Gebze Technical University
Computer Engineering

CSE 222
2017 Spring

HOMEWORK 1 REPORT

DENİZ CAN ERDEM YILMAZ
151044001

Course Assistant: Nur Banu Albayrak

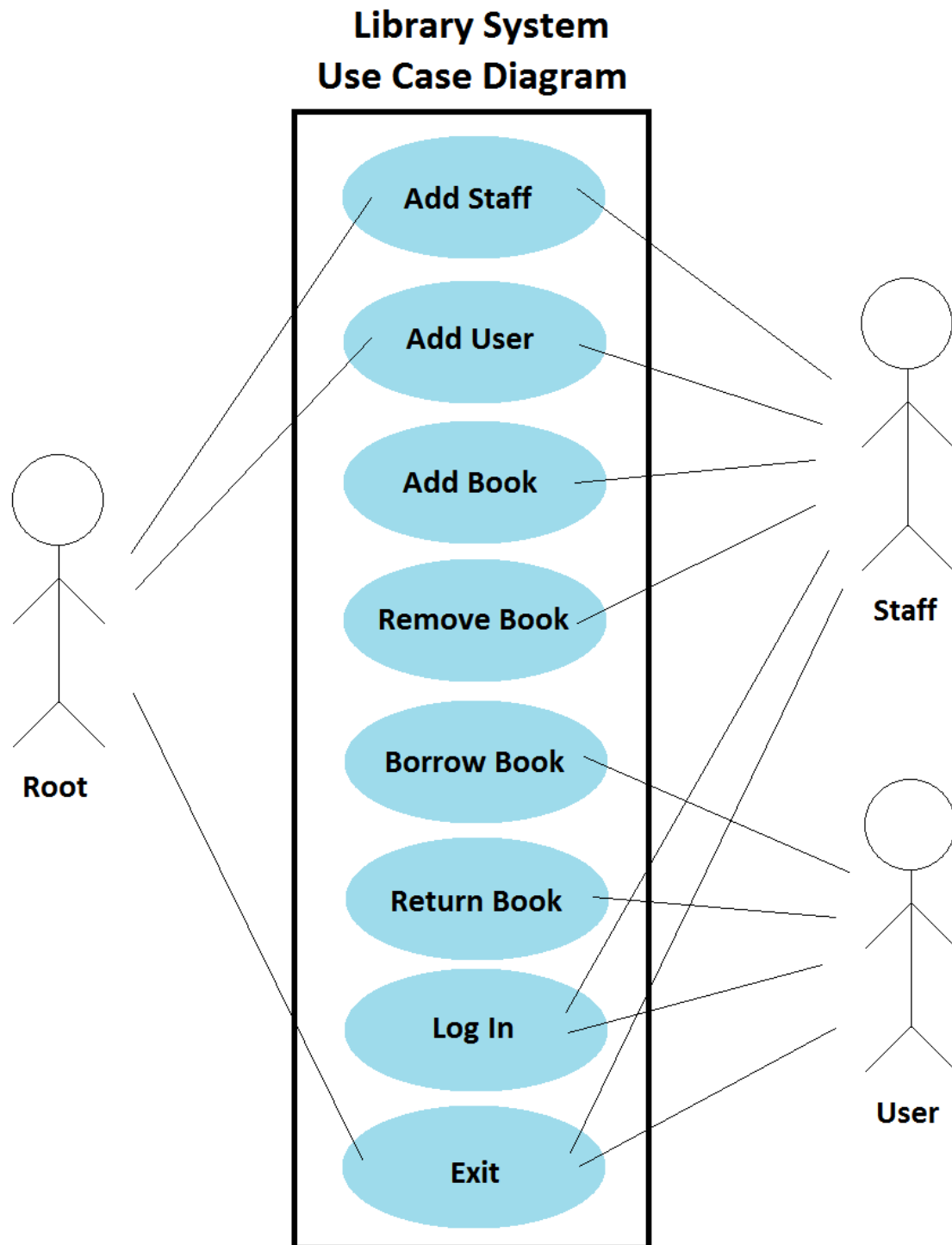# 1. System Requirements

A library system
Root (master user) for the first log-in
Library staff to add users and add/emove books
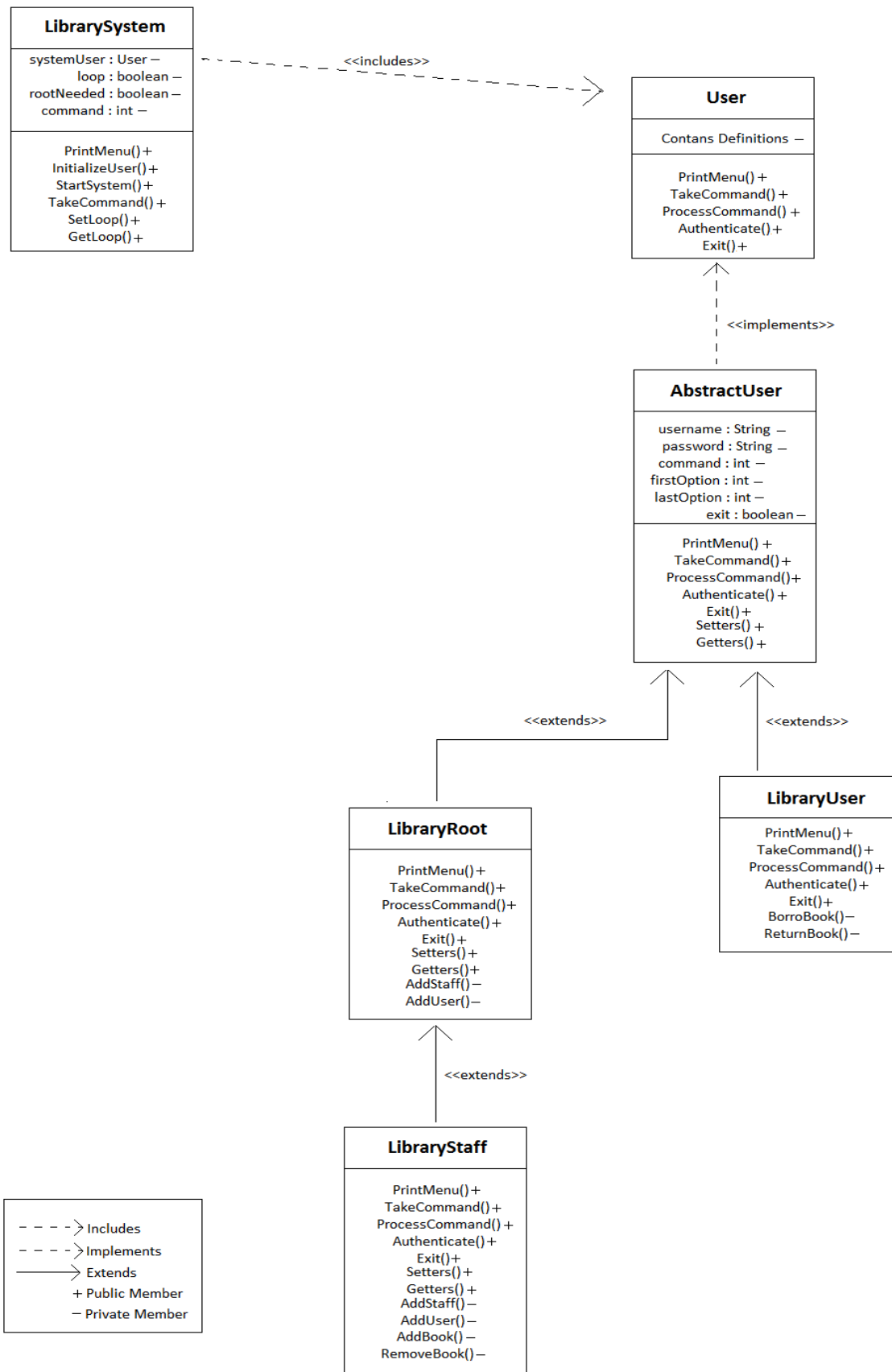Library user to use system to borrow or return books

2. Use Case Diagrams

PNG file can also be found in homework folder

Library System
Use Case Diagram

3. Class Diagrams

PNG file can also be found in homework folder

LibrarySystem

systemUser : User —
loop : boolean —
rootNeeded : boolean —
command : int —

PrintMenu()+
InitializeUser()+
StartSystem()+
TakeCommand()+
SetLoop()+
GetLoop()+

<<includes>>

User

Contans Definitions —

PrintMenu()+
TakeCommand()+
ProcessCommand()+
Authenticate()+
Exit()+

<<implements>>

AbstractUser

username : String —
password : String —
command : int —
firstOption : int —
lastOption : int —
exit : boolean —

PrintMenu()+
TakeCommand()+
ProcessCommand()+
Authenticate()+
Exit()+
Setters()+
Getters()+

<<extends>>   <<extends>>

LibraryRoot

PrintMenu()+
TakeCommand()+
ProcessCommand()+
Authenticate()+
Exit()+
Setters()+
Getters()+
AddStaff()—
AddUser()—

LibraryUser

PrintMenu()+
TakeCommand()+
ProcessCommand()+
Authenticate()+
Exit()+
BorroBook()—
ReturnBook()—

<<extends>>

LibraryStaff

PrintMenu()+
TakeCommand()+
ProcessCommand()+
Authenticate()+
Exit()+
Setters()+
Getters()+
AddStaff()—
AddUser()—
AddBook()—
RemoveBook()—

- - - -> Includes
- - - -> Implements
———> Extends
+ Public Member
— Private Member

4. Problem Solutions Approach

It is a simple library system. There must be some functionalities like creating books/users(and also staffs), removing books, borrowing books, returning books etc. OOP design used to handle this situation. Library itself and different user types implemented as objects. "User" interface is the origin of the different type of users. It has methods that all kind of users must have to make things work with library system. An abstract user class named "AbstractUser" has implemented the interface "User". Added some member variables to make things work on sub-classes. Class "LibraryRoot" and "LibraryUser" extended the abstract class "AbstractUser". They both have some basic methods to run system. "LibraryRoot" activates when there is no registered staff has found to add "LibraryStaff". Which is a class extended "LibraryRoot". They both had same methods for user register but "LibraryStaff" has also book add/remove methods. That's why it extends "LibraryRoot". Class named "LibrarySystem" has a member data named systemUser with type "User". From the menu program user chooses which type of account he/she wants to login. User and book informations stored inside folders named usr.csv and lib.csv. Root role can not be activated unless there is usr.lib or it has at least 1 staff registry. Files created automatically as empty with constructor of "LibrarySystem" if they do not exists

5. Test Cases

Program runned without csv files, they all created themselves as empty
Program has been tried to quit without staff registry while root access is on, program denied request
Program has been tried to quit after user registry while root access is on, program denied request
Program has been tried to quit after staff registrt while root access is on, program exited
Tried to log in as staff but member account informations entered, Access denied
Entered a long string while menu waits an integer, program requested input again
New book registry tried with an already existing book, another book created with ready status
User tried to borrow a borrowed book, system denied request
User tried to borrow an unexisting book, system denied request

6. Running and Results
Screenshot can be found in homework folder.

7. How can encapsulation be done using C?
Encapsulation , Object Oriented programlama dillerinin en önemli özelliklerindendir. C programlama dilinde bunu yapacak doğrudan bir yapı olmamasına karşın birkaç yapının birlikte kullanılmasıyla gerçekleştirilebilir.

Object Oriented daki class yapışana en yakın yapı olan struct yapısı ile header file sadece oluşturmak istediğimiz adı vererek ve sadece fonksiyon protatiplerini yazarak oluşturup daha sonra implement ettiğimiz source file da struct yapısı ile bir class ın member field kısımı olacak gibi içine bulunmasını istediğimiz memberları yerleştirebiliriz. Fonksiyonlarımızın implementlerinde memberlara doğrudan ulaştığımız source file daki struct ile tanımladıklarımızı header file daki struct a cast edersek class gibi kullandığımız yapımızı yapıyı sadece yazdığımız fonksiyonlarla ulaşılır hale getirebiliriz .Bu struct yapılarında void * kullanarak polymorphism i gerçekleştirebiliriz . Void * ların yerini alacak olan struct isimleriyle oluşturulmuş pointerlar (Person_t *)işlevsel olacaktır.