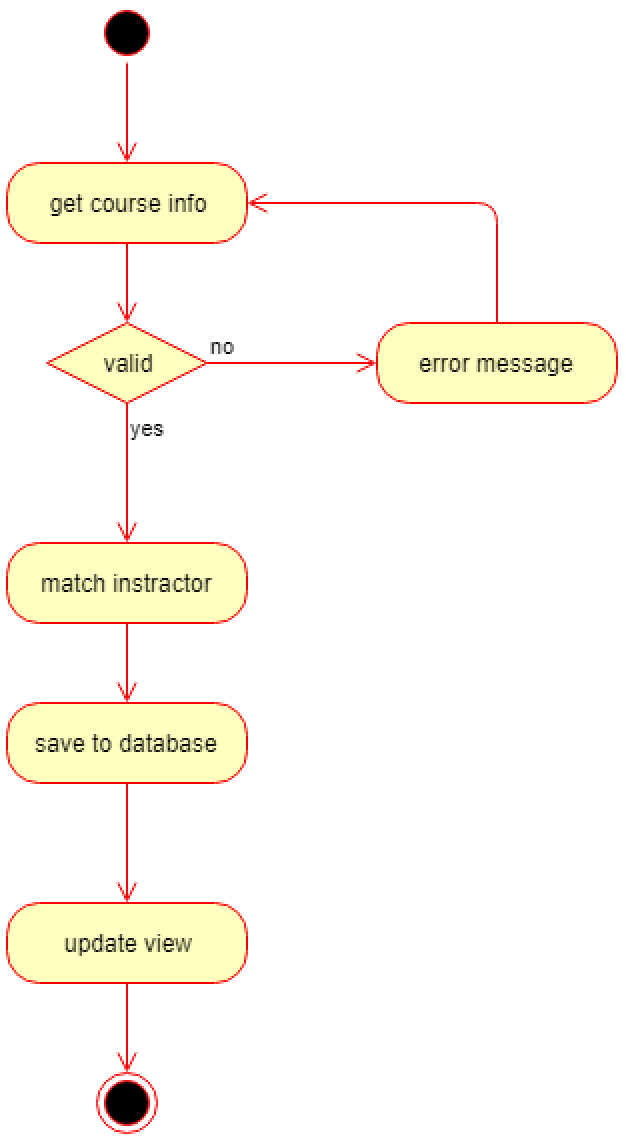


Fig:41

uploadAssignment()

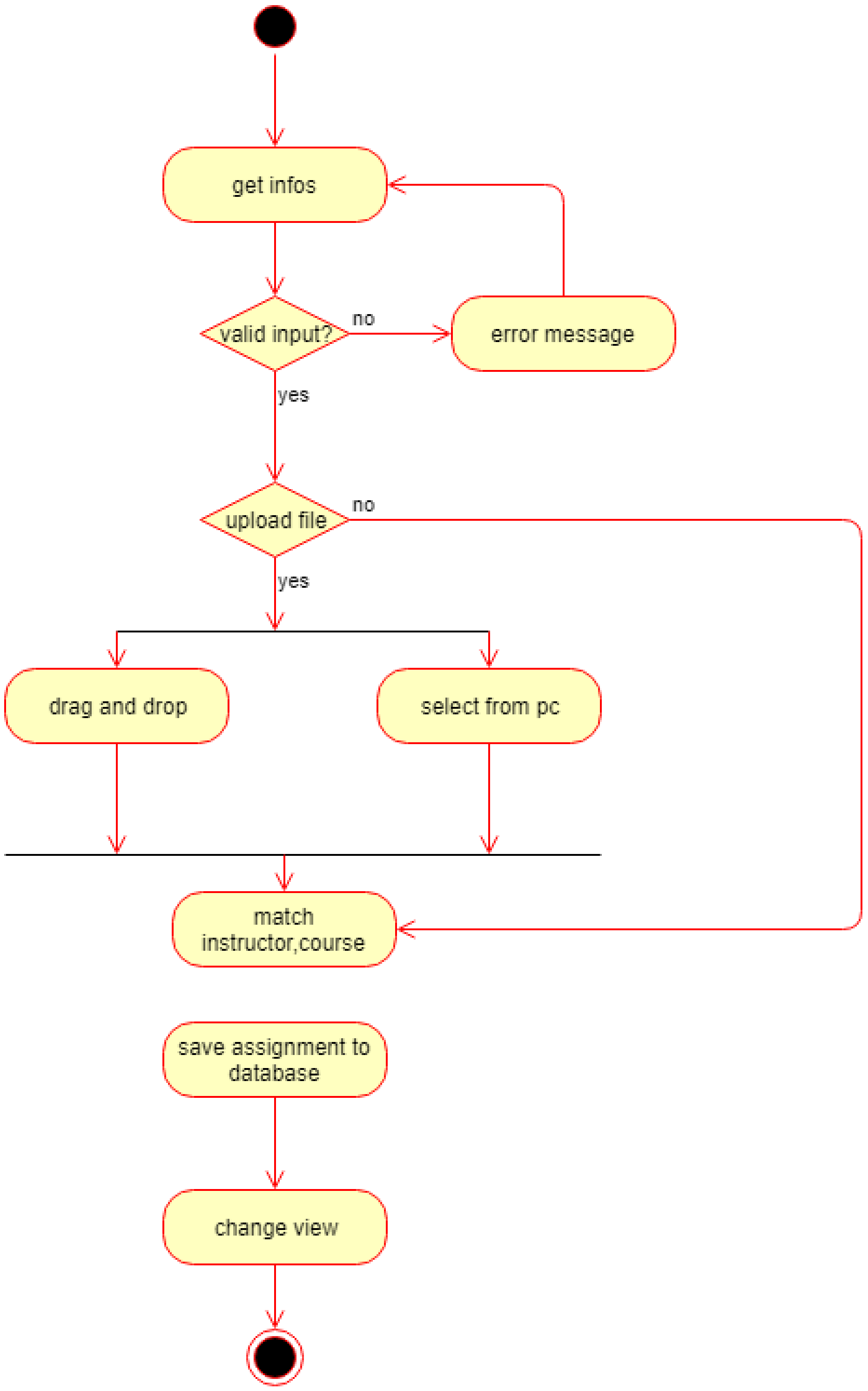


Fig:42

updateDeadline()

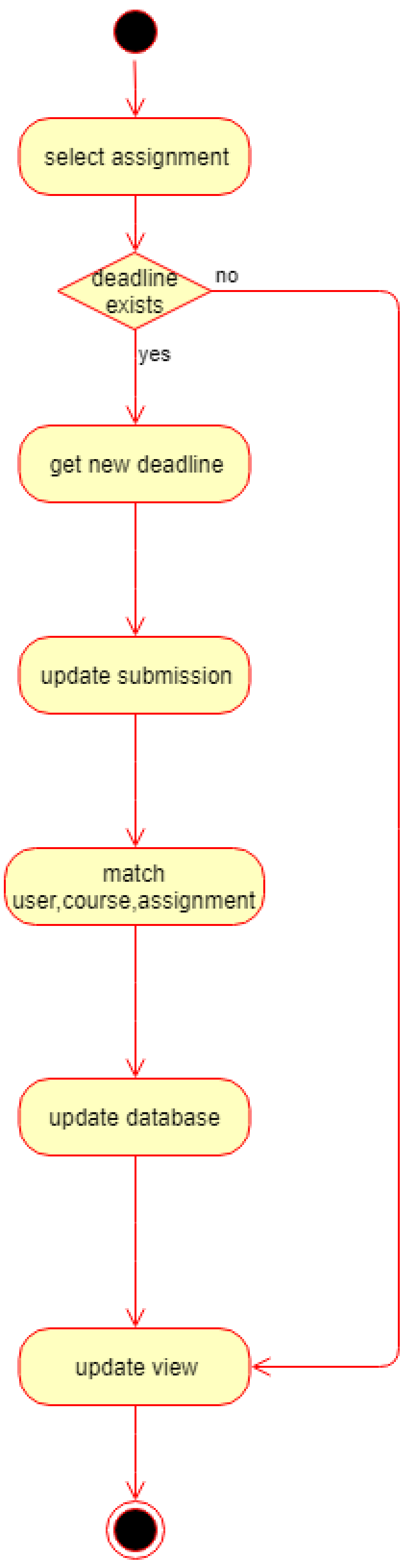


Fig:43

uploadStatement()

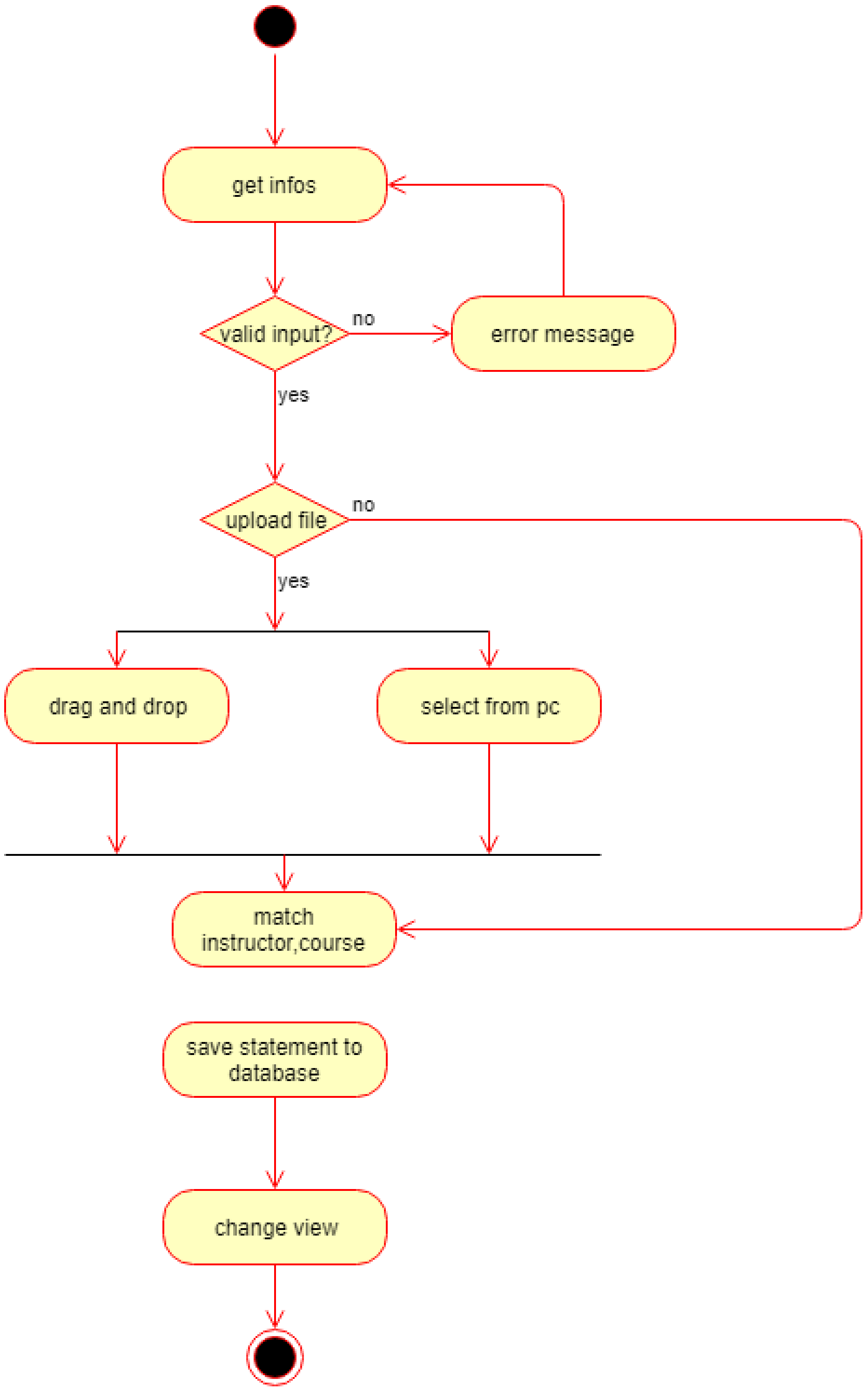


Fig:44

updateAssignment()

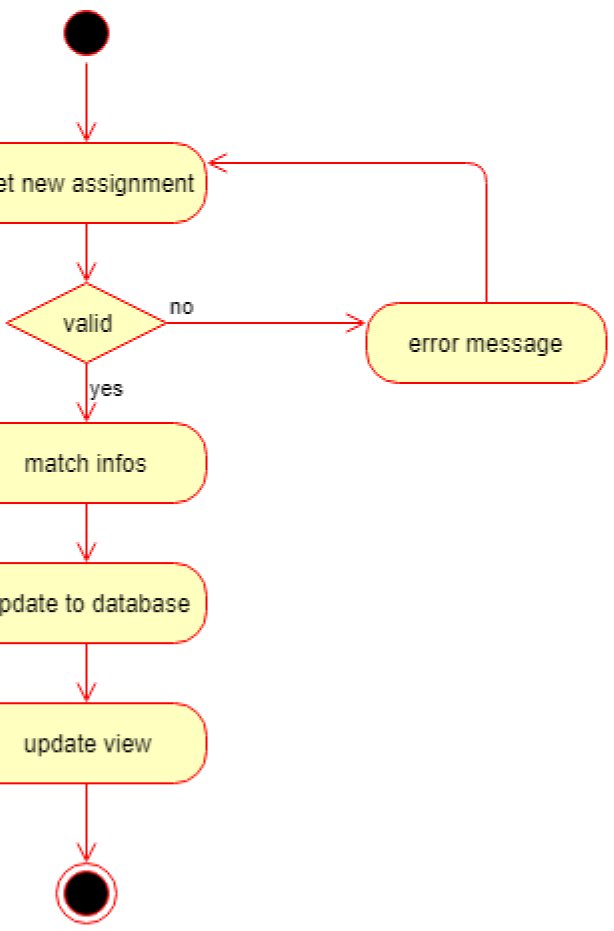


Fig:45

updateStatement()

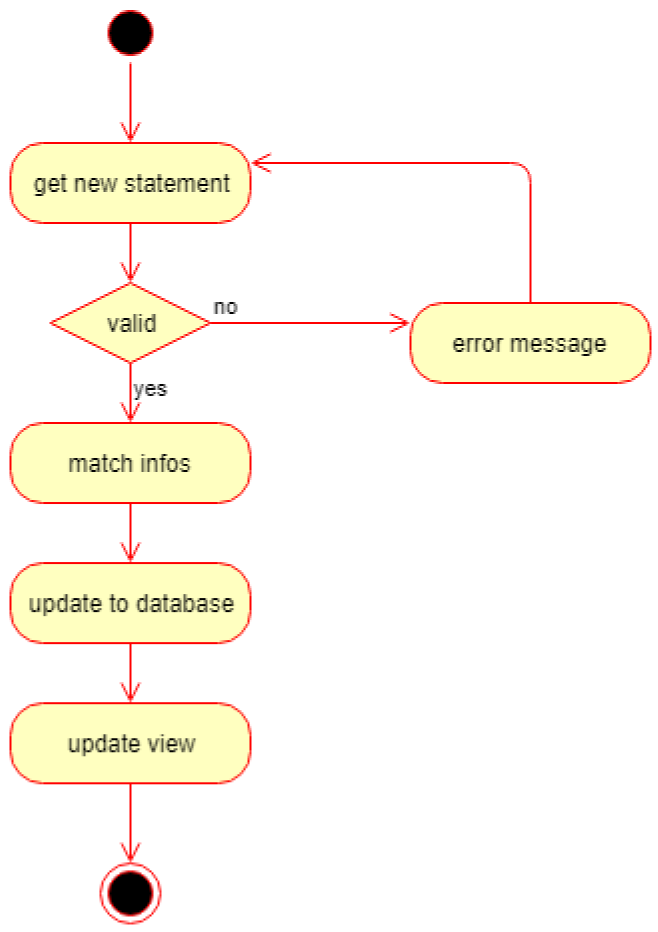


Fig:46

deleteAssignment()

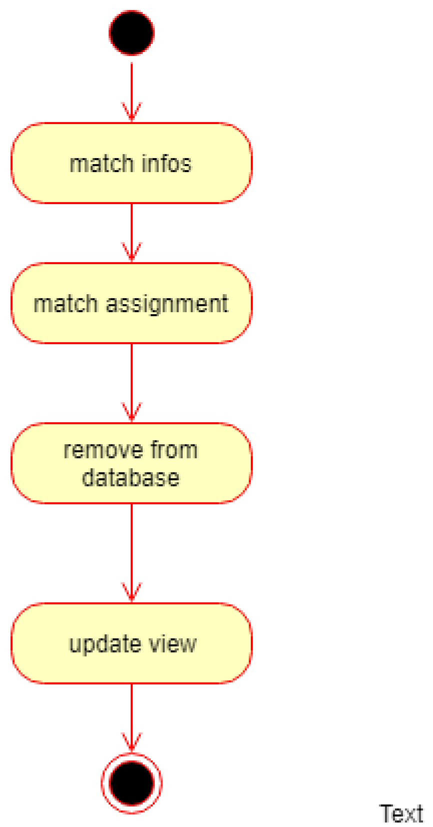


Fig:47

deleteStatement()

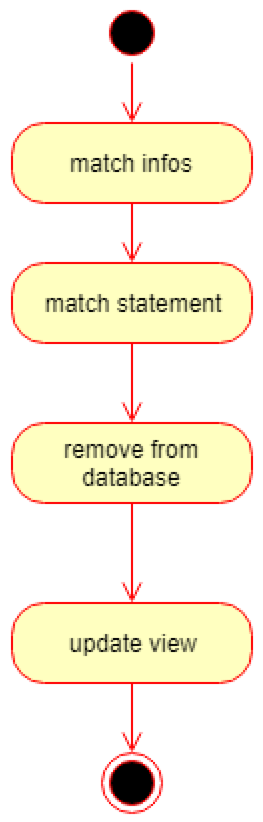


Fig:48

joinCourse()

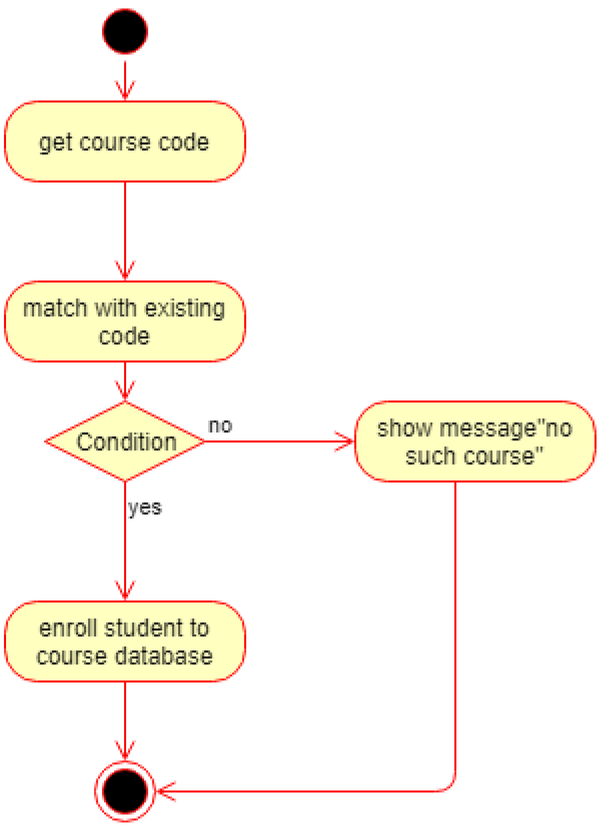

get course code
match with existing code
Condition
no
show message"no such course"
yes
enroll student to course database

Fig:49

submitAssignment()

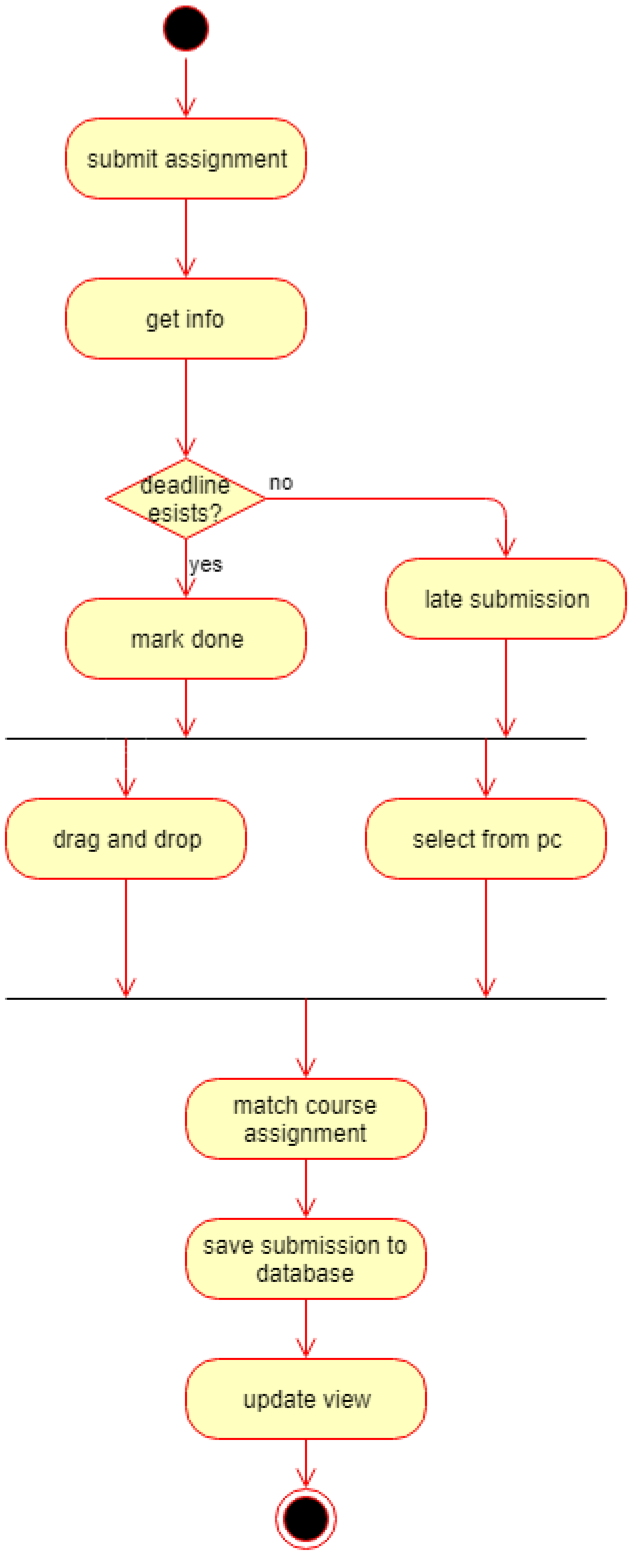


Fig:50

withdrawSubmission()

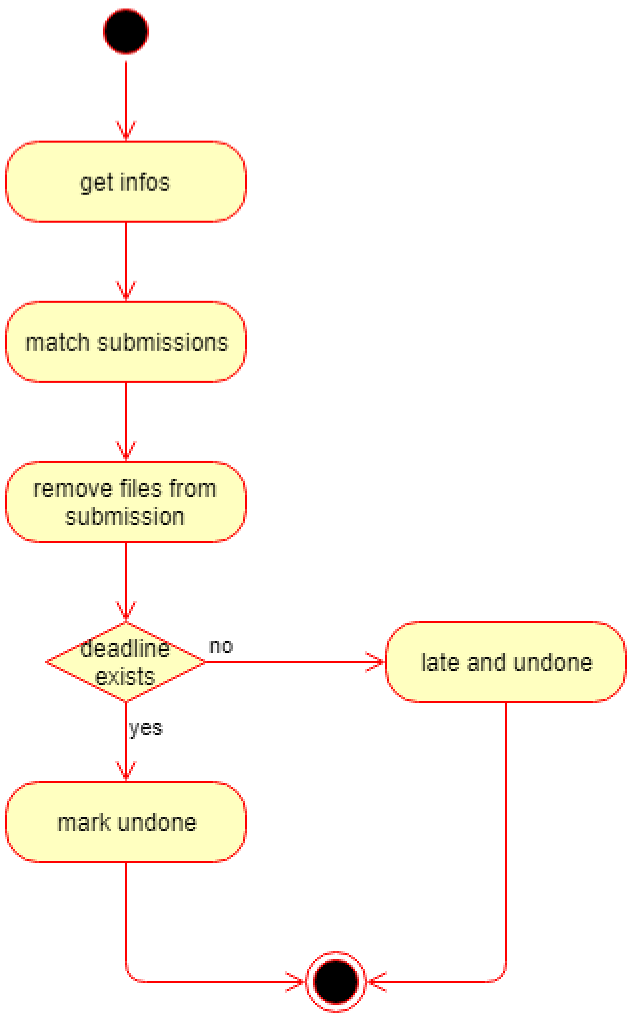


Fig:51

updateSubmission()

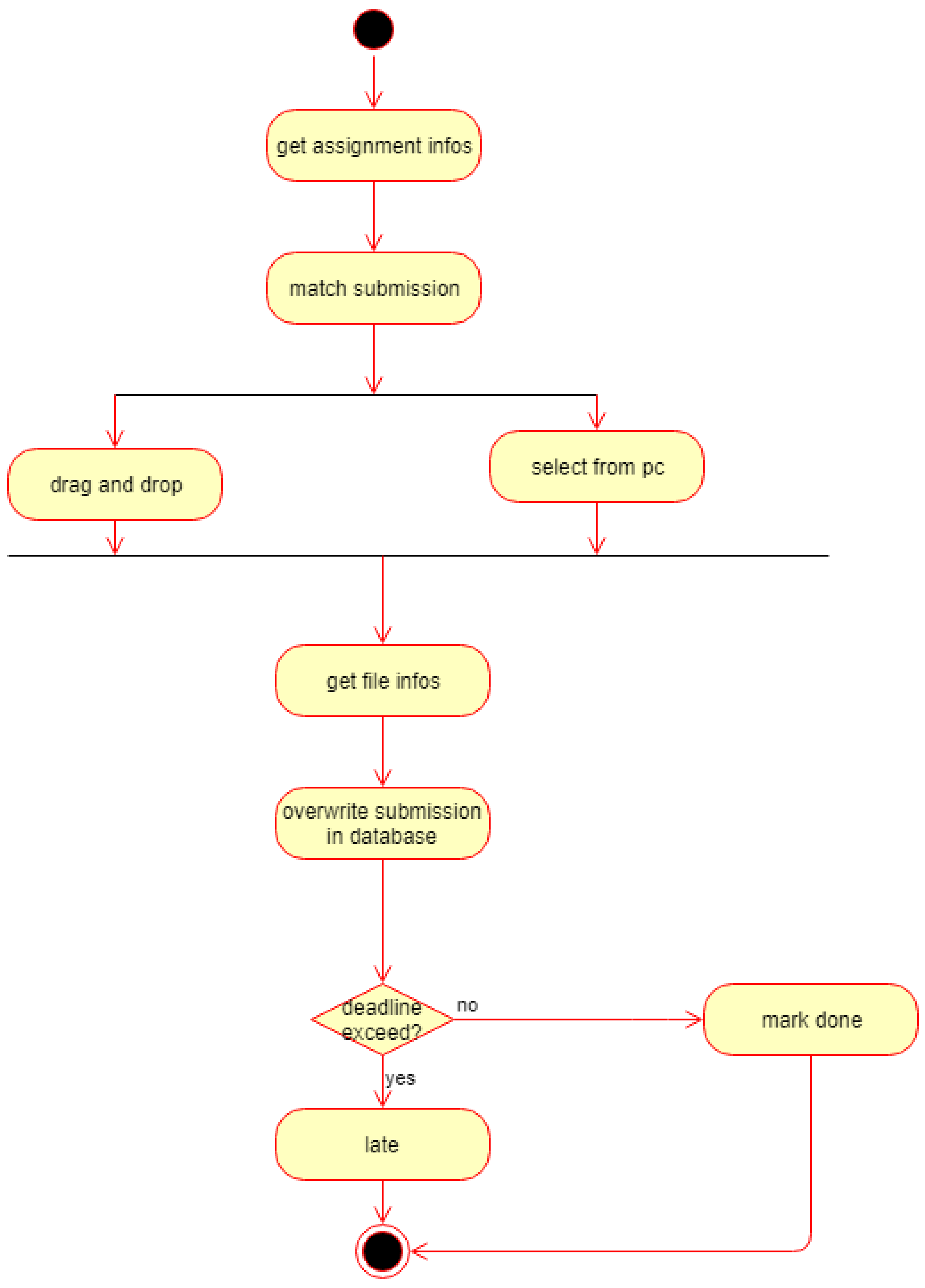


Fig:52

createNotification()

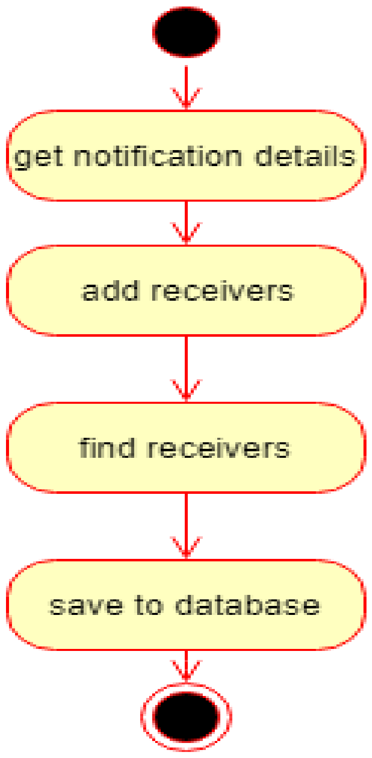


Fig:53

updateStatus()

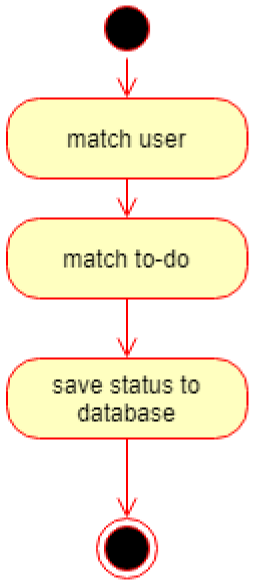


Fig:54

download()

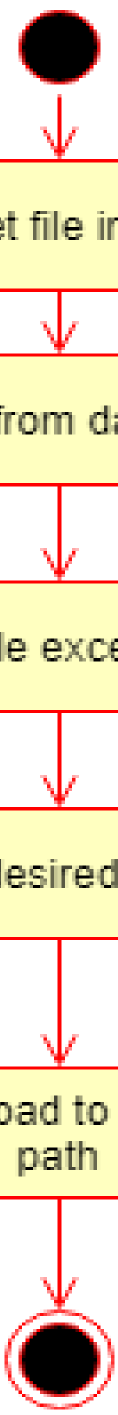
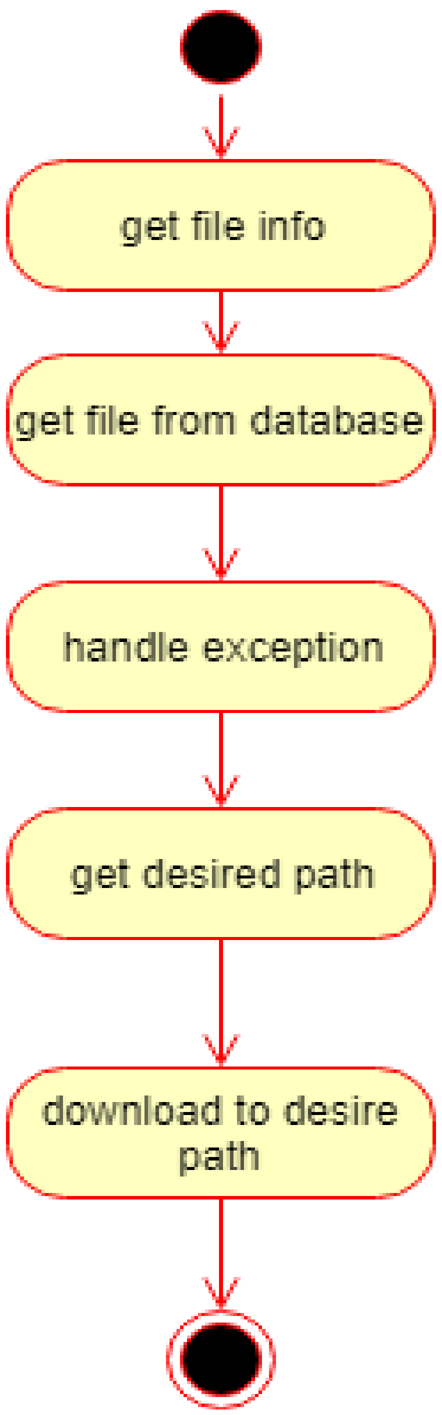



Fig:55

share()

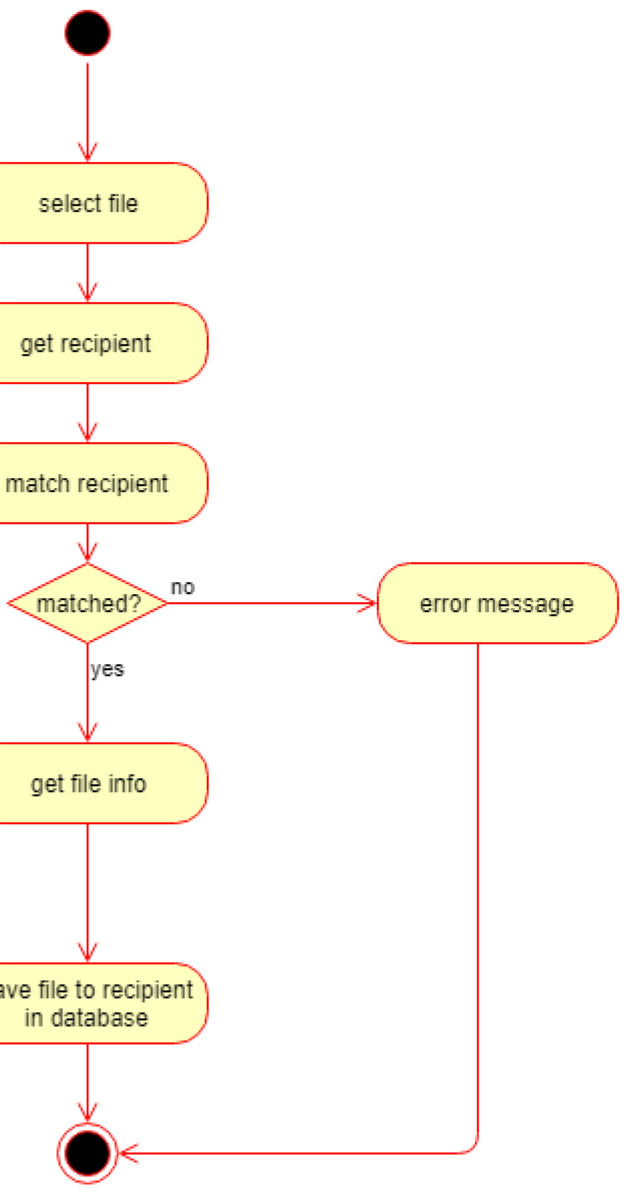

select file
get recipient
match recipient
matched?
no
error message
yes
get file info
save file to recipient in database

Fig:56

delete()

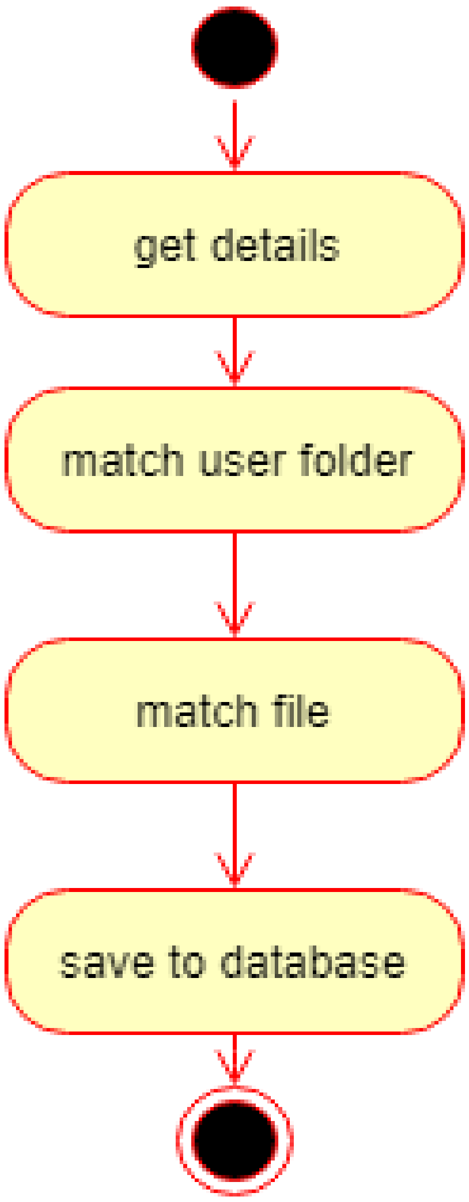


Fig:57

commentOnAssignment()

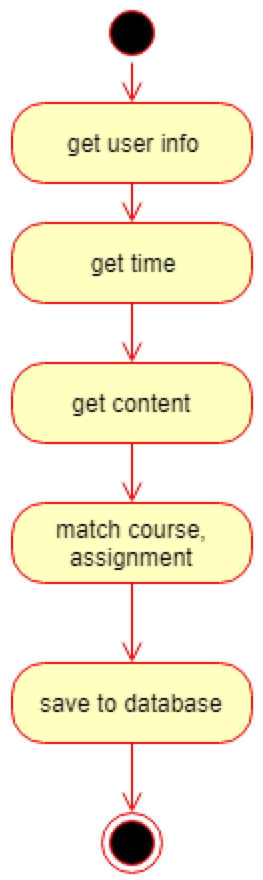


Fig:58

commentOnStatement()

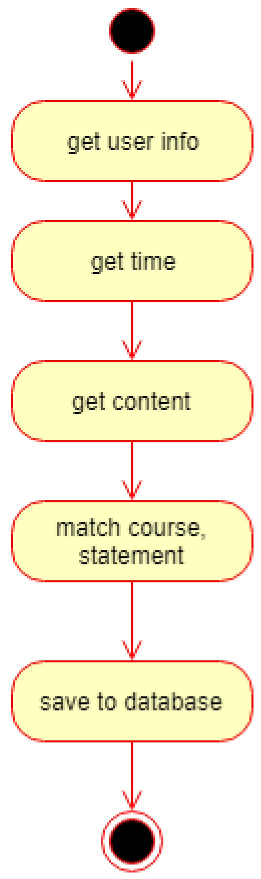


Fig:59

logOut()

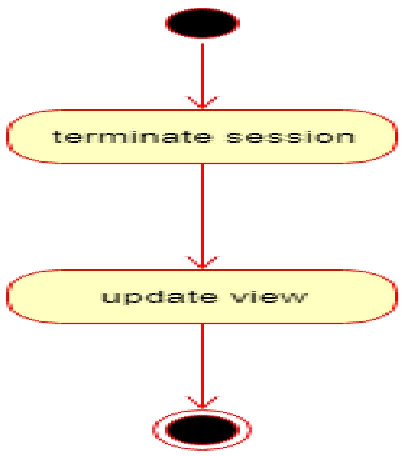


Fig:60

signUp()

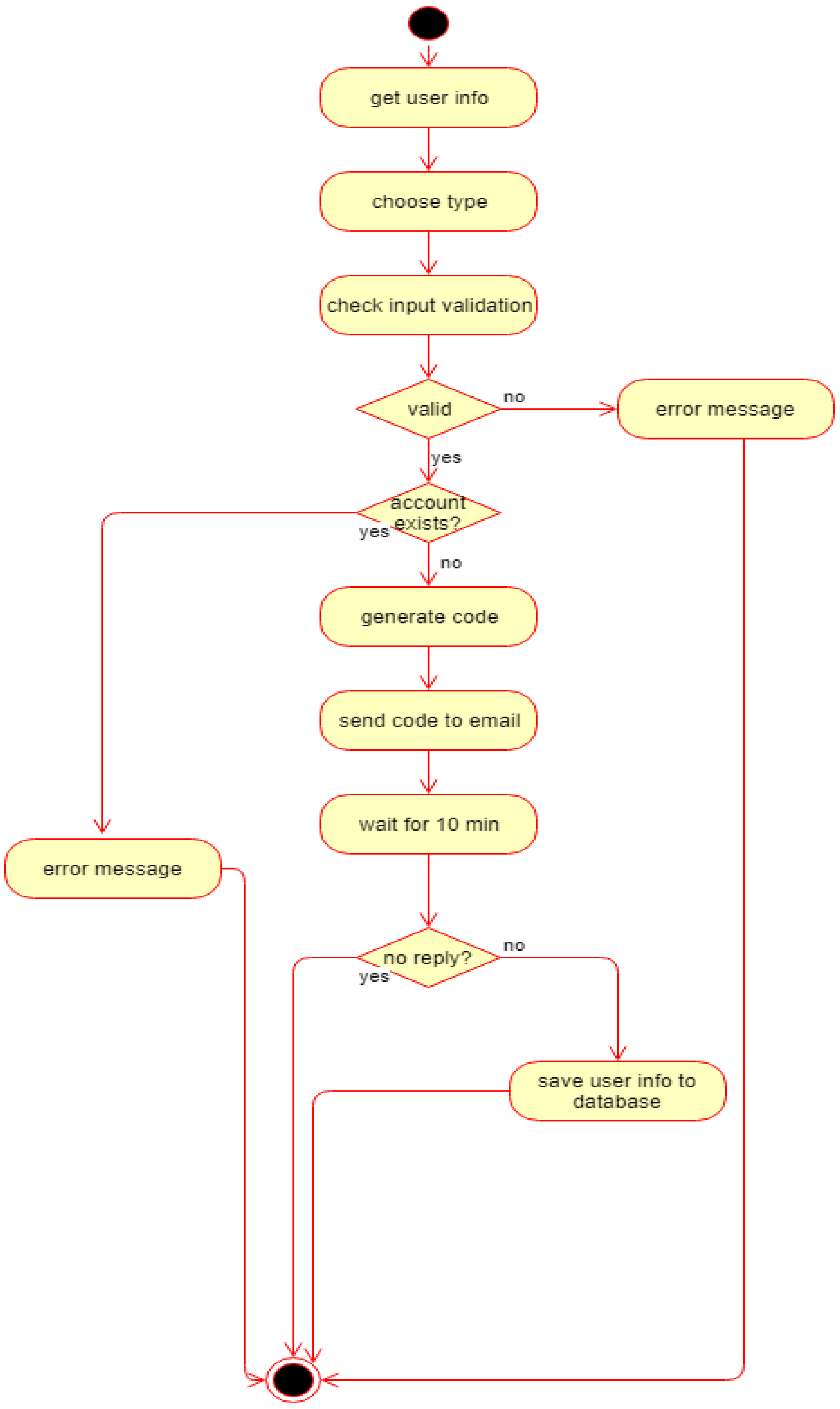


Fig:61

signIn()

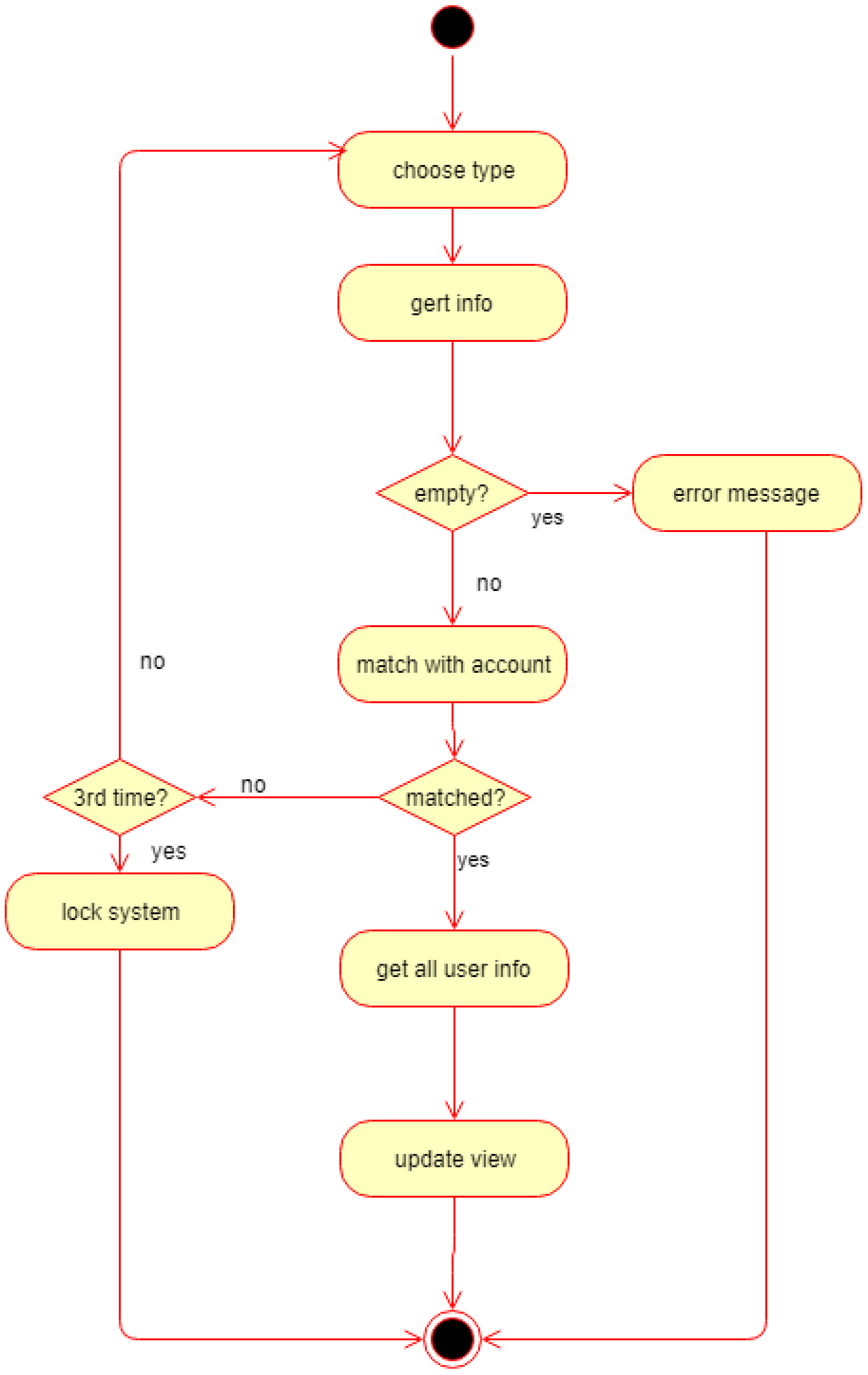


Fig:62

changePassword()

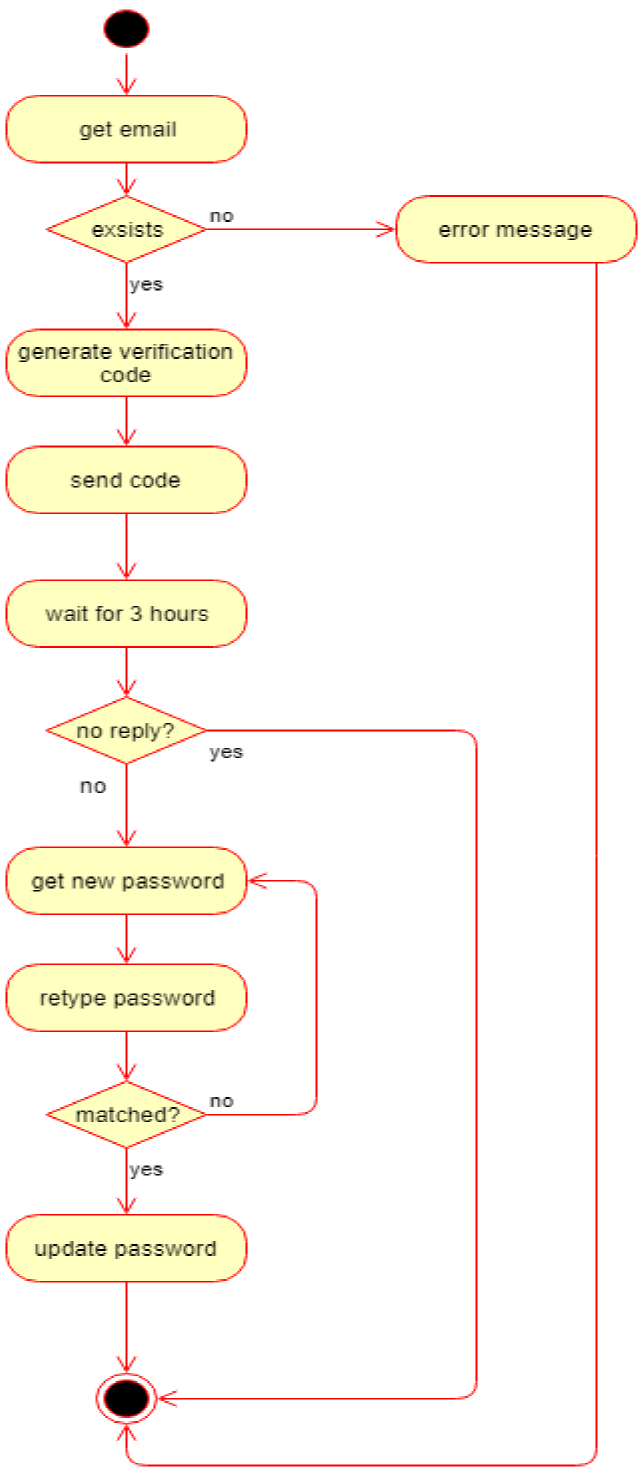


Fig:63

4.persistent database

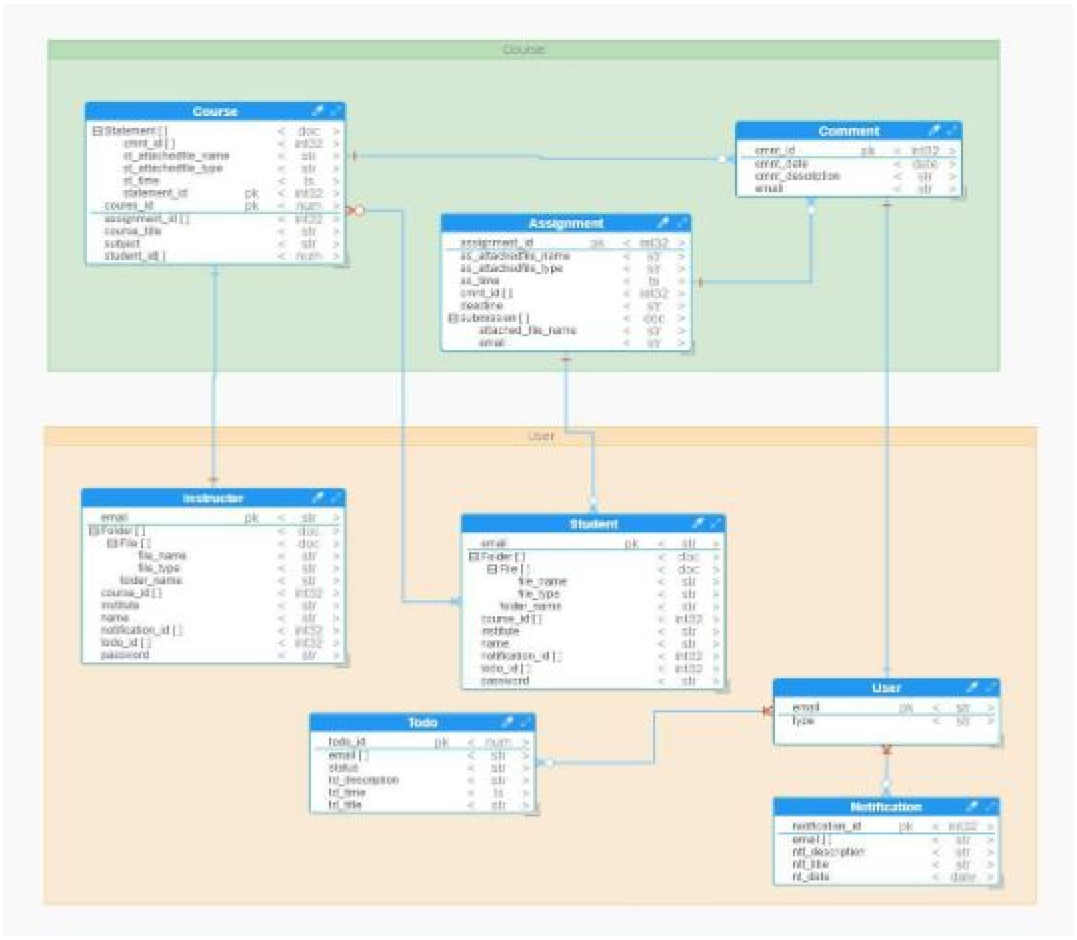


fig:64

5. Elaborate Behaviour

- It is sometimes necessary to model the behavior of a design class.
- Transitions from state to state have the form: Event-name (parameter-list) [guard-condition] / action expression

State transition diagram:

Controller

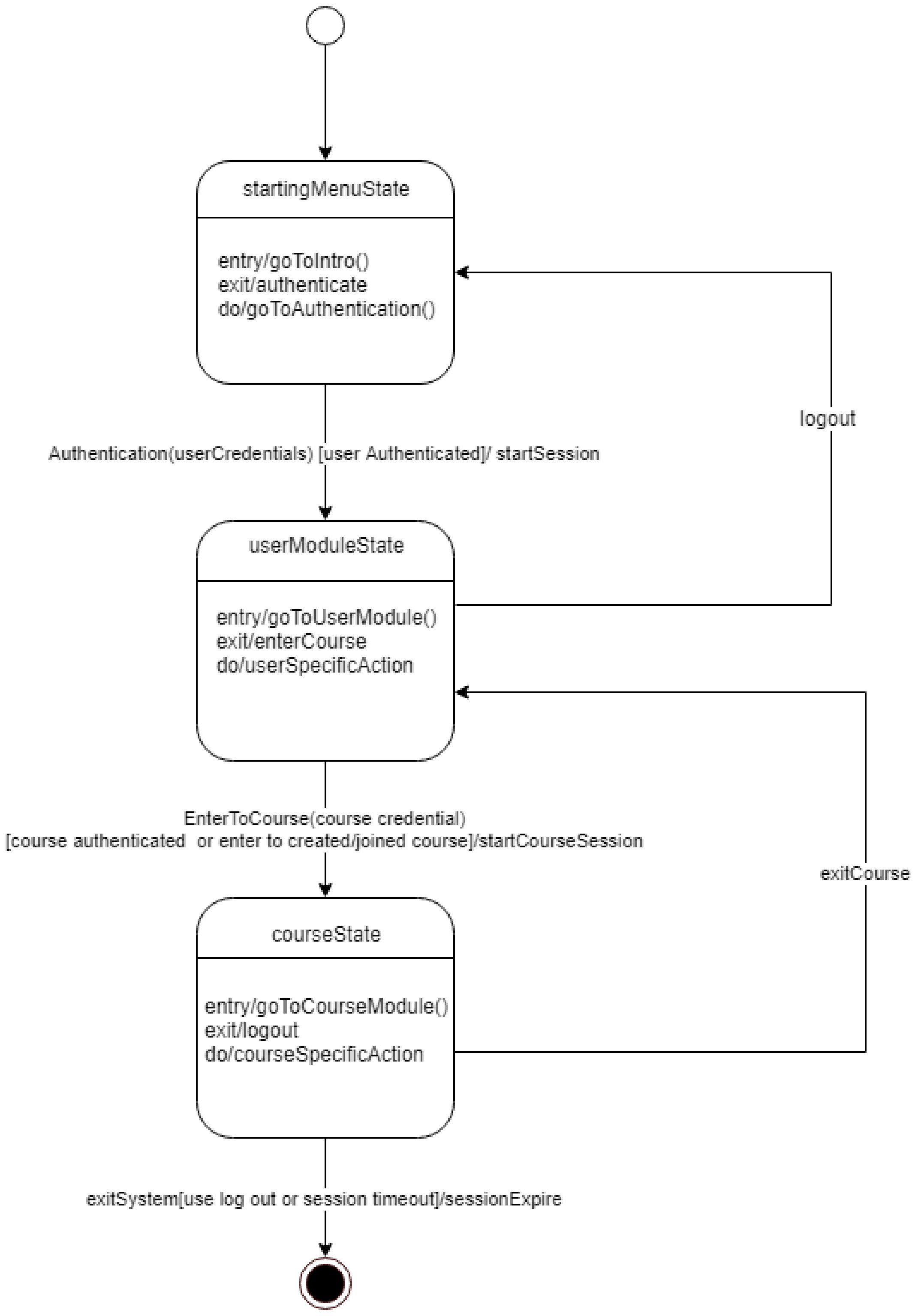


Fig:65

Course

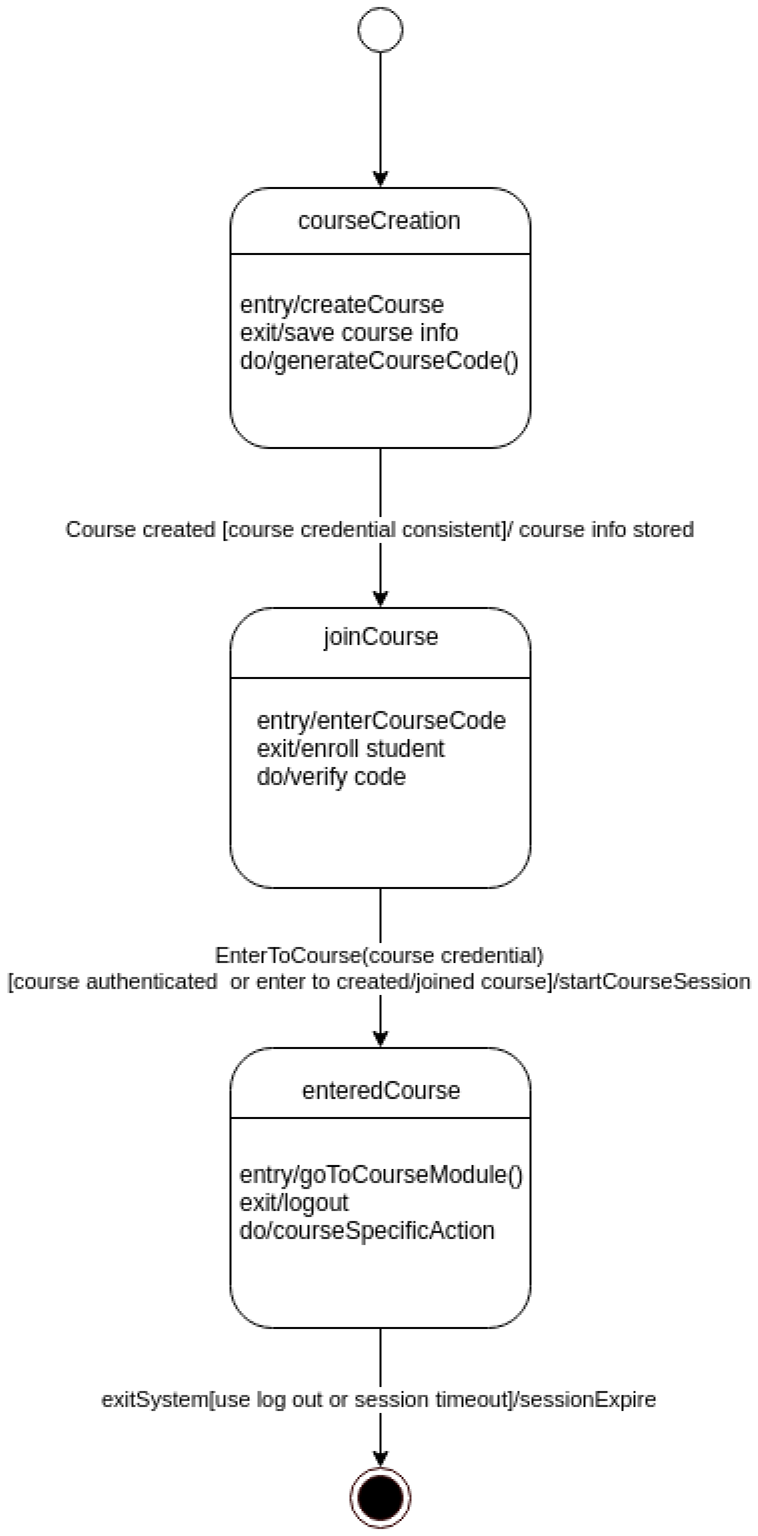

courseCreation
entry/createCourse
exit/save course info
do/generateCourseCode()
Course created [course credential consistent]/ course info stored
joinCourse
entry/enterCourseCode
exit/enroll student
do/verify code
EnterToCourse(course credential)
[course authenticated or enter to created/joined course]/startCourseSession
enteredCourse
entry/goToCourseModule()
exit/logout
do/courseSpecificAction
exitSystem[use log out or session timeout]/sessionExpire

Fig: 66

User

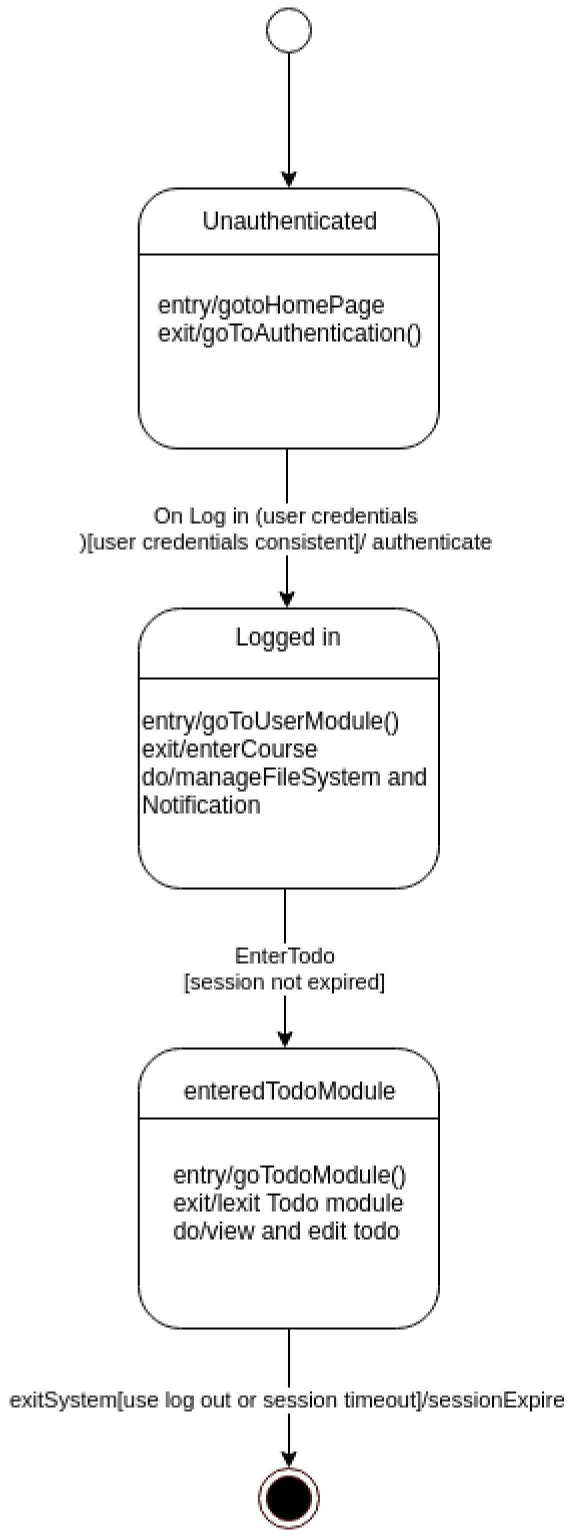

Unauthenticated
entry/gotoHomePage
exit/goToAuthentication()
On Log in (user credentials
)[user credentials consistent]/ authenticate
Logged in
entry/goToUserModule()
exit/enterCourse
do/manageFileSystem and
Notification
EnterTodo
[session not expired]
enteredTodoModule
entry/goTodoModule()
exit/lexit Todo module
do/view and edit todo
exitSystem[use log out or session timeout]/sessionExpire

fig:67

Instructor

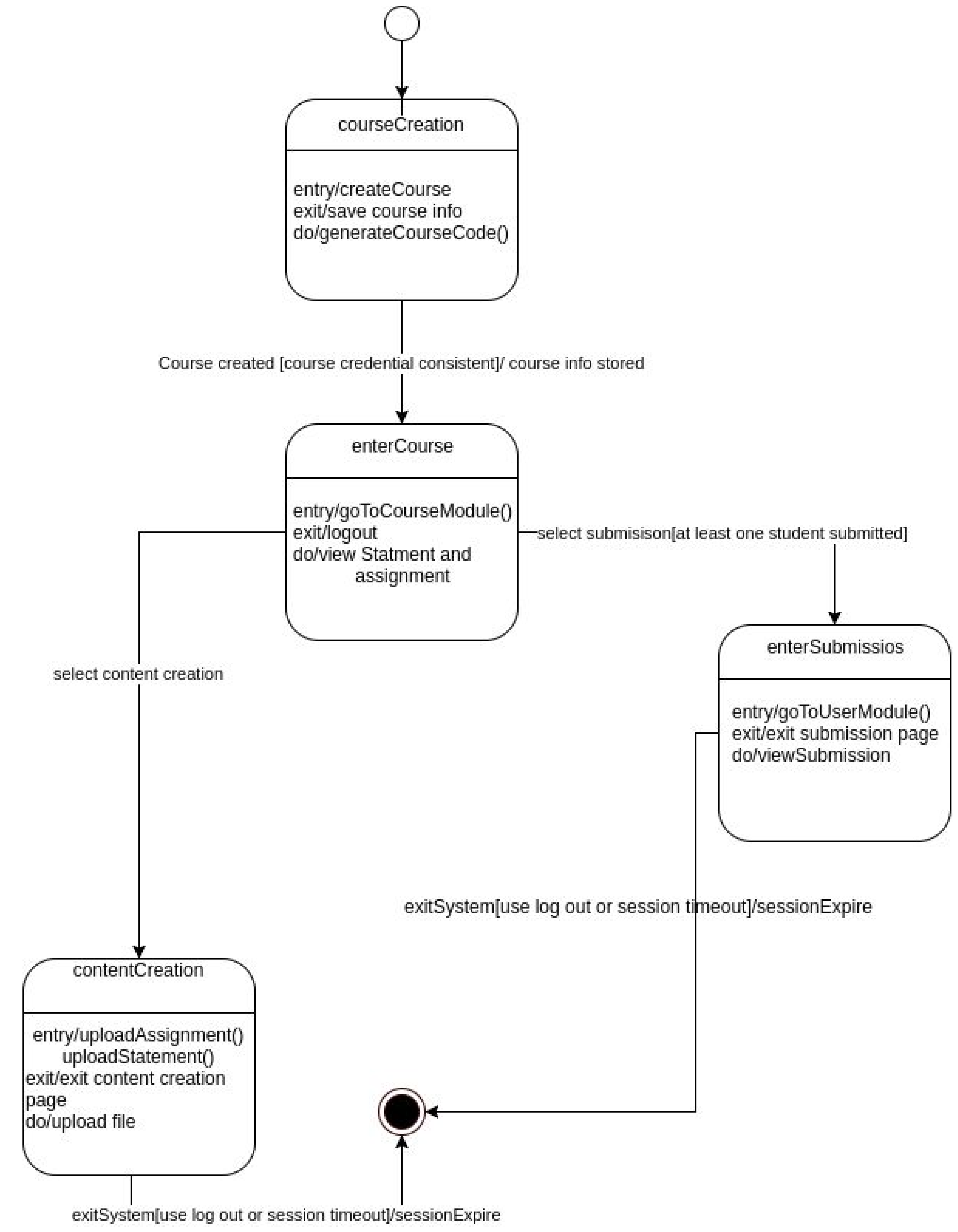

courseCreation
entry/createCourse
exit/save course info
do/generateCourseCode()
Course created [course credential consistent]/ course info stored
enterCourse
entry/goToCourseModule()
exit/logout
do/view Statment and assignment
select submisison[at least one student submitted]
enterSubmissios
entry/goToUserModule()
exit/exit submission page
do/viewSubmission
select content creation
exitSystem[use log out or session timeout]/sessionExpire
contentCreation
entry/uploadAssignment() uploadStatement()
exit/exit content creation page
do/upload file
exitSystem[use log out or session timeout]/sessionExpire

Fig: 68

Student

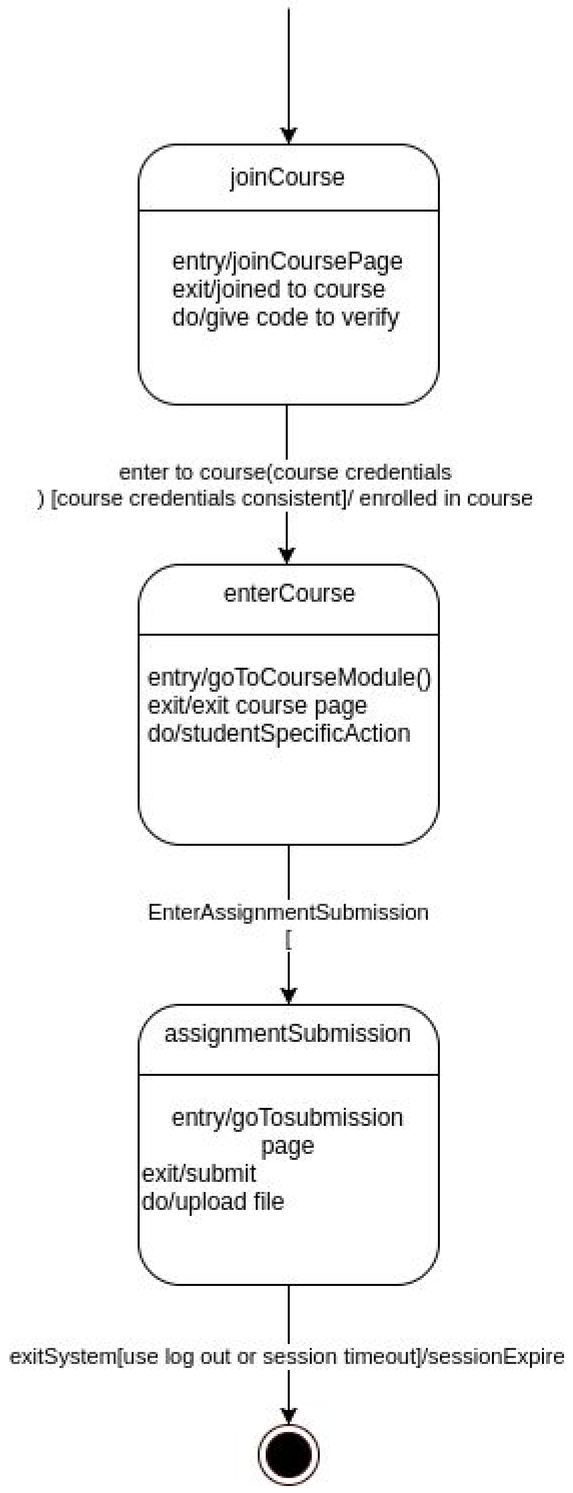
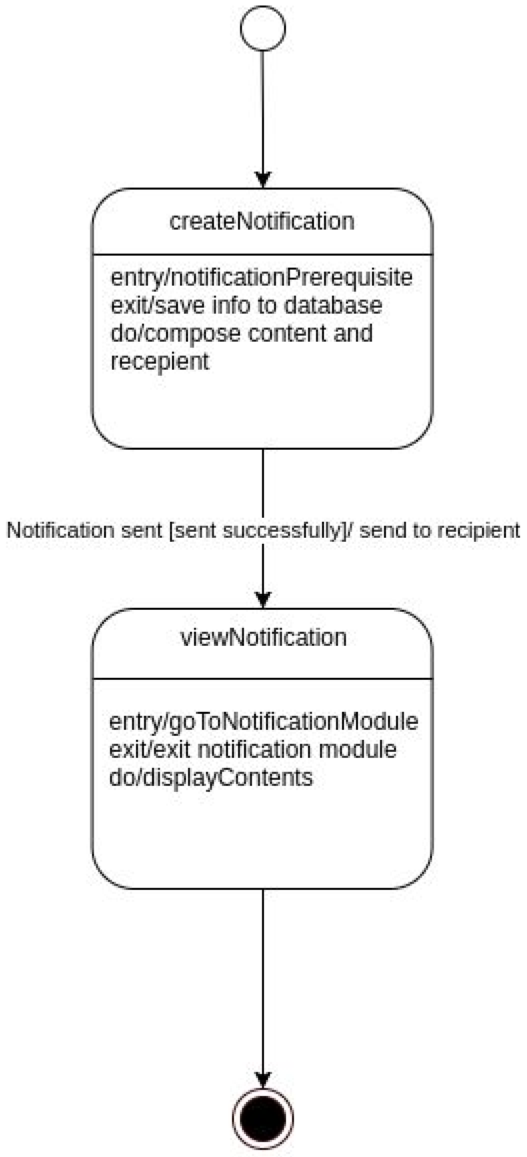


Fig: 69

Notification


createNotification
entry/notificationPrerequisite
exit/save info to database
do/compose content and
recepient
Notification sent [sent successfully]/ send to recipient
viewNotification
entry/goToNotificationModule
exit/exit notification module
do/displayContents

Notification fig:70

ToDo

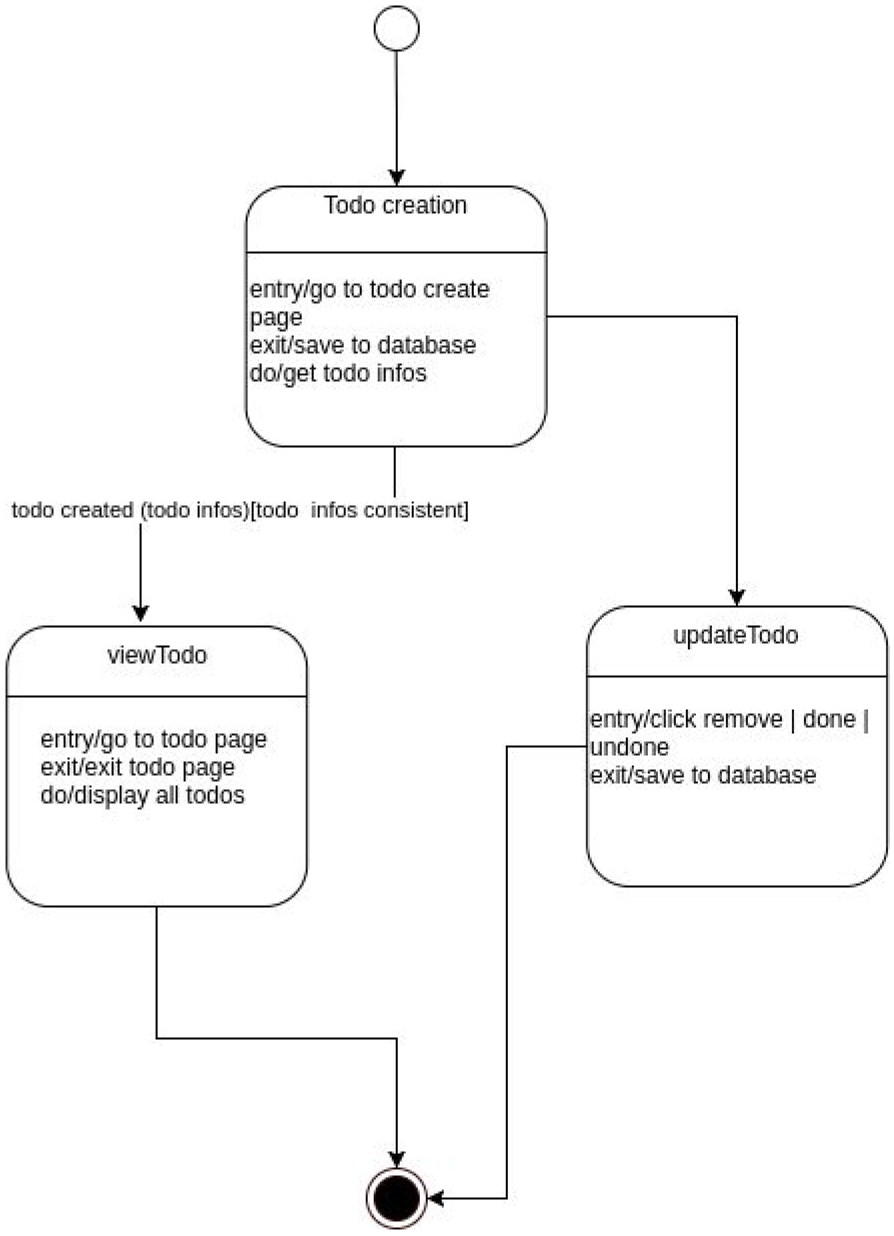

Todo creation
entry/go to todo create page
exit/save to database
do/get todo infos
todo created (todo infos)[todo infos consistent]
viewTodo
entry/go to todo page
exit/exit todo page
do/display all todos
updateTodo
entry/click remove | done | undone
exit/save to database

Fig:71

File

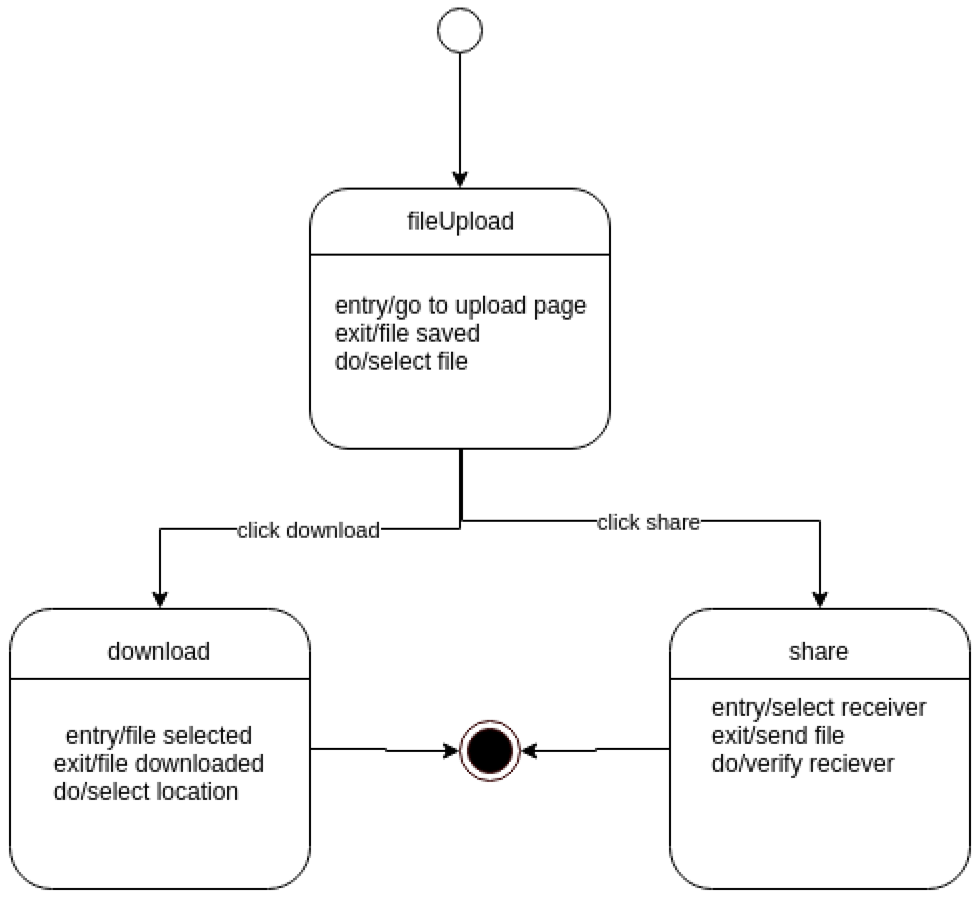
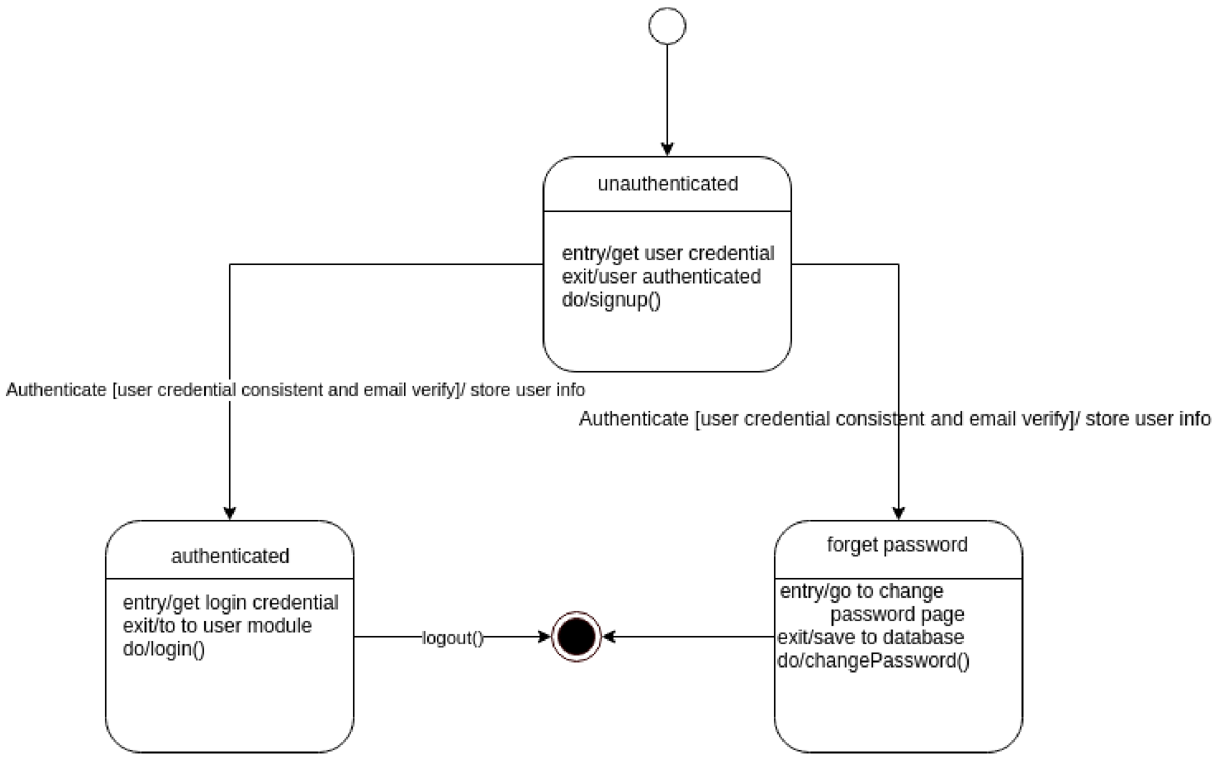

fileUpload
entry/go to upload page
exit/file saved
do/select file
click download
click share
download
entry/file selected
exit/file downloaded
do/select location
share
entry/select receiver
exit/send file
do/verify reciever

Fig:72

Authentication


unauthenticated
entry/get user credential
exit/user authenticated
do/signup()
Authenticate [user credential consistent and email verify]/ store user info
Authenticate [user credential consistent and email verify]/ store user info
authenticated
entry/get login credential
exit/to to user module
do/login()
logout()
forget password
entry/go to change password page
exit/save to database
do/changePassword()

Authentication

Fig:73

UserGUI

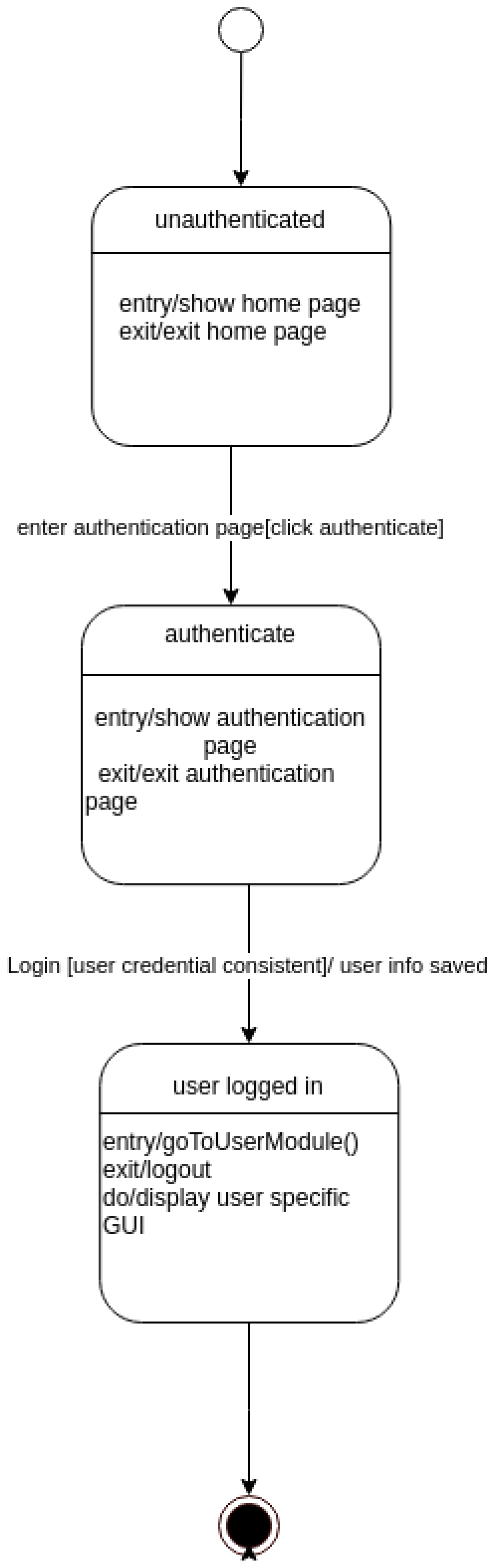

unauthenticated
entry/show home page
exit/exit home page
enter authentication page[click authenticate]
authenticate
entry/show authentication page
exit/exit authentication page
Login [user credential consistent]/ user info saved
user logged in
entry/goToUserModule()
exit/logout
do/display user specific GUI

Fig:74

CourseGUI

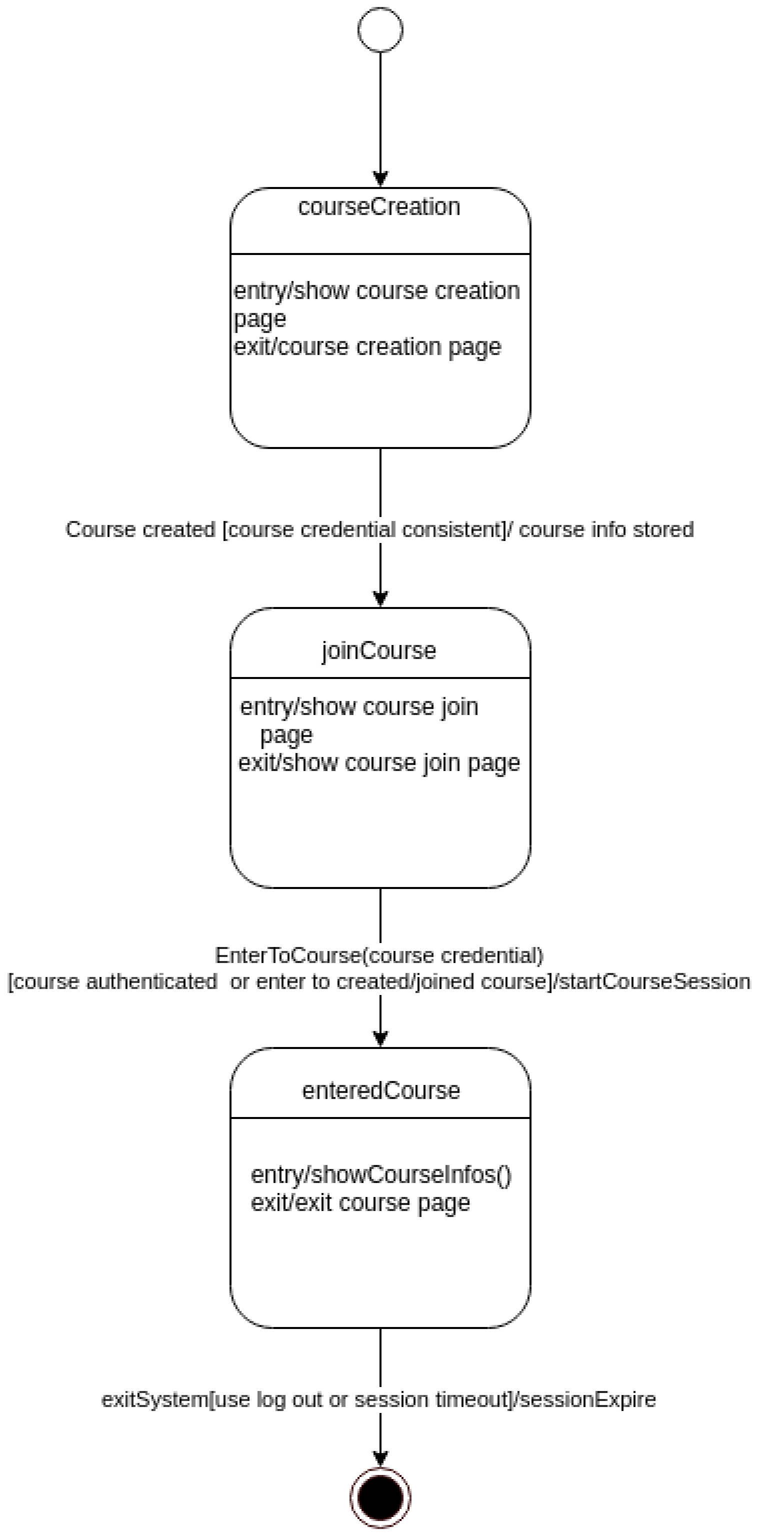

courseCreation
entry/show course creation page
exit/course creation page
Course created [course credential consistent]/ course info stored
joinCourse
entry/show course join page
exit/show course join page
EnterToCourse(course credential)
[course authenticated or enter to created/joined course]/startCourseSession
enteredCourse
entry/showCourseInfos()
exit/exit course page
exitSystem[use log out or session timeout]/sessionExpire

Fig:75

Database

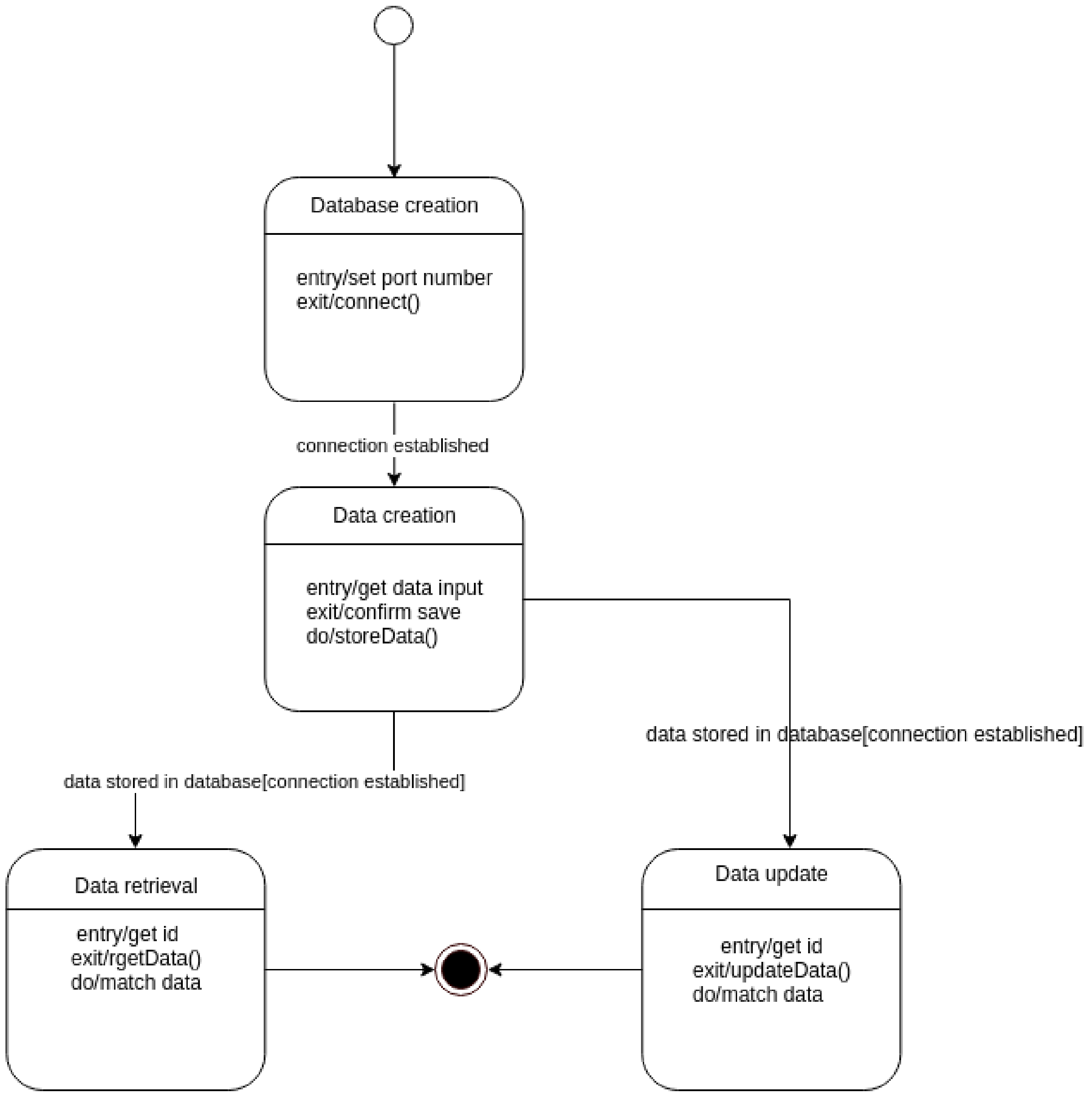


figure:76