



NYU

**TANDON SCHOOL
OF ENGINEERING**

Computer Science and Engineering

Sharit

System Analysis Specification

Version 1.0

Document Number: SAS-001

Team A6

Allen Zheng (az1010)

Hui Huang (hh1128)

Kenneth Liang (kl1792)

Warlon Zeng (wz634)

REVIEW AND APPROVALS

Printed Name and Title	Function (Author, Reviewer, Approval)	Date	Signature
Allen Zheng	Author	April 18, 2016	Allen Zheng
Hui Huang	Author	April 18, 2016	Hui Huang
Kenneth Liang	Author	April 18, 2016	Kenneth Liang
Warlon Zeng	Author	April 18, 2016	Warlon Zeng

REVISION LEVEL

Date	Revision Number	Purpose
April 18, 2016	Version 1.0	Initial Release

Table of Contents

1. INTRODUCTION	5
1.1 PURPOSE	
2. SCOPE	5
2.1 IDENTIFICATION	
2.2 BOUNDS	
2.3 OBJECTIVES	
2.4 SYSTEM OVERVIEW	
2.5 DOCUMENT OVERVIEW	
3. REFERENCE DOCUMENTS	7
4. BUSINESS REQUIREMENTS	7
4.1 TECHNOLOGY	
4.2 ECONOMICS	
4.3 REGULATORY AND LEGAL	
4.4 MARKET CONSIDERATIONS	
4.5 RISKS AND ALTERNATIVES	
4.6 HUMAN RESOURCES AND TRAINING	
5. LOGICAL ARCHITECTURAL SPECIFICATION	9
5.1 CONTEXT DIAGRAM	
5.2 SYSTEM CAPABILITY REQUIREMENTS	
5.2.1 CAPABILITIES	
5.2.2 USE CASE DIAGRAMS	
5.2.3 USE CASE DESCRIPTIONS	
5.3 USER INTERFACE REQUIREMENTS	
5.4 COMPONENT (COMPONENT/PACKAGE/SUBSYSTEM) ARCHITECTURE	
5.4.1 COMPONENT DESCRIPTIONS	
5.4.2 COMPONENT ARCHITECTURE DIAGRAM	
5.5 CLASS DIAGRAMS	
5.6 CLASS RELATIONSHIP/INTERACTION DIAGRAMS	
5.7 EVENTS	
5.7.1 MOTIVES	
5.7.2 EVENT DIAGRAMS	
5.8 ACTIVITY/STATE (SCENARIO) SECTION	
5.9 STATE LOGIC	

5.10	<u>BEHAVIOR</u>	
5.10.1	<u>SEQUENCE DIAGRAMS</u>	
5.10.2	<u>COLLABORATION DIAGRAMS</u>	
5.11	<u>DICTIONARIES</u>	
6.	<u>NON-FUNCTIONAL/OPERATIONAL SPECIFICATIONS</u>	28
6.1	<u>SYSTEM EXTERNAL INTERFACE REQUIREMENTS</u>	
6.2	<u>SAFETY REQUIREMENTS</u>	
	<u>Access to the system is maintained by login. Files are also managed by the user who uploaded it. This permission can be changed only by the user and may provide difficulties in sharing to a large group of people.</u>	
6.3	<u>SECURITY AND PRIVACY REQUIREMENTS</u>	
6.4	<u>SYSTEM ENVIRONMENT REQUIREMENTS</u>	
6.5	<u>COMPUTER RESOURCE REQUIREMENTS</u>	
6.5.1	<u>COMPUTER HARDWARE REQUIREMENTS</u>	
6.5.2	<u>COMPUTER HARDWARE RESOURCE REQUIREMENTS</u>	
6.5.3	<u>COMPUTER SOFTWARE REQUIREMENTS</u>	
6.5.4	<u>COMPUTER COMMUNICATIONS REQUIREMENTS</u>	
6.6	<u>SYSTEM QUALITY FACTORS</u>	
6.7	<u>DESIGN AND CONSTRUCTION CONSTRAINTS</u>	
6.8	<u>PERSONNEL-RELATED REQUIREMENTS</u>	
6.9	<u>TRAINING-RELATED REQUIREMENTS</u>	
6.10	<u>LOGISTICS-RELATED REQUIREMENTS</u>	
6.11	<u>PACKAGING REQUIREMENTS</u>	
6.12	<u>PRECEDENCE AND CRITICALITY REQUIREMENTS</u>	
6.13	<u>OTHER REQUIREMENTS</u>	
7.	<u>SYSTEM TEST PLAN REQUIREMENTS</u>	31
8.	<u>QUALIFICATION PROVISIONS</u>	33
9.	<u>REQUIREMENTS TRACEABILITY</u>	33
10.	<u>RATIONALE</u>	34
11.	<u>NOTES</u>	34
12.	<u>APPENDICES</u>	34
12.1	<u>DICTIONARIES</u>	
12.2	<u>UML DIAGRAMS</u>	
12.3	<u>SCHEDULE TRACKING</u>	
12.4	<u>DEFECT TRACKING</u>	
12.5	<u>Gantt Chart/Microsoft Project Schedule</u>	

1. INTRODUCTION

1.1 PURPOSE

The purpose of this document is to specify the structure and functionality for the website Sharit. This document will outline details, such as system requirements and testing procedures. This document is intended for the whole development team which includes developers, managers, and architectural designers.

2. SCOPE

2.1 IDENTIFICATION

This document is the System Analysis Specification for Sharit, SAS-001. This is version 1.0, revision 1, and is completed on April 18, 2016.

2.2 BOUNDS

The client's idea stems from the need to share files/documents in an organized, coherent manner that is also accessible to many others who you want to share with. There are services out there that do this job but is disorganized and does not provide any kind of feedback system.

Students can use this service to post notes for the underclassmen and receive feedback on their work. Students will be able to collaborate and create notebooks that can be shared with other students in the class. This will benefit those who may not have been able to attend school for that day or those who want to use this platform to study for an upcoming exam. The key to our platform is organization of uploaded files; we want people to be able to look up the file they want and quickly access it or ask the owner of the file to grant permission to access.

The project is a website that facilitates conversations and file sharing between people from a common organization. The layout will be a clean and simple format for

easy intuitive understanding. Organizations will have a domain that will be split into smaller pieces, such as the subdomain and topics. Users will have permissions regarding which organization domain and subsequent subdomains they may access.

2.3 OBJECTIVES

The priority is to release the Software Design Document by next semester. The project will be utilizing an iterative and incremental life cycle with the deliverables listed below:

Software Requirements Specifications	March 21, 2016
Software Project Management Plan	April 6, 2016
Software Analysis Specification	April 18, 2016
Software Design Document	Postponed to next semester

2.4 SYSTEM OVERVIEW

The project is to establish software that can be used as a means of communication in an organization. The structure will be similar to that of the social site “reddit.” An organization will have its own domain on the server. User will belong to an organization and can only access their organization’s **domain**. Users may belong to several organizations. In each domain there will be several **subdomains** that usually correspond to different parts of the organization. Users can create topics in each section and comment on topics. Files can be attached to and downloaded from topics or comments. Some users will be given special privileges in a subdomain, allowing them to delete invalid threads or comments. These users will be known as **moderators**. A user can be the moderator of many subdomains. The domain itself will have moderators, allowing those users to create and delete subdomains. By being a domain moderator, that user will also be a moderator of all subdomains.

A website will be created to allow users access to their domains. The website will feature a simple and clean interface. For this project, the website will feature logging in, logging out, and access for authenticated domains, subdomains, and topics. There will also be a user account page in case the user wishes to change his information.

2.5 DOCUMENT OVERVIEW

This document has the analysis specifications for the software product Sharit. The team plans on implementing the system using an object oriented paradigm and this document will provide a simple analysis of the methodology. More detail will be available in the Software Design Document. This document also contains updated models from the previous two documents (SRS and SPMP) which includes, logical architectural specification, business requirements, and operational requirements.

3. REFERENCE DOCUMENTS

Project Sharit Team A6 Project Proposal, Version 1.0, March 7, 2016.

Project Sharit SRS-002, Version 2.0, March 23, 2016.

Project Sharit SPMP-001, Version 1.0, April 6, 2016.

4. BUSINESS REQUIREMENTS

4.1 TECHNOLOGY

To deliver fast, reliable, and high quality website service, recent dated versions of software in servers, databases, and programming languages should be used. These software, in combination with development tools, such as git, google cloud, or any other preference tool, will help build a website to meet users' expectations.

4.2 ECONOMICS

1. Domains will be sold to organizations.
2. Revenue comes from selling disk space to organizations

4.3 REGULATORY AND LEGAL

1. No regulatory or legal requirements at this time.
-

4.4 MARKET CONSIDERATIONS

Since registration is entirely dependent on having a @nyu.edu email address, the targeted group of consumers will be university students. To the students, the service will be free of charge. The project will proceed to follow a subscription model, in which the school will pay for users' subscription, and will appear free to users.

4.5 RISKS AND ALTERNATIVES

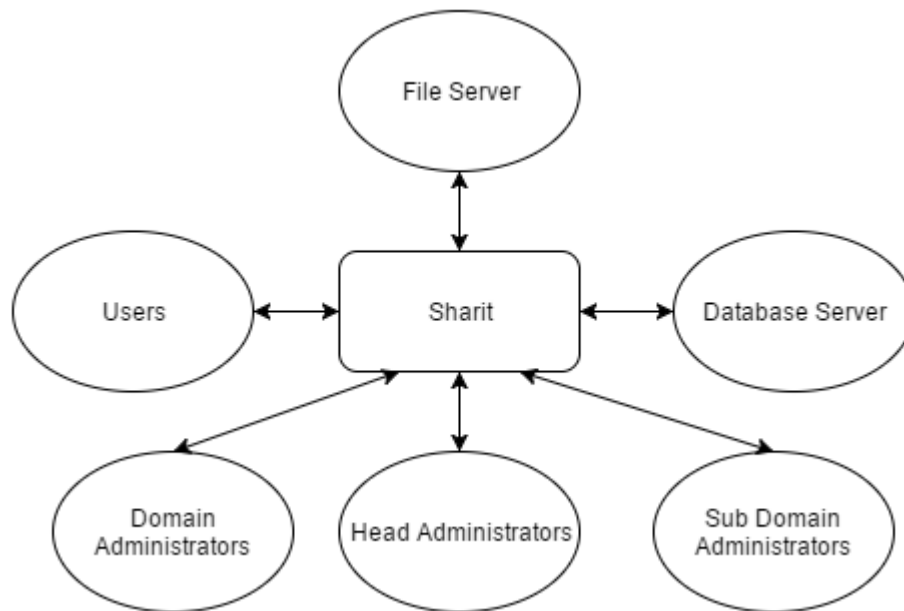
1. Not enough users signing up
 - 1.1. Can potentially expand to more schools outside of NYU
2. Not enough content being produced
 - 2.1. Can offer monetary incentives for students to contribute as a start-up and later retract offers after a self-sufficient number of contributions generated
3. Work that are not acceptable to be shared
 - 3.1. Can be removed by moderators

4.6 HUMAN RESOURCES AND TRAINING

1. Student programmers
2. Full stack training
 - 2.1. HTML
 - 2.2. CSS
 - 2.3. Javascript
 - 2.4. PHP/Python - general purpose language
 - 2.5. postgres/oracle - database system
 - 2.6. C++ - web server
 - 2.7. Windows - operating system
 - 2.8. Git - version control

5. LOGICAL ARCHITECTURAL SPECIFICATION

5.1 CONTEXT DIAGRAM



5.2 SYSTEM CAPABILITY REQUIREMENTS

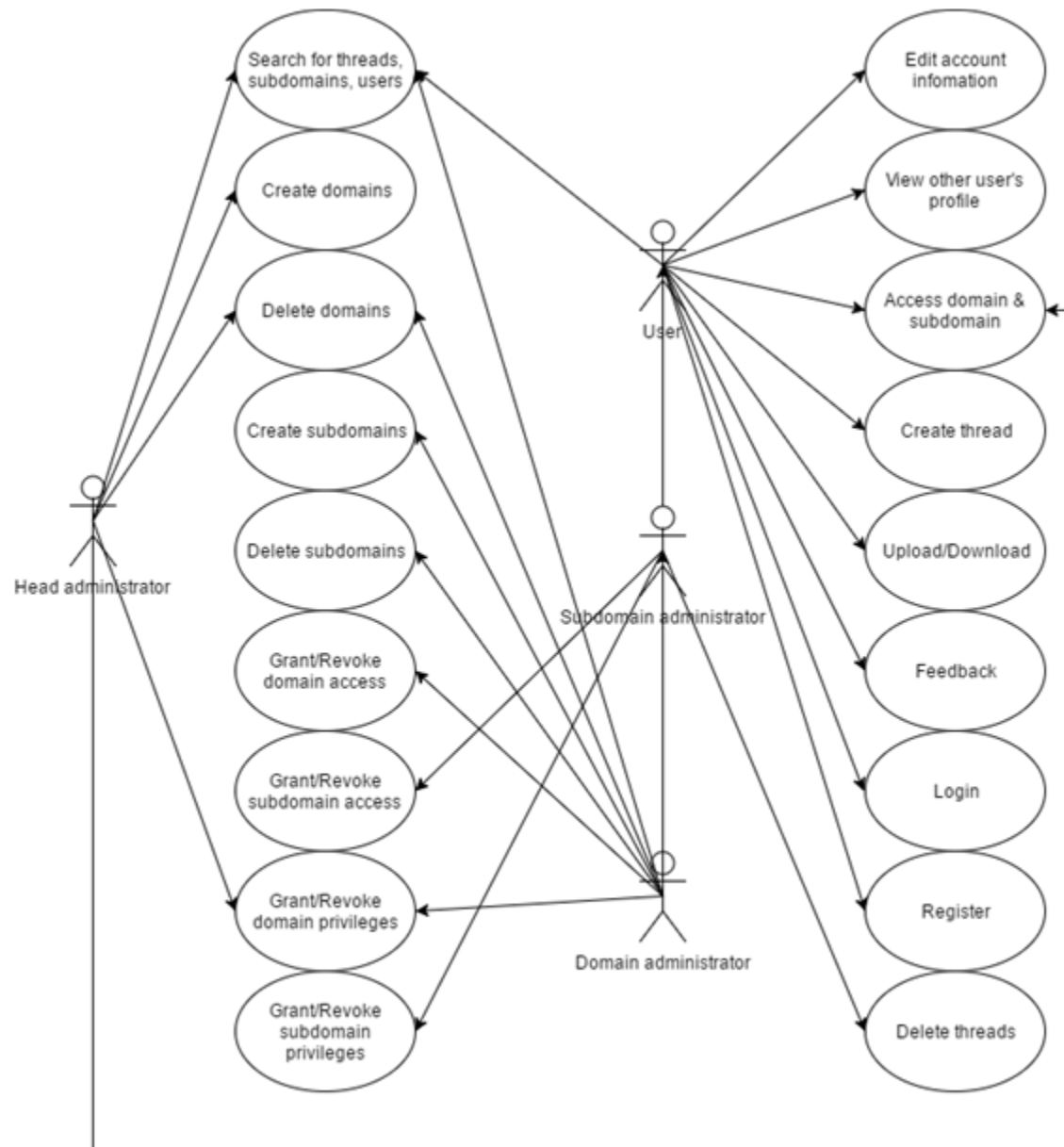
5.2.1 CAPABILITIES

Each of the use cases correspond to each bubble (or combination of bubbles) in the use case diagram in section 5.2.2.

Use Case	Description
Register	A new user can register for Sharit
Login	A registered user can login by providing a valid username and password
Edit Information	A user can edit basic account information such as password, school, company, and profile.

View other user's profile	A user can view the basic information of another user such as comments, thread posts, and upvotes.
Access domain and subdomain	An authorized user can access a domain or subdomain, they can create threads, leave comments, and upvote.
Create thread	An authorized user of a domain can create threads within the domain.
Upload/Download	Users can upload/download files from a thread.
Feedback	The user can leave a comment or upvote/downvote in a thread.
Search for threads, subdomains, users	Head and domain administrators, as well as users, can search for existing threads, subdomains, and/or users
Grant/revoke domains, subdomains access/privilege	Administrators will be able to grant or revoke permission/access to users or other administrators
Create domains, sub-domains	Administrators can create domains and subdomains.
Delete domains, subdomains, threads	Administrators can delete domains, subdomains, and threads.

5.2.2 USE CASE DIAGRAMS



5.2.3 USE CASE DESCRIPTIONS

Register		
Description	Newcomers could register for an account on the website.	
Pre-Conditions	Registering for an account requires a @nyu.edu email address.	
Flows	Basic or Normal Flows	1. Registrants registers for an account using @nyu.edu email address. 2. After registering, the server will send an email confirmation (click link to verify) to the registered email address. 3. Email address will be verified by server and registration complete.
	Alternative Flows	1. Registrant attempts to register with an invalid email address - not @nyu.edu 2. Registrant account creation denied
Post Conditions	Registrant becomes registered, functionality of the website is granted.	
Special Requirements	Email address used to register must be @nyu.edu.	
Extension Points	Invalid email addresses will result in rejection to account creation.	

Login		
Description	A user who has previously registered will be able to log in using the registered username and password.	
Pre-Conditions	The user must have been previously registered.	
Flows	Basic or Normal Flows	1. User entered login information. 2. Website validates with database and the information is in the system. 3. The user is redirected to the homepage.
	Alternative Flows	1. User enters invalid credentials. 2. User is prompted to enter again. 3. Process repeats until valid credentials have been provided.

Post Conditions	The user is logged in and can use the website and its functionalities.
Special Requirements	The system will create a session to store the user's information.
Extension Points	<ol style="list-style-type: none"> 1. User enters invalid credentials <ul style="list-style-type: none"> - Prompt user to reenter credentials 2. User forgot the login information <ul style="list-style-type: none"> - Has option to recover via email

Edit account information		
Description	Users can edit their account information	
Pre-Conditions	Must login to edit account information	
Flows	Basic or Normal Flows	<ol style="list-style-type: none"> 1. User specifies which fields they desire to edit. 2. User then edit desired fields. 3. User then saves the information and server will acknowledge saved changes.
	Alternative Flows	<ol style="list-style-type: none"> 1. User specifies which fields they desire to edit. 2. User provides invalid information or information that the server does not know how to process. 3. User edited fields will not be saved and error messages will be displayed.
Post Conditions	User account information is updated with saved changes.	
Special Requirements	All updated changes will be saved into the database.	
Extension Points	If a user tries to enter invalid information, such as wrong formatting, error message(s) pertaining to incorrect fields will be displayed.	

View other users' profile	
Description	A user can view the information of other users
Pre-Conditions	The user must be logged in and within the same domain

Flows	Basic or Normal Flows	<ol style="list-style-type: none"> 1. User is redirected to the other user's profile page 2. User is able to view basic information like name, school, and threads 3. User can see account activity
	Alternative Flows	There are no alternate flows.
Post Conditions	The user is able to view another user's basic information and account activity which includes: comments, threads and upvotes	
Special Requirements	The user must be logged in and within the same domain as the user they are trying to view.	
Extension Points	If the user tries to view the profile of another user who has deleted their account, an error message will be displayed.	

Access domain and subdomain		
Description	Head administrators and users can access domains that expand to further subdomains and threads.	
Pre-Conditions	The groups must be logged in and authorized.	
Flows	Basic or Normal Flows	<ol style="list-style-type: none"> 1. User chooses which domain to enter 2. User enters domain 3. User can see subdomains and, if have access to, its threads
	Alternative Flows	<ol style="list-style-type: none"> 1. User does not have access to domain 2. User enters a domain they don't have access to 3. Access denied
Post Conditions	User can view accessed subdomains and threads of entered domain.	
Special Requirements	User must have access to enter desired domain and subdomain	
Extension Points	If user tries to enter a domain they do not have permission to, they will not see its domain contents: subdomains and its threads.	

Create thread		
Description	Users can create threads within subdomains.	
Pre-Conditions	The user must be authorized within the subdomain and logged in.	
Flows	Basic or Normal Flows	1. User is logged in and is authorized in the subdomain that they want to post to. 2. User browses to the subdomain 3. User posts a thread
	Alternative Flows	1. User is logged in but is not authorized in the subdomain. 2. User browses to the subdomain. 3. User cannot post a thread.
Post Conditions	There is now an option to upload a file into the subdomain and leave comments.	
Special Requirements	User must be logged in and received permission from the administrator.	
Extension Points	If the user does not have authorization, they will not see the create thread button or be able to view any threads in the subdomain.	

Upload/Download		
Description	Users can upload and/or download file(s) to thread	
Pre-Conditions	The user must be authorized within the subdomain and logged in.	
Flows	Basic or Normal Flows	1. User finds which thread to upload/download to/from 2. User uploads/downloads to/from a thread (comment-link) 3. User can post new thread (thread-link)
	Alternative Flows	1. User uploads something inappropriate 2. File(s) uploaded 3. File(s) deleted by moderator
Post Conditions	User will successfully download file from thread or upload file(s) to thread (comment-link) or make a new thread (thread-link).	
Special Requirements	User uploaded file(s) will be saved to database.	

Extension Points	If user tries to upload something inappropriate, the file(s) will be uploaded, but will be quickly brought down by moderators.
-------------------------	--

Feedback		
Description	The user can leave a comment or upvote/downvote in a thread.	
Pre-Conditions	The user must be logged in and authorized within the subdomain.	
Flows	Basic or Normal Flows	<ol style="list-style-type: none"> 1. User navigates to a subdomain. 2. User finds a thread that they want to leave feedback for. 3. User leaves feedback.
	Alternative Flows	<ol style="list-style-type: none"> 1. User navigates to a subdomain. 2. User does not have authorization within the subdomain. 3. User is unable to see threads or leave feedback.
Post Conditions	The feedback that the user leaves will be visible to other users.	
Special Requirements	User must be authorized within the subdomain	
Extension Points	There will be a character limit on the comments that the user leaves.	

Search for threads, subdomains, users		
Description	Head and domain administrators, as well as users, can search for existing threads, subdomains, and/or users	
Pre-Conditions	Must be logged in and have authorization.	
Flows	Basic or Normal Flows	<ol style="list-style-type: none"> 1. Specify which thread, subdomain, and/or user to search for 2. Searches for specified item 3. Item found and displayed
	Alternative Flows	<ol style="list-style-type: none"> 1. User chooses which thread/subdomain/user to search for 2. Specified item does not exist

	3. Error message displayed
Post Conditions	Request of item will be delivered if it exists and request party has access to.
Special Requirements	In order to use the search functionality, head and domain administrator, as well as user, will require access beforehand.
Extension Points	If search functionality is used without proper access rights, error message will be displayed

Grant/revoke domains, subdomains access/privilege		
Description	Administrators will be able to grant or revoke permission/access to users or other administrators	
Pre-Conditions	The administrator must be registered in the database	
Flows	Basic or Normal Flows	1. User requests privilege from administrators. 2. Administrator receives the request. 3. Administrator grants request.
	Alternative Flows	1. User requests privilege from administrators. 2. Administrator receives the request. 3. Administrator denies request.
Post Conditions	The user or other administrators now have privilege/access to a domain or subdomain.	
Special Requirements	Authorization must be granted from a current administrator.	
Extension Points	1. Access to a subdomain grants access to all threads in that sub-domain 2. Privilege to a domain or subdomain means you are the administrator	

Create domains, sub-domains	
Description	Administrators can create domains and subdomains.
Pre-Conditions	Administrators must be logged in.

Flows	Basic or Normal Flows	1. Administrator logs in. 2. Administrator creates a domain. 3. Domain is now visible.
	Alternative Flows	1. Administrator logs in. 2. Administrator creates a subdomain. 3. Subdomain is now visible.
Post Conditions	The domain or subdomain can be accessed by those who have permission,	
Special Requirements	Only head administrator can create domains. Domain administrator can create subdomains.	
Extension Points	Head administrator has the most power, followed by domain administrator, followed by subdomain administrator.	

Delete domains, subdomains, threads		
Description	Administrators can delete domains, subdomains, and threads.	
Pre-Conditions	Administrators must be logged in.	
Flows	Basic or Normal Flows	1. Administrator logs in 2. Administrator deletes domain
	Alternative Flows	1. Administrator logs in 2. Administrator deletes subdomain
		1. Administrator logs in 2. Administrator deletes a thread
Post Conditions	The domain/subdomain/thread is deleted.	
Special Requirements	Only head administrators and domain administrators can delete domains. Only domain administrators and subdomain administrators can delete subdomains.	
Extension Points	Deleting a domain deletes the domain and all subdomains within. Deleting a subdomain deletes the subdomain and all threads within.	

5.3 USER INTERFACE REQUIREMENTS

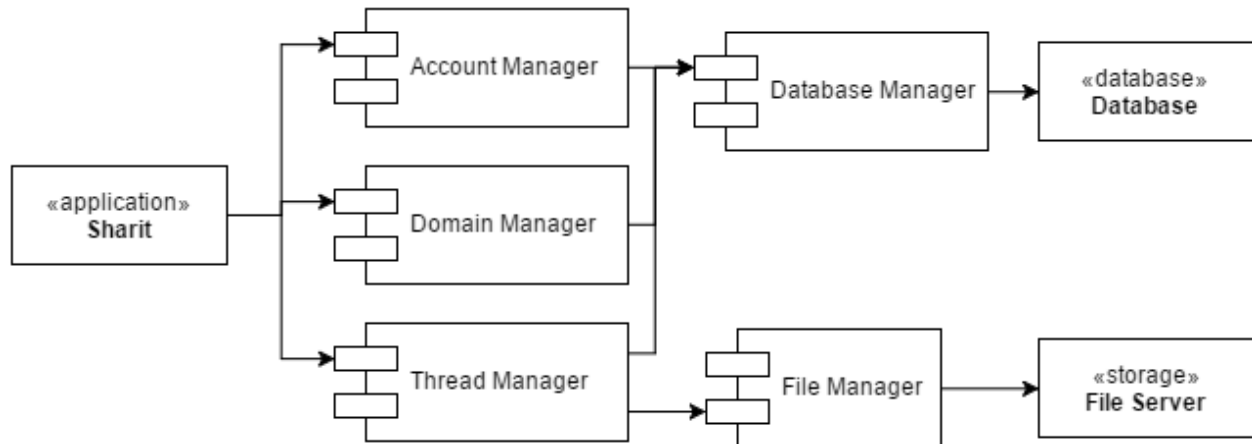
The user interface will be simple and clean. The goal is to produce a fairly intuitive design that meets the client's needs. Additional details will not be provided, since a prototype is not necessary.

5.4 COMPONENT (COMPONENT/PACKAGE/SUBSYSTEM) ARCHITECTURE

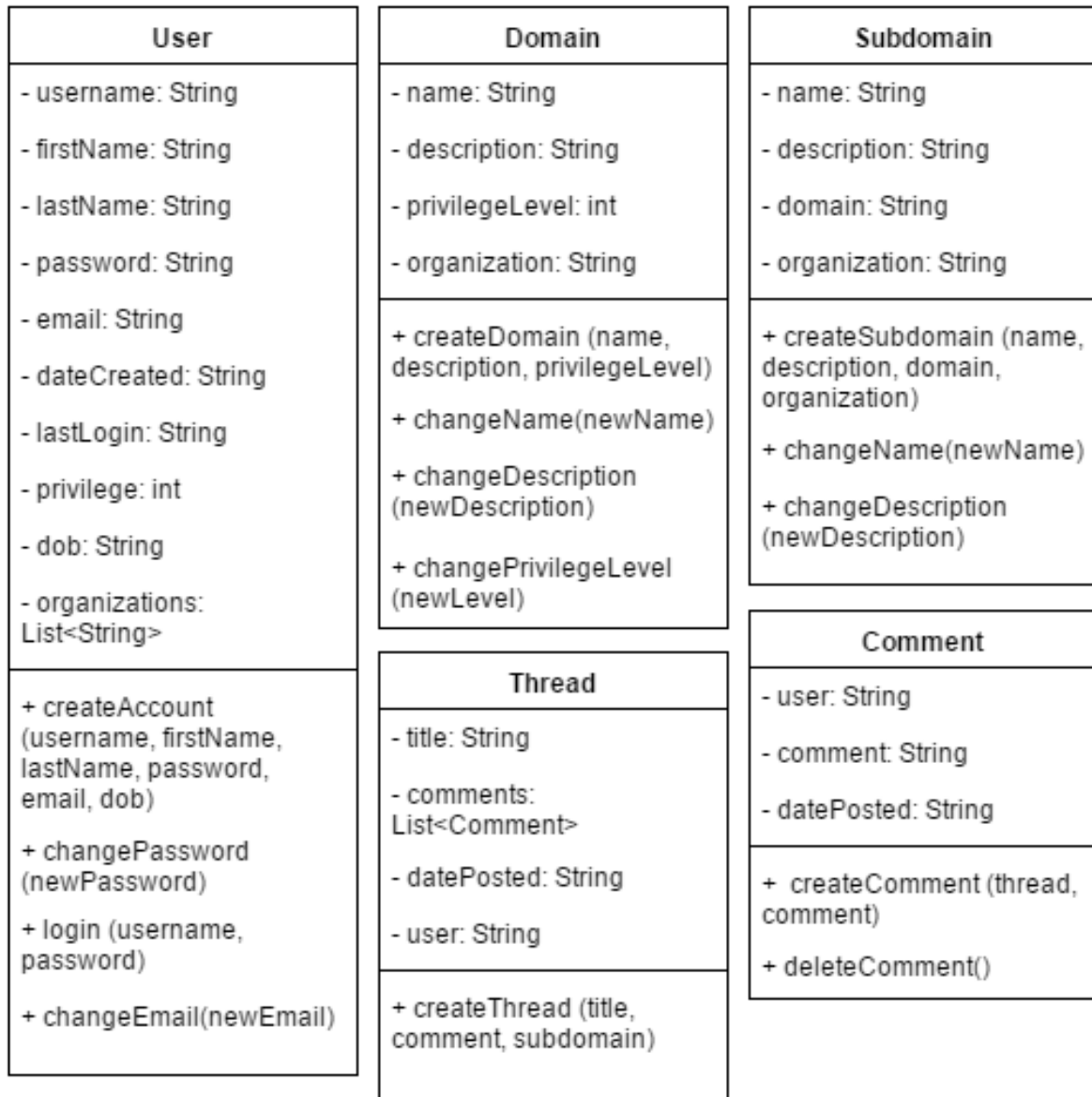
5.4.1 COMPONENT DESCRIPTIONS

Component	Description
Account Manager	Handles user login and various other account activities, such as change password and update email address.
Domain Manager	Handles user access in domains, rejecting them if the user has insufficient privilege.
Thread Manager	Handles viewing threads, comments, and posting comments.
File Manager	Handles uploading files to the file server and downloading files from the server.
Database Manager	Handles access to the database server for information.

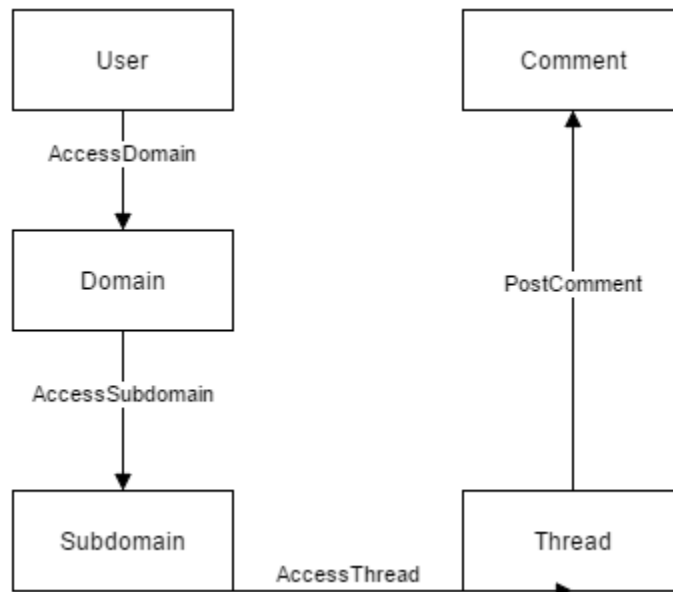
5.4.2 COMPONENT ARCHITECTURE DIAGRAM



5.5 CLASS DIAGRAMS



5.6 CLASS RELATIONSHIP/INTERACTION DIAGRAMS



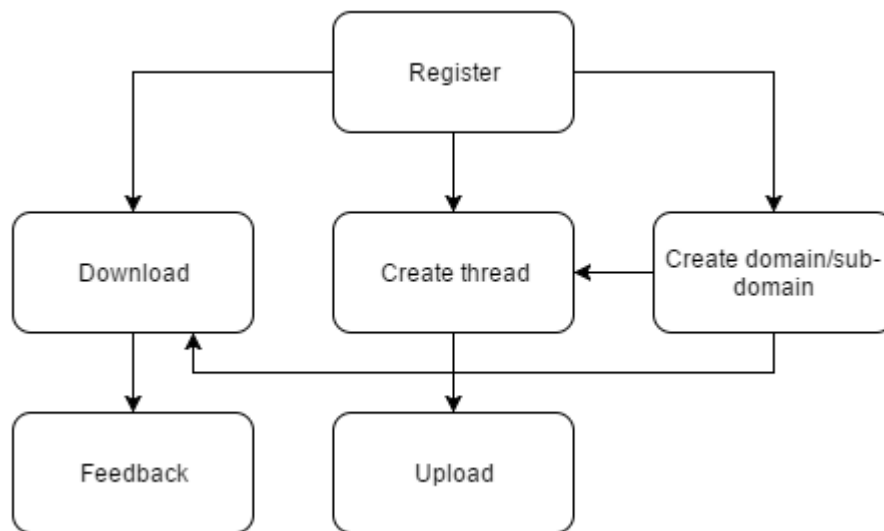
5.7 EVENTS

5.7.1 MOTIVES

Motive	Description	Objective
Register	Allow user to create account	Grant access to Sharit
Upload file	User uploads a file	Share files so others in domain can view
Download file	User downloads a file	Gain resource that was shared by others
Create thread	User creates a thread in a domain	A thread allows users to interact with others
Create domain/sub-domain	Authorized user can create a subdomain	Compartmentalize domains into more specific groups

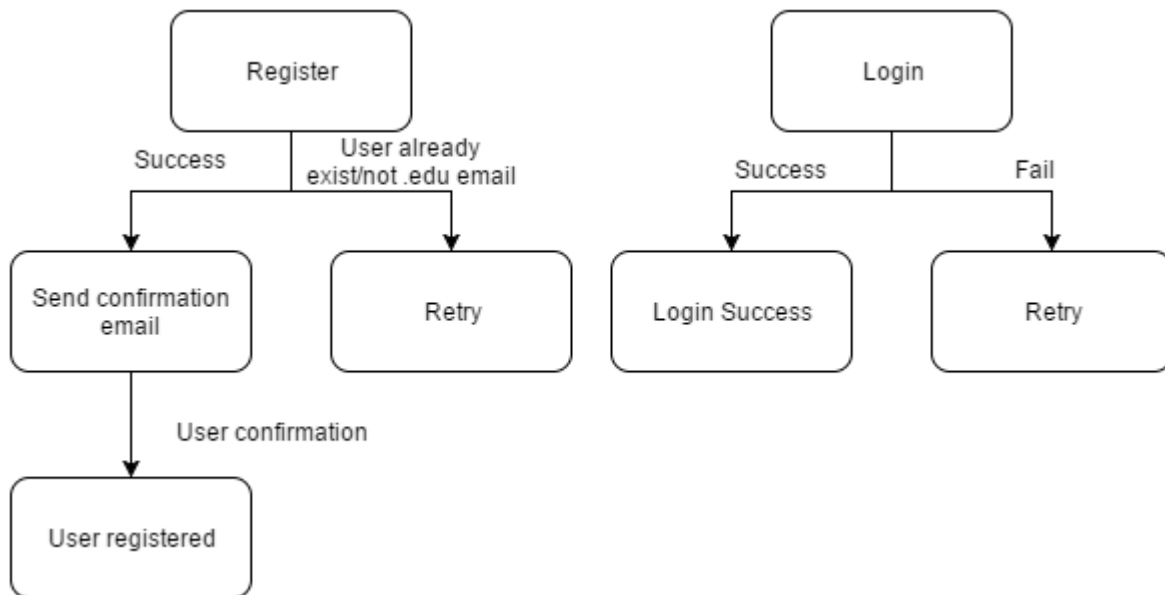
Feedback	User can comment, upvote/downvote	This allows users to engage with each other
----------	-----------------------------------	---

5.7.2 EVENT DIAGRAMS

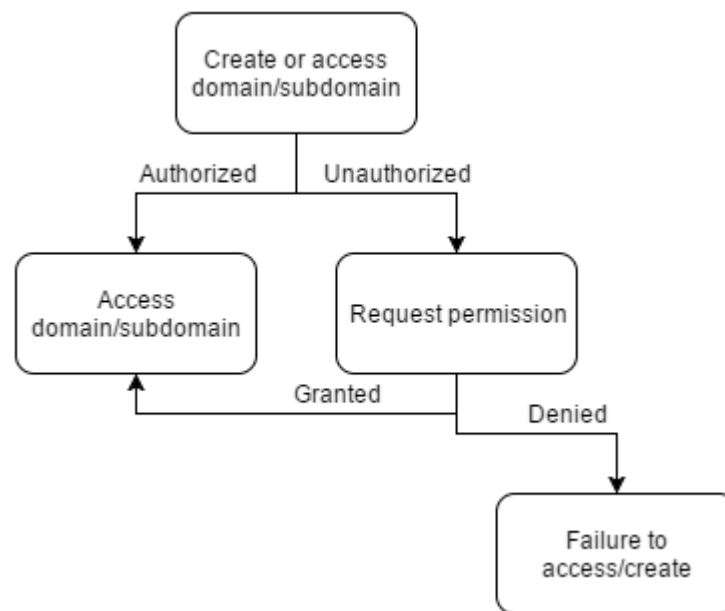


5.8 ACTIVITY/STATE (SCENARIO) SECTION

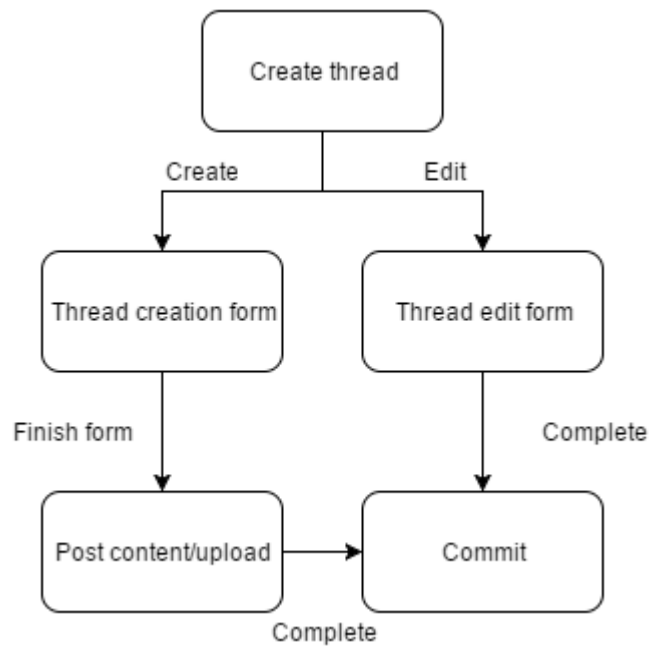
The user tries to login or register into Sharit.



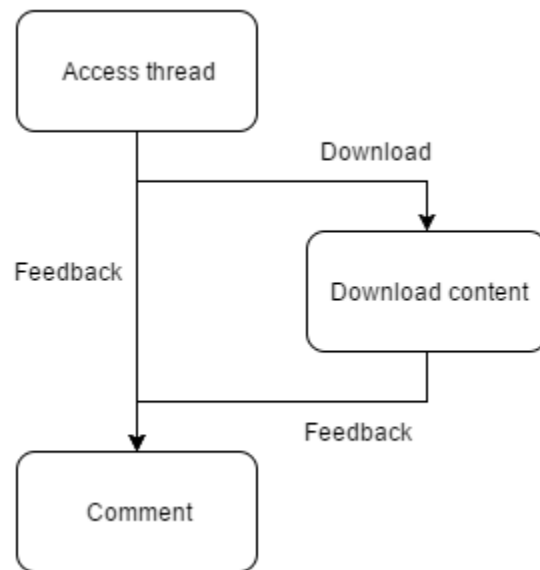
The user tries to access or create a domain or a subdomain.



The user tries to create a thread after gaining access to a domain or subdomain.



The user wants to download content from a thread or leave feedback

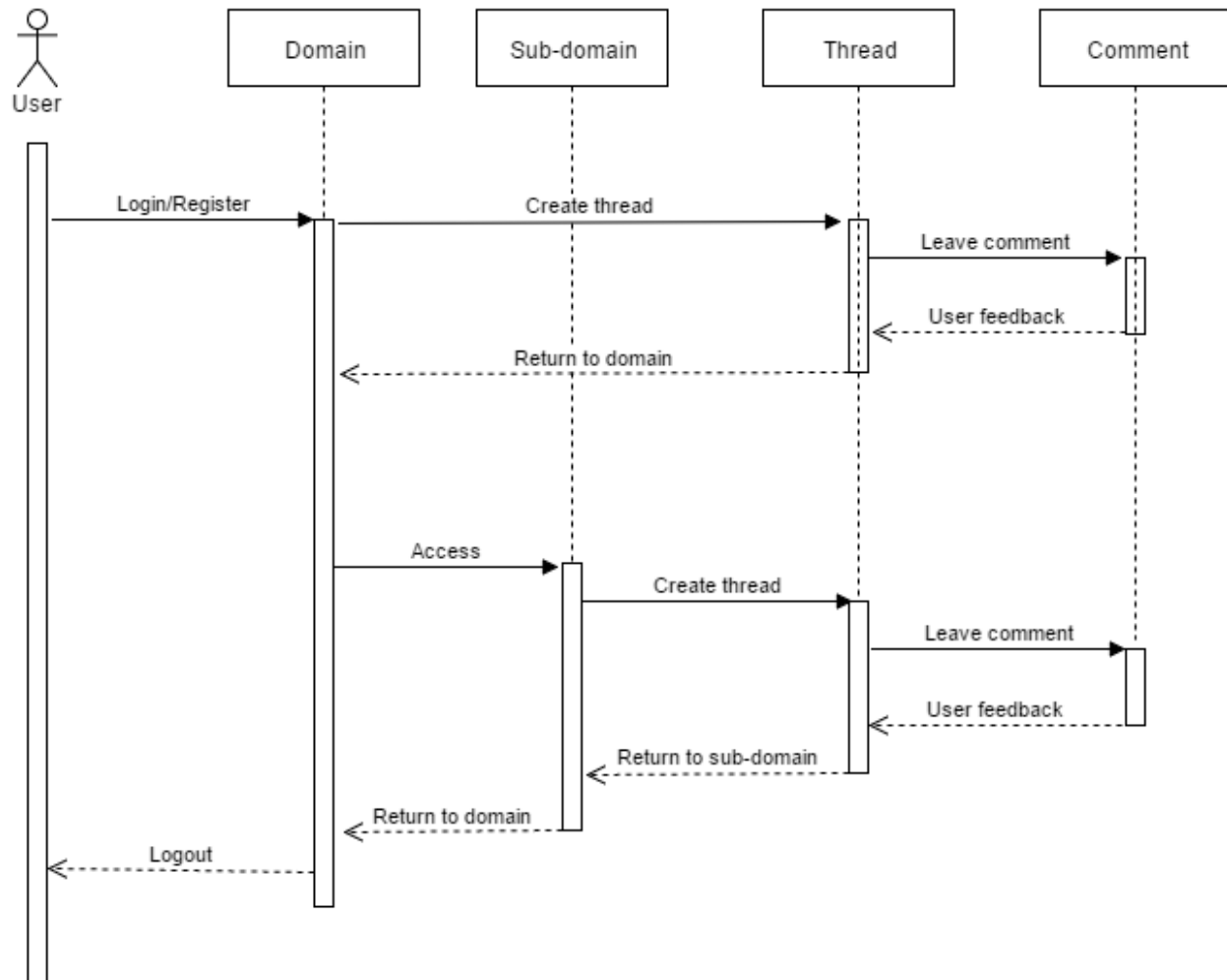


5.9 STATE LOGIC

The state logic will be postponed to the Software Design Document.

5.10 BEHAVIOR

5.10.1 SEQUENCE DIAGRAMS



5.10.2 COLLABORATION DIAGRAMS

The collaboration diagrams will be completed in the software design document.

5.11 DICTIONARIES

See appendix 12.1 for the relevant dictionary information.

6. NON-FUNCTIONAL/OPERATIONAL SPECIFICATIONS

6.1 SYSTEM EXTERNAL INTERFACE REQUIREMENTS

Sharit will be using a third party server to host the whole system. A project manager will be in charge of all operations of the software product. It will also use a SQA group to ensure that there are no defects/faults in the system once the system has been completed.

6.2 SAFETY REQUIREMENTS

Access to the system is maintained by login. Files are also managed by the user who uploaded it. This permission can be changed only by the user and may provide difficulties in sharing to a large group of people.

6.3 SECURITY AND PRIVACY REQUIREMENTS

Each user has a login in the form of email and password. Email should be school username as clients are students in the school. Passwords will have to follow three basic requirements. They must be at least 8 characters, must be a mix of numbers and letters, and contain one symbol. Forgotten passwords can be reset through an email link when requested by the user.

All files on the servers have privacy features. The user that uploaded the file has the ability to select permissions and access given to all users within the system.

6.4 SYSTEM ENVIRONMENT REQUIREMENTS

System should be able to run in any environment. Users can use it as long as they have access to the internet. The system can be access on phones and/or computers that meet the minimum requirements.

6.5 COMPUTER RESOURCE REQUIREMENTS

6.5.1 COMPUTER HARDWARE REQUIREMENTS

	Minimum requirements
Processor	Intel Pentium® D 2.8 GHz or AMD Athlon™ 64 X2 4400+
Video	Intel Q35 Express or Radeon HD 2400 PCI
Memory	512 MB RAM
Resolution	1280 × 720
Internet	1 Mbps

6.5.2 COMPUTER HARDWARE RESOURCE REQUIREMENTS

I/O Devices	QWERTY keyboard Mouse Hard drive with sufficient free space
-------------	---

6.5.3 COMPUTER SOFTWARE REQUIREMENTS

Operating System	Windows® XP/Vista/7/8/8.1/10 OS X Chrome OS Linux
Browser	Google Chrome, Firefox, Safari, Opera, Internet Explorer (Latest version recommended)

6.5.4 COMPUTER COMMUNICATIONS REQUIREMENTS

Communication between users will occur through comments on a specific thread. The only communications requirement will therefore be the minimum computer requirements to access the website and a keyboard.

6.6 SYSTEM QUALITY FACTORS

System will be checked by all coders after each iteration as well as after each implementation of features. This will ensure that the quality of the code is maintained properly. After each iteration there will also be a walkthrough and inspection.

6.7 DESIGN AND CONSTRUCTION CONSTRAINTS

The project uses the iterative and incremental development model to deliver the software products. We chose this model due to its reflex nature and constant ability to revisit the five workflows: requirements, analysis, design, implementation, and test.

During the implementation phase for each iteration, Git will be used as the version control software. This will help alleviate problems such as different members working on different versions of the software, and allows for backtracking when things goes wrong. GitHub will store all the files while Git controls the version that controls the version of the project.

6.8 PERSONNEL-RELATED REQUIREMENTS

There will personnel with administrator privileges who will manage the database, file server, and web server. There will be a minimum of three personnel, but as the database and file server grow, there will be an increase in personnel to assist in maintenance.

6.9 TRAINING-RELATED REQUIREMENTS

Users get a simple tutorial that allows them to understand how to upload, download, and change permissions.

6.10 LOGISTICS-RELATED REQUIREMENTS

Server will be maintained on a monthly basis. This will ensure that the file server has enough space for new files as well as add new features or fix any problems. Staff will have to log changes made to the system. Changes will be alerted to users via patch notes.

6.11 PACKAGING REQUIREMENTS

There are no packaging requirements needed to use Sharit.

6.12 PRECEDENCE AND CRITICALITY REQUIREMENTS

The requirement with the most precedence is security and privacy for our users. Every other requirement has the same precedence and criticality.

6.13 OTHER REQUIREMENTS

The system will require upgrades in the future to ensure space on file server can handle all files in system. The database will also be upgraded to accommodate new users.

7. SYSTEM TEST PLAN REQUIREMENTS

This test plan will assess the requirements as specific in section 7.1 of the SRS. The following test scenario will demonstrate how a user might use Sharit.

Spec. #	Action	Input	Expected Output
7.1.1.1	Sign up for account	smith@nyu.edu	Account created
7.1.1.1	Sign up for account	jane@gmail.com	Fail
7.1.1.2	Login to account	smith@nyu.edu	Login Successful
7.1.1.2	Login to account	foo@nyu.edu	Fail

7.1.1.3	View profile	Smith's profile	Smith's information(name, company/school)
7.1.1.4	Change information	John	Smith's name is now John
7.1.2.1	Access domain	Smith's domain	Success
7.1.2.1	Access domain	Jane's domain	Fail
7.1.2.2	Check administrator	Smith	True
7.1.2.3	Delete thread	Ice skating	Thread deleted
7.1.2.3	Grant permission	Jane	Permission granted
7.1.2.4	Create subdomain	Tests	Subdomain successfully created
7.1.3.1	Create thread	Prof. Bob's Fall 2015 midterm	Thread successfully created
7.1.3.3	Upload file	fall_2015_midterm.pdf	File upload success
7.1.3.3	Download file	fall_2015_midterm.pdf	Download success
7.1.4.1	Upvote/downvote thread	-1	Vote success
7.1.4.2	Comment on thread	"Useful sample"	Comment posted
7.1.4.3	Comment on comment	"Incorrect info"	Comment replied
7.1.4.4	Upvote/downvote comment	+1	Comment upvoted/downvoted
7.1.5.1	Search file/thread	fall_2015_midterm	File/thread found
7.1.5.2	Search user	John	User profile found
7.1.5.3	Search subdomain	CS2214	CS2214: Computer Architecture course

			found
--	--	--	-------

8. QUALIFICATION PROVISIONS

The primary method used to review documentation and code for quality is inspection. The inspection will be performed by the SQA group, a group formed by all the members of the team. For document-related corrections, copies of the completed document will be distributed to members of the team. Initially, an overview of the document will be inspected. This document will be carefully read to understand the document in detail. Each member will review the document to identify faults, but not correct them. The faults identified will be documented and the one responsible for the document will then resolve those faults. In the follow-up, the document will be inspected again to see if the faults were not satisfactorily resolved.

For code-related corrections, we will use a version-control system using Git. To synchronize the project with all other programmers, the project will be frozen by the programmer making the bug fix to the code. After that programmer is done fixing the fault, the programmer will push the new version of the project using Git version control. The other programmers will work on this newer version with the bug fix implemented.

9. REQUIREMENTS TRACEABILITY

During weekly meetings and reviews, members will update the team with the requirements they were assigned and their progress. Completed requirements will be updated in the traceability matrix. On completion of a requirement, the team member will be assigned another requirement to work on from the matrix. Changes in requirements will result in updates in both the documentation and the traceability matrix.

10. RATIONALE

Sharit is created with the goal of an easier file sharing system that fosters community within a group. Its intentions are to be simplistic in nature while boasting required functionalities to fulfill its purpose. Clear organization will allow intuitive understanding to allow the user to quickly execute their actions.

11. NOTES

Sharit is developed to be accessible only by those with an NYU email address. In the future access may be expanded to allow all email addresses.

12. APPENDICES

12.1 DICTIONARIES

CLASSES

User - represents a user in the system		
username	The user's username	String
firstName	The user's first name	String
lastName	The user's last name	String
password	The user's password	String
email	The user's email address	String
dateCreated	The timestamp the user was created	String

lastLogin	The last timestamp when the user logged in	String
privilege	The user's privilege	int
dob	The user's date of birth	String
organizations	A collection of the organizations the user is part in	Collection<String>
createAccount(username, fristname, lastName, password, email, dob)	Creates a new user	Boolean - returns true if successful
changePassword(newPassword)	Changes the user's password	void
login(username, password)	Logs in the user	Boolean - returns true if successful
changeEmail(newEmail)	Changes the user's email address	void

Domain - represents an organization's domain		
name	The domain's name	String
description	A description of the domain	String
privilegeLevel	The user privilege required to access the domain	int
organization	The organization this domain belongs to	String
createDomain(name, description, privilegeLevel)	Creates a new domain with the passed in parameters	Boolean - returns true if successful
changeName(newName)	Changes the domain's name	void

changeDescription(newDescription)	Changes the domain's description	void
changePrivilegeLevel(newPrivilegeLevel)	Changes the domain's access privilege level	void

Subdomain - represents a subdomain, a mini-domain part of a domain		
name	The subdomain's name	String
description	A description of the subdomain	String
domain	The domain name this subdomain is part of	String
organization	The organization this subdomain belongs to	String
createSubdomain(name, description, domain, organization)	Creates a new subdomain with the passed in parameters	Boolean - returns true if successful
changeName(newName)	Changes the subdomain's name	void
changeDescription(newDescription)	Changes the subdomain's description	void

Thread - represents a topic of discussion		
title	The thread's name	String
comments	A collection of all the Comments in the thread where the first comment is original post	Collection<Comment>

datePosted	The timestamp the thread was created	String
user	The user who created the thread	String
createThread(title, comment, subdomain)	Creates a new thread in the specified subdomain	Boolean - returns true if successful

Comment - represents a user's comment		
user	The user who posted this comment	String
comment	The user's comment	String
datePosted	The timestamp the thread was created	String
createComment(thread, comment)	Creates a new comment in the specified thread)	Boolean - returns true if successful
deleteComment()	Deletes the comment from the thread	void

RELATIONSHIPS

Class 1	Class 2	Description	Cardinality
User	Domain	A user can access a domain	Many-to-many
Domain	Subdomain	A subdomain is part of domain	One-to-many
Subdomain	Thread	A thread is part of a subdomain	One-to-many
Thread	Comment	A comment is part of a thread	One-to-many

12.2 UML DIAGRAMS

UML diagrams can be found in section 5.

12.3 SCHEDULE TRACKING

Artifact or Deliverable	Whom	Estimated	Actual	Difference
Initial SRS	Allen Zheng	3 hr	4 hr	1 hr
	Hui Huang	3 hr	5 hr	2 hr
	Kenneth Liang	3 hr	7 hr	4 hr
	Warlon Zeng	3 hr	5 hr	2 hr
	Summary	12	21 hr	9 hr
SRS 2.0	Allen Zheng	2 hr	2 hr	0 hr
	Hui Huang	2 hr	4 hr	2 hr
	Kenneth Liang	2 hr	3 hr	1 hr
	Warlon Zeng	2 hr	4 hr	2 hr
	Summary	8 hr	13 hr	5 hr
SPMP	Allen Zheng	3 hr	3 hr	0 hr
	Hui Huang	3 hr	5 hr	2 hr
	Kenneth Liang	3 hr	4 hr	1 hr
	Warlon Zeng	3 hr	4 hr	1 hr
	Summary	12 hr	16 hr	4 hr
SAS	Allen Zheng	3 hr	4 hr	1 hr
	Hui Huang	3 hr	5 hr	2 hr
	Kenneth Liang	3 hr	3 hr	0 hr
	Warlon Zeng	3 hr	2 hr	1 hr

	Summary	12 hr	14 hr	2 hr
--	---------	-------	-------	------

Cumulative

Whom	Estimated	Actual	Difference
Allen Zheng	11 hr	13 hr	2 hr
Hui Huang	11 hr	19 hr	8 hr
Kenneth Liang	11 hr	17 hr	6 hr
Warlon Zeng	11 hr	15 hr	4 hr
Summary	44 hr	64 hr	20 hr

12.4 DEFECT TRACKING





































Artifact or Deliverable	Whom	Estimated	Actual	Difference
Initial SRS	Allen Zheng	5	3	2
	Hui Huang	5	2	3
	Kenneth Liang	5	4	1
	Warlon Zeng	5	2	3
	Summary	20	11	9
SRS 2.0	Allen Zheng	4	2	2
	Hui Huang	4	5	1
	Kenneth Liang	4	3	1
	Warlon Zeng	4	2	2
	Summary	16	12	6
SPMP	Allen Zheng	5	4	1
	Hui Huang	5	5	0
	Kenneth Liang	5	6	1
	Warlon Zeng	5	3	2
	Summary	20	18	4
SAS	Allen Zheng	7	3	4
	Hui Huang	7	8	1
	Kenneth Liang	7	5	2
	Warlon Zeng	7	4	3

	Summary	28	20	10
--	---------	----	----	----

Cumulative

Whom	Estimated	Actual	Difference
Allen Zheng	21	12	9
Hui Huang	21	20	1
Kenneth Liang	21	18	3
Warlon Zeng	21	11	10
Summary	84	61	23

12.5 Gantt Chart/Microsoft Project Schedule

ID		Task Mode	Task Name	Duration	Start	Finish	Resource Initials
1			Sharit Project Requirements - Analysis and	68 days	Mon 2/1/16	Wed 5/4/16	KL,HH,WZ,AZ
2			Project Proposal	13 days	Mon 2/1/16	Wed 2/17/16	KL,HH,WZ,AZ
3			Software Requirements Specifications	21 days	Wed 2/24/16	Wed 3/23/16	KL,HH,WZ,AZ
4			All sections excluding 7.2, 7.3	9 days	Wed 2/24/16	Mon 3/7/16	KL,HH,WZ,AZ
5			Sections 7.2, 7.3	12 days	Tue 3/8/16	Wed 3/23/16	KL,HH,WZ,AZ
6			Software Project Management Plan	10 days	Thu 3/24/16	Wed 4/6/16	KL,HH,WZ,AZ
7			Risk Analysis	2 days	Thu 3/24/16	Fri 3/25/16	KL,HH,WZ,AZ
8			Creation	3 days	Mon 3/28/16	Wed 3/30/16	KL,HH,WZ,AZ
9			Review	3 days	Thu 3/31/16	Mon 4/4/16	KL,HH,WZ,AZ
10			Post Maintenance	2 days	Tue 4/5/16	Wed 4/6/16	KL,HH,WZ,AZ
11			Software Analysis Specification	8 days	Thu 4/7/16	Mon 4/18/16	KL,HH,WZ,AZ
12			Risk Analysis	2 days	Thu 4/7/16	Fri 4/8/16	KL,HH,WZ,AZ
13			Creation	2 days	Mon 4/11/16	Tue 4/12/16	KL,HH,WZ,AZ
14			Review	2 days	Wed 4/13/16	Thu 4/14/16	KL,HH,WZ,AZ
15			Post Maintenance	2 days	Fri 4/15/16	Mon 4/18/16	KL,HH,WZ,AZ
16			Software-Design-Document	NEXT-SEMESTER	NEXT-SEMESTER	NEXT-SEMESTER	KL,HH,WZ,AZ
17			Risk-Analysis	NEXT-SEMESTER	NEXT-SEMESTER	NEXT-SEMESTER	KL,HH,WZ,AZ
18			Creation	NEXT-SEMESTER	NEXT-SEMESTER	NEXT-SEMESTER	KL,HH,WZ,AZ
19			Review	NEXT-SEMESTER	NEXT-SEMESTER	NEXT-SEMESTER	KL,HH,WZ,AZ
20			Post-Maintenance	NEXT-SEMESTER	NEXT-SEMESTER	NEXT-SEMESTER	KL,HH,WZ,AZ
21			Oral Presentation	12 days	Tue 4/19/16	Wed 5/4/16	KL,HH,WZ,AZ

