Software Requirements Specification

for

KSU CSWS

**Version 1.0 approved**

**Prepared by Yeojin Kim, Shiva Voruganti, Nidhi Shah, JunanZhao, Jean-Jacques Muteteke**

**Senior Project**

**10/2/2016**

**Table of Contents**

**Table of Contents**

**Revision History**

**1. Introduction**

1.1 Purpose

1.2 Document Conventions

1.3 Intended Audience and Reading Suggestions

1.4 Product Scope

1.5 References

**2. Overall Description**

2.1 Product Perspective

2.2 Product Functions

2.3 User Classes and Characteristics

2.4 Operating Environment

2.5 Design and Implementation Constraints

2.6 User Documentation

2.7 Assumptions and Dependencies

**3. External Interface Requirements**

3.1 User Interfaces

3.2 Hardware Interfaces

3.3 Software Interfaces

3.4 Communications Interfaces

**4. System Features (Use Cases)**

4.1 UML Diagram

4.2 Student Login

4.3 Faculty Login

4.4 Course Substitution Request - Student

4.5 Course Substitution Evaluation - Faculty

4.6 Managing Requests - Faculty

4.7 File Storage

4.8 E-Sign

4.9 E-mail notification

**5. Other Nonfunctional Requirements**

5.1 Performance Requirements

5.2 Safety Requirements

5.3 Security Requirements

5.4 Software Quality Attributes

5.5 Business Rules

# Introduction

## Purpose

Our objective is to make the processes of course substitution a lot better in terms of performance, time and flexibility between advisors and department coordinators. Our objective would be use digital and web technology to create and implement a better system where timeframe to process substitution forms would be much lower. As if now, the process takes at least 3 weeks to update in student's transcript profile. We want to create a system where students can log in and electronically fill out the substitution form and advisors and departments chairs can't take necessary actions such as approve, disapprove, comments and request other information.

## Document Conventions

There are logs of abstractions to represent in a more convenient way the objects and their behavior on the system. Every requirement statement is assumed to have its own priority as to define in most appropriate way the system behavior. In addition there are various figures that represent the described system, where it is needed, and serve only for better understanding of the deployment.

The conventions used to prepared the document is given below:

* Font: Times New Roman, size 11
* Main headings, Bold size 18
* Sub Headings, Bold size 14
* The system features are stated in a use case format.
* Our Priority are ranked from 0/10 to 10/10 with 10/10 meaning high priority. 0/10 ~ 4/10 is low priority which means the system feature can be implemented or verified more simple and easy. 5/10 ~ 7/10 is medium priority which means the system feature can be implemented little complicated and verified more difficult. 8/10 ~ 10/10 is high priority which means the system feature is very important and difficult to be implemented.

## Intended Audience and Reading Suggestions

This document is intended for all the members of CSWS team to use for references. This document is also intended for all the stakeholders such as advisors, Deans and department chairs who will be users our product. It is also intended for our professor Dr.Wagner.

The potential uses for each one of the reader types:

* Developer: The developer who wants to read, change, modify or add new requirements into the existing program, must firstly consult this document and update the requirements with appropriate manner so as to not destroy the actual meaning of them and pass the information correctly to the next phases of the development process.
* User: The user of this program reviews the diagrams and the specifications presented in this document and determines if the software has all the suitable requirements and if the software developer has implemented all of them.
* Tester: The tester needs this document to validate that the initial requirements of this programs actually corresponds to the executable program correctly.

## Product Scope // should finish this part

Project Vision Statement

Project Scope Document

* In Scope
* Out of Scope

## References

This document builds on the following references:

* IEEE Std 830-1998, [www.latech.edu/~box/ase/srs\_template.doc](http://www.latech.edu/~box/ase/srs_template.doc)
* KSU CSWS: <https://github.com/yeogina/senior_project>

# Overall Description

## Product Perspective

The system will be a web application. Essentially it will be an add on to the owl express. This will be a lot easier for people to access. Product Perspective varies from different users.The main function of the system is making course substitution process online. Our web application would be used by computer science advisors, department coordinators, department dean and students. Each user screen is different based on their credentials.

The student will log in and put in the request for the course substitution. They will be putting in the course number, year taken, what they want to replace it as and hit submit. The faculty will get an email notification and log in from owl express. He or she will see the request and accept/deny the request. Put in the comments that they need and hit submit. The faculty will be able to see a page with all the requests. The only thing that the student is able to see their request.

The follow are the main features of the CSWS:

* Request page: The students are able to make a request for the course they want to substitute.
* Acceptance or Denied section: The faculty can accept or deny the request made.
* E-signature: To make it official
* Comments: An area for extra comments or explanation as to why the request is being denied.
* Contact Information: If you are having an error with the system or need more help emails and/or numbers will be listed.
* FAQ: Frequent question that students or faculty may have about processing time and more.

## Product Functions

This web application has many functions.

* Create a secure a system using login credentials.
* Users should able to write and read information from tab menu
* User should able to receive announcements about application process
* Email notification should be sent out to faculty members and students once the application is submitted.
* Based on the credentials, application process may skip a step at a time

## User Classes and Characteristics

CSWB is intended to be used by many users. Based on the respective position and title, users may see different options and features. Our web system primarily divided into two different sections. Faculty members and students.

Faculty members have more access than students. They can authorize and take necessary actions based on provided information. They can not also verify given information but also add comments and provide other important details for students. Faculty member should have two options or buttons in course substitution tab, where first option should give users to input necessary information to complete the application. The second option would have information about the submitted applications and their process.

Students have limited access compare to faculty members. Once student clicks on course substitution tab, they will have a webpage, where Students can input name, student id, course name etc. and initialize the application. Students may check their application process. Students will receive email notifications about the process.

## Operating Environment

CSWS would be platform independent. It will be compatible on Microsoft Windows, Mac OS X, and Linux operating systems. Users can use any browsers such as Firefox, Internet Explorer, Microsoft Edge, Chrome etc to access domain. Our system can also be accessed from Mobile phones as well.

## 2.5 Design and Implementation Constraints

Our biggest constraint will be whether or not we are able to differentiate between the student and the admins. We may not have the exact permission to do so. Kirk inman will have to find out more about the whole process and if we are able to get permission. Because there are some flags which are able to differentiate. Another one will be getting permission to get access to LDAP system. The entire project is determined on these two factors.

## User Documentation

There will be an area for the users on how the online processes works. They can read that document to guide them with any technical questions. This document will include, how request the course substitution and how to accept it. It will also include what steps they need to take if the student wants to cancel the request and how to deny the request. There will be an FAQ area which helps both the students and faculty.

The page will something along the lines of:

* Student Help
  + How to Make a Request?
  + What to expect?
  + Average Amount of Time.
* Faculty Help
  + How to accept?
  + Comment Section.
* FAQ
  + Contact info
  + What happens if the course in denied?
  + List of acceptable course substitution

## Assumptions and Dependencies

## The basic assumptions for CSWS are:

* LDAP system is accessible
* We are able to implement everything into Owl Express
* All team members work together
* There will be a way to differentiate between the credentials.
* Student and Faculty can edit their section
* It appears in a PDF format
* There will be a FAQ and/or help page for the users

The task dependency in this project is that if the first task is not completed then the second task can not start. For example if the student does not finish the request and click submit the advisor cannot get a notification or see the request. When the advisor accepts the request and hits submit the Department chair gets a notification. And he has to do the same. In the event that the advisor does not accept the request. It will show up as denied and then the student will have to see the advisor. First, we must have access to the system in order to start coding. Once everything is working we well we will then find a way to implement it into owl express. It will be “Finish to Finish” task dependency.

# External Interface Requirements

## User Interfaces

Once the user entered the url to access our web application, The user will see a login screen where user have to enter login credentials. Based on user credentials, information displayed may vary. for example, student user only able to see application . faculty member can not only see the application but also see’s a tab where user can see application process.

From student Interface, Once the student clicks on course substitution tab, The system would let user input name, student id, course name, course number, completed year, semester taken, grade student received etc to finalize and send an notification to department advisor, who further accepts or rejects the application, once the application is accepted, an notification would be sent out to department coordinator, who verifies the information and decides if a taken course can be substituted for a requested course.

From faculty Interface, Once the Faculty member clicks on course substitution tab, they see two options or buttons. the first option would let user input name, student id, course name, course number, completed year, semester taken, grade student received etc to finalize and send an notification to appropriate person. The system still takes count who initialized it, such as the faculty member or student. If a faculty advisor initialized the application and submitted, then an notification would be sent out to department chair and so on. The verification process varies based on who initialized the student application. if an advisor is initialized application, then advisor shouldn’t receive email announcement,so the entire step would be skipped. The second option would show the process of submitted applications.

## Hardware Interfaces

The Course System Web System will run in all computers. It will follow the Web developement Protocols such as TCP/IP, FTP,SMTP . The CSWS system will come into interaction with memory of the machine it is running on, the network that is used to access the Internet, the operating system that is currently installed on the machine the system is running on, and other system utilities. Network interfaces could include the socket interfaces, memory management interfaces could include memory-mapped file interfaces, memory locking, or memory protection. Similarly, operating system interfaces could include the process interfaces, clocks and timers, as well as threads.

The basic hardware requirements will require a processor speed of 266 MHz, with a free available 300MB RAM on client side. Input devices such as keyboard and mouse will be required to interact with the application. A monitor and stable internet connection will be required to view the user interface.

**Our Server hardware Specifications are listed below.**

**CPU Specifications**

**Command :** /$ cat /proc/cpuinfo

**processor :** 0

**vendor\_id :** GenuineIntel

**cpu family :** 6

**model :** 45

**model name :** Intel(R) Xeon(R) CPU E5-2690 0 @ 2.90GHz

**stepping :** 7

**microcode :** 0x710

**cpu MHz** : 2899.999

**cache size :** 20480 KB

**physical id :** 0

**siblings** : 1

**core id :** 0

**cpu cores :** 1

apicid : 0

**initial apicid** : 0

**fpu** : yes

**fpu\_exception** : yes

**cpuid level** : 13

**wp** : yes

**flags** : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts mmx fxsr sse sse2 ss syscall nx rdtscp lm constant\_tsc arch\_perfmon pebs bts nopl xtopology tsc\_reliable nonstop\_tsc aperfmperf pni pclmulqdq ssse3 cx16 pcid sse4\_1 sse4\_2 x2apic popcnt tsc\_deadline\_timer aes xsave avx hypervisor lahf\_lm ida arat epb pln pts dtherm tsc\_adjust

**bogomips :** 5799.99

**clflush size :** 64

**cache\_alignment** **:** 64

**address sizes :** 42 bits physical, 48 bits virtual

**RAM/Memory Specifications:**

**total used free shared buff/cache available**

**Mem:** 7824 285 4528 80 3010 7155

**Swap:** 8079 0 8079

**File Disk Specifications:**

**Filesystem Size Used Avail Use% Mounted on**

/dev/mapper/rhel-root 52G 3.2G 49G 7% /

devtmpfs 4.1G 0 4.1G 0% /dev

tmpfs 4.2G 0 4.2G 0% /dev/shm

tmpfs 4.2G 85M 4.1G 3% /run

tmpfs 4.2G 0 4.2G 0% /sys/fs/cgroup

/dev/mapper/rhel-home 26G 34M 26G 1% /home

/dev/sda1 521M 144M 378M 28% /boot

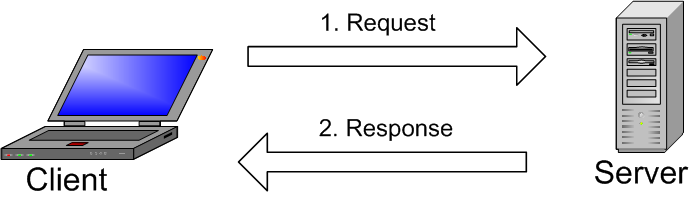
tmpfs 821M 0 821M 0% /run/user/1000

## Software Interfaces

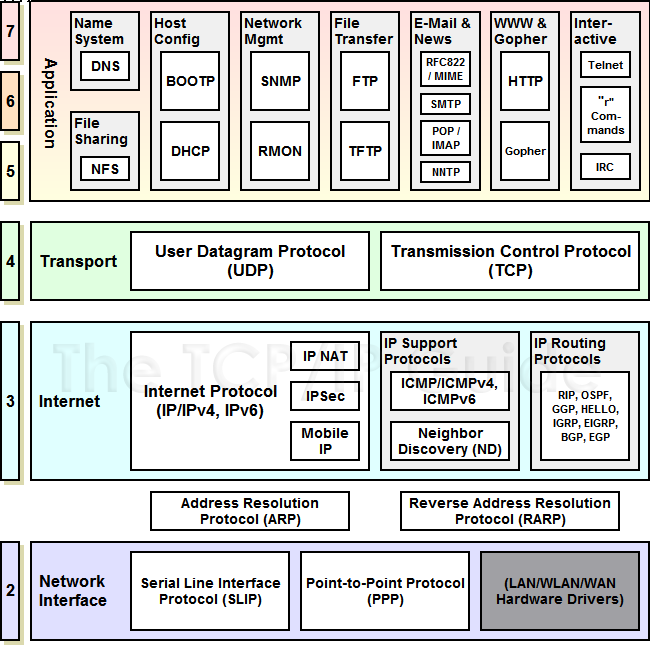
CSWS would be will interact with memory of the computer it is running on, the internet connection established, and the operating system installed on the computer the system is being used on. CSWS is platform independent, It can run on any system which has stable internet browser and internet connection to interact with the website. User needs to have proper credentials to access the website.

The frontend website would be interacting with backend server system to verify the credentials and give appropriate access to user. Our team would be using LAMP archetypal model of web service solution stacks, short for the Linux operating system, the Apache HTTP Server, the MySQL relational database management system (RDBMS), and the PHP programming language, since solutions stack is most useful for web application and dynamic websites.

CSWS would be using linux REDHAT operating system for server. Our web application would also be using API’s and built libraries for for login and automation purposes. For example, our application would be using SMTP for sending announcements to users and faculty members. Another example would be for data items or messages would be login process, where client initially inputs username and password and sends it to server and server reviews the requests and process the request and returns response. User must enter correct username and password in order to access the secure information. Our application would be using TCP/IP Protocols for network interface and communication. we may have to use different protocols to do different things. It may also use different services such as File Sharing, Print sharing, Directory Services (LDAP), and User administration.

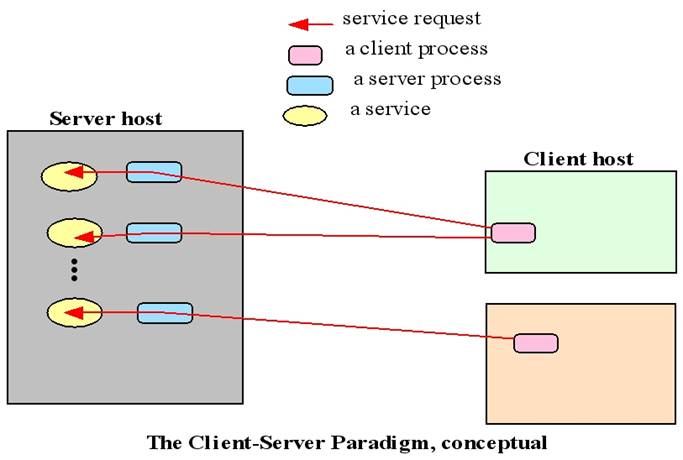


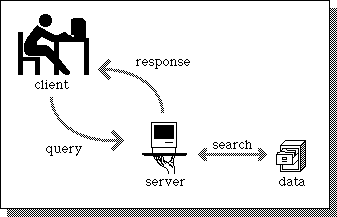
**Figure1. Client and Server model**

**Figure2: TCP/IP Protocols**

## Communications Interfaces

Our application would be using client-server paradigm. Where client initiates the connection with server and server process and delivers the requests. A server is a high-performance host that is a registering unit and shares its resources with clients. A client does not share any of its resources, but requests a server's content or service function.Each instance of the client browser can send data requests to our server. In turn, the server can accept these requests, process them, and return the requested information to the client.

**Figure3: Client and Server Paradigm**



**Figure4: Client is requesting and server is processing and delivering the request.**

This application may use the following network protocols including but not limited to IP, ARP, SMTP, DNS, FTP, LDAP.

* IP - Internet Protocol (IP). Except for ARP and RARP all protocols' data packets will be packaged into an IP data packet. Provides the mechanism to use software to address and manage data packets being sent to computers.
* ARP - Address resolution protocol (ARP) enables the packaging of IP data into ethernet packages. It is the system and messaging protocol that is used to find the ethernet (hardware) address from a specific IP number. Without this protocol, the ethernet package could not be generated from the IP package, because the ethernet address could not be determined.
* SMTP - Simple Mail Transfer Protocol (SMTP).
  + MUA - Mail user agent. This is the program a user will use to type e-mail. It usually incorporates an editor for support. The user types the mail and it is passed to the sending MTA.
  + MTA - Message transfer agent is used to pass mail from the sending machine to the receiving machine. There is a MTA program running on both the sending and receiving machine. Sendmail is a MTA.The MTA on both machines use the network SMTP (simple mail transfer protocol) to pass mail between them, usually on port 25.
  + LDA - Local delivery agent on the receiving machine receives the mail from its MTA. This program is usually procmail.
  + Mail notifier - This program notifies the recipient that they have mail. Normally this requires two programs, biff and comsat. Biff allows the administrator or user to turn on comsat service.

* DNS - Domain Name Service, allows the network to determine IP addresses from names and vice versa.
* FTP - File Transfer Protocol (FTP). Allows file transfer between two computers with login required.
* LDAP (Lightweight Directory Access Protocol) is a software protocol for enabling anyone to locate organizations, individuals, and other resources such as files and devices in a network, whether on the public Internet or on a corporate intranet.

# System Features (Use Cases)

## UML Diagram use case UML.jpg

## Figure 1. UML Diagram

Figure 1. The UML diagram is used to display actors and their interactions with the system. In this UML diagram we have a list of actors (KSU OWL server, KSU CSWS Manager, KSU CSWS, Student, Advisor, Program Coordinator, and Department Chair), and note that actors such as KSU OWL server and Degree Auditor are existing systems. There are a number of use cases in our system, and they serve as functions of our system. Use cases can be independent functions, they can “uses” another use case to complete their task, or they can “extends” another use case, meaning they are a subtask of that function. The file storage is a saved document of the course substitution forms from student and faculty.

## Student Login

**ID and name**: UC-1 Student Login

**author:** Yeojin Kim

**data created:** 9/27/2016

**actors:** KSU Student, KSU OWL Server

**summary:** This is for KSU students who would like to login so that they want to request the course substitution. If users enter invalid ID and password, the users cannot be logged in.

**trigger:** Users enter valid ID and password. Users press “Login with Student” button.

**preconditions:** User

**postconditions:** The system should log the students in.

**flow of events:**

* **normal flows:**

1. Students enter ID and password
2. Students press “Login with Student” button
3. Account validation occurs
4. Students are successfully logged in and can see the “New Request”, “My Request”, and “LogOut” tab.

* **alternative flows:**

1a. Users press“Forgot Password”

1b. Users enter ID

1c. Password is emailed.

* **exceptional flows:**

2a. Users enter Student ID and password but press “Login with Faculty” button

2b. Input is not valid and users cannot be logged in.

3a. If the ID or password is not correct, the users will see an error notification.

**priority**: 7/10

**assumptions:** Users are KSU students.

## Faculty Login

**ID and name**: UC-2 Faculty Login

**author:** Yeojin Kim

**data created:** 9/27/2016

**actors:** KSU CS Adviser, Program Coordinator, Department Chair, KSU OWL Server

**summary:** This is for KSU CS faculties who would like to login so that they can see all the course substitution requests and approve or deny the course substitution requests.

**trigger:** User presses login button with user ID and password.

**preconditions:** Users have faculty accounts. Faculties (Adviser, Program Coordinator, Department Chair) must enter correct user ID and password.

**postconditions:** The system should log the faculty in.

**flow of events:**

* **normal flows:**

1. Faculties enter ID and password.
2. Faculties press the “Login with Faculty” button.
3. Account validation occurs.
4. Faculties are successfully logged in and can now see all the course substitution requests from students sorted by time.

* **alternative flows:**

1a. Users press “forgot password”

1b. Users enter user ID

1c. Password is emailed.

* **exceptional flows:**

2a. Users enter incorrect/invalid user ID/password

2b. Users can not be logged in

**priority:** 7/10

**assumptions:** Users are KSU CS Faculties

## Course Substitution Request (Student)

**ID and name**: UC-3 Course Substitution Request

**author:** Yeojin Kim

**data created:** 9/29/2016

**actors:** Students

**summary:** Users can see the “New Request” button. After users click the button, they can see a web page with various text inputs. They should fill in all the required information such as KSU student ID, E-mail, and Course name... etc. After they finish filling out the form, they click “Submit” button. Now the course substitution request form will be stored in the file storage.

**trigger:** Users press the “New Request” button. After they’ve done, press the “Submit” button.

**preconditions:** Users logged in as a student account, and they will see the categories: my requests, new requests, help, and logout.

**postconditions:** After submit the new request, users will receive the email that notifies them “You have submitted the course substitution request form successfully!”

**flow of events:**

* **normal flows:**

1. “New Request” page is opened
2. Students type in all the required informations to create a course substitution request form.
3. Students should sign by E-signature feature (UC-7 E-Sign).
4. The Course Substitution Request Form will be generated when students press the “Submit” button.
5. The Course Substitution Request Form will be stored in the section of “Active” list on the File Storage feature (UC-6 File Storage).

* **alternative flows:**

2a. Students press “clear all” button to re-type in the new form.

4a. Students press “Cancel” button.

4b. “Are you sure you want to cancel the request form?” message is popped up.

4c. Student can chooses Yes or No

* **exceptional flows:**

1a. Students press “LogOut” button.

1b. The webpage browses the “Login” field, and whatever the information they

are filling in before they log out can not be loadable.

4a. Error message popes up when students press “Submit” button with leaving

any blank on the required information that they must fill in.

**priority:** 8/10

**assumptions:** User knows about their course requirements and course curriculum, and they have provided supplemental information about the completed course.

## Course Substitution Evaluation (Faculty)

**ID and name**: UC-4 Course Substitution Evaluation

**author:** Yeojin Kim

**data created:** 9/29/2016

**actors:** CSAdviser, CS Program Coordinator, CS Department Chair, File Storage

**summary:** First, CS advisers initiate the course substitution requests. They look over the request if the requests are approvable. If the requests are approvable, they sign on the evaluation page. Second, CS program coordinator receives the course substitution evaluations from the advisors. The coordinator looks over the evaluations if the approval has any errors. If the evaluations has no error, the program coordinator signs on the evaluation page. Third, CS department chair receives the course substitution evaluation from the program coordinator. If the evaluations have no error, the chair signs for the permission to approve the course substitution requests. Finally, after the permission, the evaluation form will be generated as pdf file and stored in “File Storage”. Denial from any faculties will be notified by email to students, otherwise students can check their status by clicking the “My Requests” tab on the home.

**trigger:** Users press “Managing Requests” button,

**preconditions:** Faculties (adviser, program coordinator, and department chair) have gotten requests from students.

**postconditions:** The course substitution evaluation is generated and stored in the File storage. The notification for the denial of requests will be emailed.

**flow of events:**

* **normal flows:**

1. Advisers open “Awaiting My Review” tab.
2. Advisers can see all the new requests from students in time order (oldest will be on the bottom and newest will be on the top).
3. Advisers should check if the requests is approvable.
4. Advisers add additional comments on the note section if necessary and put signature using Electronic Signature feature (UC-7 E-Sign).
5. Advisers press “Approve and Forward” button.
6. The Course Substitution Evaluation Form is generated by adviser, and the form will be stored in the “File Storage”, also the form is now transferred to the list in the “I’ve Submitted” tab.
7. Program coordinator opens “Awaiting My Review” tab.
8. Program coordinator can see all the new requests that are already passed by advisers with signature.
9. Program coordinator should check lastly if the evaluations are approvable.
10. Program coordinator puts the signature using Electronic Signature feature (UC-7 E-Sign).
11. Program coordinator presses “Approve and Forward” button.
12. The revised Course Substitution Evaluation Form is generated by program coordinator, and the form will be stored in the “File Storage”, also the form is now transferred to the “I’ve Submitted”
13. Department Chair opens “New Requests” tab.
14. Department Chair can see all the new requests that are already passed by advisers and program coordinator with signatures.
15. Department Chair checks the course substitution evaluation form and puts on the signature using Electronic Signature feature (UC-7 E-sign).
16. Department Chair presses “Submit” button.
17. The final Course Substitution Evaluation Form is generated by Department Chair.
18. The evaluation form, has all three signatures from adviser, program coordinator, and department chair, will be stored in the section “Approved” in the “File Storage” with file name, “student ID number\_file number”(000679386\_01).
19. When the course substitution evaluation is approved by department chair, the KSU CSWS will notify to student, adviser, and department chair. They can check the approval from the “My Requests” tab and email notification as well.

* **alternative flows:**

4a. Advisers add brief reasons why they cannot approve the requests.

4b. Advisers press “Deny” button.

4c. The denied course substitution evaluation form is generated and stored in the

“Denied” section in the “File Storage”.

4d. KSU CSWS will notify to student that the request is denied, and student can

see the notification on the “My Request” tab and receives email notification

as well.

10a. Program coordinator adds brief reasons why this request cannot be approved.

10b. Program coordinator presses “Deny” button.

10c. The denied course substitution evaluation form is generated and stored in the

“Denied” section in the “File Storage”.

10d. KSU CSWS will notify to student and adviser that the request is denied, and

they can see the notification on the “My Request” tab and receives email

notification as well.

15a. Department chair adds brief reasons why this request cannot be approved.

15b. Department chair presses “Deny” button.

15c. The denied course substitution evaluation form is generated and stored in the “Denied” section in the “File Storage”.

15d. KSU CSWS will notify to student, adviser, and program coordinator that the

request is denied, and they can see the notification on the “My Request” tab

and receives email notification as well.

* **exceptional flows:**

4, 10, 15a. Users press “Deny” button or “Approve and Forward” button without

their signature.

4, 10, 15b. The error message comes out “Your Signature is Required.”

**priority:** 9/10

**assumptions:** Faculties should check thoroughly not only if the degree works but also the supplemental information are suitable for approving the course substitution request.

## Managing Requests (Faculty only)

**ID and name**: UC-5 Manage Requests

**author:** Yeojin Kim

**data created:** 9/29/2016

**actors:** Adviser, Program Coordinator, Department Chair, File Storage

**summary:** Managing Request is the feature that extends from the “File Storage”. When users press “Managing Request”, they can see the several tabs to manage: “Awaiting My Review”, “I’ve Submitted”, and “Search Results”. The individual faculties can look their awaiting new requests by clicking “Awaiting My Review”. Also they can search every requests by type the student ID numbers so that they can check the request history.

**trigger:** Open “Managing Requests”, and Press “Awaiting My Review”, Open “I’ve Submitted”, and “Search Results” tabs.

**preconditions:** N/A

**postconditions:** N/A

**flow of events:**

* **normal flows:**

1. Users click on the “Managing Request” to access or view all requests.
2. Users click on the “Awaiting My Review” to see all new requests.
3. Users can view all the requests submitted by clicking the “I’ve Submitted” tab.
4. Users can view all the requests awaiting his/her approval and request submitted for approval in the “Search Results” tab.
5. Showing “All Requests” will display both active and completed requests. Approved Requests will display with a check mark in the status field, while denied requests will display an “X”. Pre-approved requests will display an exclamation point within a triangle.

* **alternative flows:**

2a. Users can also hover over ‘i’ icon on the Status column to view the

Workflow Steps (1. Adviser, 2. Program Coordinator, 3. Department Chair).

Note that different programs may have variations of Workflow Steps.

3a. A users can also “Edit” and “Delete” request in this tab. If users have many

requests, there will be page numbers at the bottom of the table as well.

4a. Once requests have been approved and forwarded, users will not be able to

edit or delete requests.

4b. Search results can be filtered to Open Requests, Pre-Approved, or All

requests just as the other tabs can.

4c. With all request tabs, users have the option of changing the results to show

only “Pre-Approved Requests” or “All Requests” by changing the Filter

option above the list of requests.

* **exceptional flows:**

4a. If no requests have been submitted for that particular student, no results

will be shown in the table.

**priority:** 9/10

**assumptions:** Only CS Adviser, Program Coordinator, and Department Chair can access Manage Requests features.

## File Storage

**ID and name**: UC-6 File Storage

**author:** Yeojin Kim

**data created:** 9/29/2016

**actors:** Degree Auditor, KSU CSWS

**summary:** The purpose of this use case is to store all the database. “File Storage” consists of the Archive (Denied and Approved), New Requests, and Pre-approved.

**trigger:** “New Requests”,“Pre-approved”, “Denied”, “Approved”

**preconditions:** N/A

**postconditions:** Once the faculties submit the course substitution request evaluation, the evaluation form will be stored as pdf file and cannot be deleted or edited by faculties.

**flow of events:**

* **normal flows:**

1. Students’ Course Substitution Requests Form is stored on the “New Requests” section.
2. The Course Substitution Evaluation Form by advisers and program coordinator is stored in the “Pre-approved” section.
3. The Course Substitution Evaluation Form by Department Chair is stored in the “Approved” section.
4. Denied course substitution requests by any faculties are stored in the “Denied” section.

* **alternative flows:** N/A
* **exceptional flows:** When faculties try to delete the one of the files from “Archive” folder, it would not be allowed to.

**priority:** 8/10

**assumptions:** There is sufficient capacity for storing tons of data. Once the file is stored, faculties cannot access to delete or edit that file. Only system administrative staff can access or maintain the files.

## E-Sign

**ID and name**: UC-7 E-Sign

**author:** Yeojin Kim

**data created:** 9/30/2016

**actors:** Adviser, Program Coordinator, Department Chair, Course Substitution Evaluation System.

**summary:** This feature is to make it easy for users to implement a digital, electronic workflow for all the paperwork processes, giving them the speed, efficiency and flexibility that paper processes have never delivered. Users (advisers, program coordinator, and department chair) will use this feature to sign on the course substitution request evaluation to approve or deny the requests. After they put the signature using “E-Sign” system, the forms will be generated.

**trigger:** Click “E-Signature” button.

**preconditions:** Before click the “E-Signature” button, faculties should finish to review the requests to evaluate and approve or deny UC-4 Course Substitution Evaluation.

**postconditions:** N/A

**flow of events:**

* **normal flows:**

1. Users click the “E-Signature” button.
2. E-Sign box opened.
3. Users put signature inside of box using their finger, and press “Done” button.
4. The signature is added at the bottom of the evaluation form.
5. Users click the “Deny” or “Approve and Forward” button.

* **alternative flows:**

3a. Users press “Clear” button when they want to erase the signature and redo it.

* **exceptional flows:** N/A

**priority:** 8/10

**assumptions:** Users may not use any “robot”, “spider” or other automatic or manual device or process for the purpose of compiling information on the site for purposes other than for a generally available search engine. Users may not redistribute, retransmit, republish, copy, duplicate, download or otherwise use or exploit any content on the site except for purposes permitted.

## Email Notification

**ID and name**: UC-9 Email Notification

**author:** Yeojin Kim

**data created:** 9/30/2016

**actors:** Students, Course Substitution Evaluation, Adviser, Program Coordinator, Department Chair

**summary:** The purpose of this is so that the users can receive notification of their requests status via email after they submit the Course Substitution Form so that they can be sure their requests have gone through.

**trigger:** UC-1 Student Login, UC-4 Course Substitution Evaluation

**preconditions:** The Course Substitution Request Form is generated, and the file is stored in the “File Storage”.

**postconditions:** Email with brief description of requests status is sent out.

**flow of events:**

* **normal flows:**

1. Before users press “Submit” button or “Approve and Forward” button, they put check mark by clicking the checkbox for “I Want to Receive Email Notification”.
2. Users press “Submit” button or “Approve and Forward” button.
3. The Requests/Evaluation form will be generated, and the email is sent out.

**priority:** 5/10

**assumptions:** Email address is correct.

# Other Nonfunctional Requirements

## Performance Requirements

* The system GUI could change its theme color according to user’s preference (optional).
* The system should process the user’s login information in no more than one second.
* The system should take no more than one second to logout user from system.
* The system should take no more than five seconds to generate a PDF version of the substitution application after student’s submission.
* The system should take no more than ten seconds to send an application notification email to advisor’s email account.
* The system should take no more than one second to pop up the PDF application after advisor clicks on any one application item in the application table.

## Safety Requirements

* The system design and location should adhere to any safety regulations specified by civil engineering standards, hardware engineering standards, and Information Technology security standards
* If power is lost to the KSU, the system should opt to shutdown the server properly for the sake of not losing data
* The server room should not be accessible to those who do not need to be within the room, meaning anyone who is not an IT (Information Technology) professional should find a locked room that they cannot enter without force
* The system should adequately transfer and store those old application information to an off-site data warehouse if applicable when new semester is coming.

## Security Requirements

* The system should follow security standards of the computing industry.
* Because the system’s purpose is serving KSU students, so people who are not KSU students
* shouldn’t have access to the system. If is possible, we plan to put the system into KSU Owl Express so that other people cannot log in.
* After student submit his/her substitution application, the product should guarantee that no one could make modification on the original application form, even advisor user or department chair user. User other than student could only write comments or make judgement on the application, but never change its contents.

## Software Quality Attributes

* Reliability: If the connection between the student user and the system is broken prior to an substitution application being submitted or canceled, the system should enable the student user to recover from the incompleted application. Same things when advisor is making comments.
* Availability: the system only opens to KSU students, advisor, program coordinator and department chair. No other people have access. This is realized by using login page. And the system should be available and functional 24/7 and 365 days a year expect in the case of scheduled shutdown and maintenance period.
* Portability: The database (excel file which stores all the application) will be built for a system and it can be portable so the results of the data will be portable between many environments. And the system should be able to access through various Internet browsers such as Google Chrome, MS Edge, FireFox and Safari.
* Reusability: The system shall be able to update the database for use in each new semester to come.
* Robustness: the system should have the robustness against invalid input. In the login page, if user input some information (wrong KSU ID or wrong name) which doesn’t exist, the system should be able to output the error notification and back to the login page again. After login, dropdown list will be used as many as possible to avoid invalid input.

## Business Rules

* Student can login in the system and fill in a form, then submit or cancel the substitution application.
* From advisor angle, there is a file storage table of all the applications in the system, advisor could click on any of these application to see the detail content of the application. And advisor could add comments and forward the application to program coordinator/department chair or deny.
* Program coordinator could forward the application to department chair or deny.
* Department chair reserves the final right to approve or deny, with electrical signature.