NYU | TANDON SCHOOL OF ENGINEERING

Computer Science and Engineering

Sharit

Requirements/Analysis Specification

Version 1.0

Document Number: RAS-001

Team B6

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	October 4, 2016	Allen Zheng
Hui Huang	Author	October 4, 2016	Hui Huang
Kenneth Liang	Author	October 4, 2016	Kenneth Liang
Warlon Zeng	Author	October 4, 2016	Warlon Zeng

REVISION LEVEL

Date	Revision Number	Purpose
October 4, 2016	Version 1.0	Initial Release

Table of Contents

1. INTRODUCTION

1.1 Purpose

The purpose of this document is detail all requirements for the project Sharit. Business requirements and software requirements will be presented here. This includes the component architecture, functional requirements, non-functional requirements, and personnel-related requirements.

2. SCOPE

2.1 Identification

This document is the Requirements/Analysis Specification for Sharit. This document is revision 1.0, number 1.

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 will have permissions regarding which organization domain and subsequent subdomains they may access.

2.3 Objectives

The objective is to release the final product on December 20, 2016. The next priority is the Software Design Document. The project will be utilizing an incremental life cycle with the deliverables listed below:

Software Project Management Plan	September 27, 2016
Requirements and Analysis Specifications	October 4, 2016
Software Design Document	October 25, 2016
Implementation and Demonstration	November 30, 2016
Presentation	Last Two Weeks of 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

The document is split into 4 main parts. Sections 1, 2, 3, 11, and 12 describe the purpose and intentions of Sharit and this document in general terms. Section 4 is entirely related to the business aspect of the Sharit project. Sections 5, 6, 7, 8, & 9 describe the bounds and requirements of Sharit along with testing procedures during the development of Sharit. The last part, section 13, is simply the scheduling for the project thus far and diagrams.

3. REFERENCE DOCUMENTS

Team A6 System Requirements Specification, Version 2.0, March 23, 2016
Team A6 System Analysis Specification, Version 1.0, April 18, 2016
Team B6 Software Project Management Plan, Version 2.0, September 27, 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 softwares, 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

There are no regulatory and legal requirements.

4.4 Market Considerations

Since registration is entirely dependent on having an NYU email address (@nyu.edu), the targeted group of consumers will be NYU students and faculty. The project will proceed to follow a subscription model in which the school will pay for users' subscriptions.

4.5 Risks and Alternatives

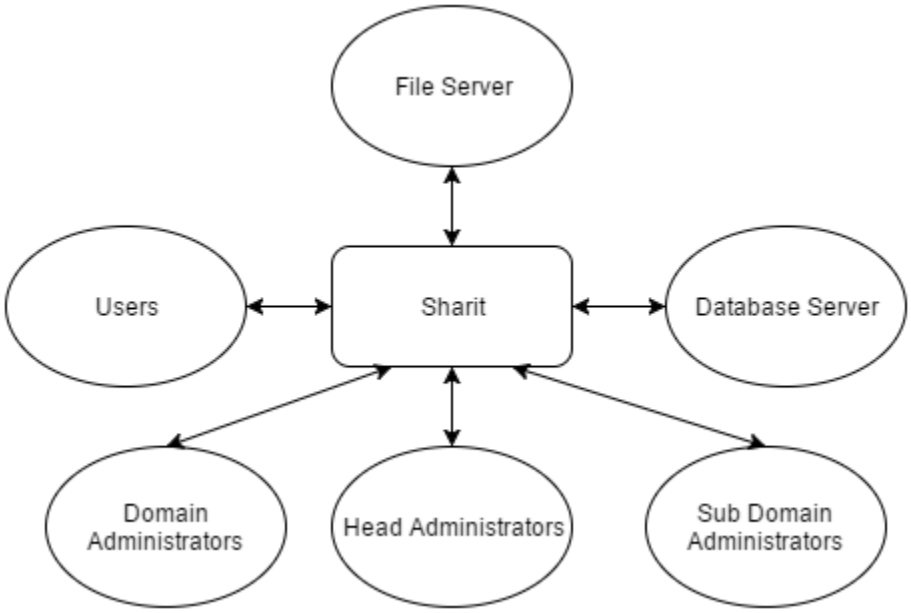
1. Not enough users signing up
   1.1. Can potentially expand to more schools outside of NYU
2. Not enough content being produced
   2.1. Can offer monetary incentives for students to contribute as a start-up and later retract offers after a self-sufficient number of contributions generated
3. Work that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

4.6 Human Resources and Training

1. Student programmers
2. PEAN Stack training
   2.1. PEAN Stack: PostgreSQL, Express, AngularJS, Node
   2.2. Windows - operating system
   2.3. Git - version control

5. CONTEXT DIAGRAM

5.1 High Level (Level 0)



6. FUNCTIONAL REQUIREMENTS ANALYSIS SPECIFICATION

6.1 Functional Descriptive Detailed Requirements

1.0 The website will provide functions for the user to login and register as well as a profile
- 1.1. A user will be able to register using their student or company email
- 1.2. Existing users will be able to log in using the credentials they registered with
- 1.3. Each user will have a profile with information such as name and school/company
- 1.4. Options will be provided to change basic information

2.0 Each organization will be allowed to create a domain on the server

2.1. Domains are only accessible to users in the organization or with permission
2.2. Domains will have administrators or moderators
2.3. Admins can modify aspects of the domain as well as grant permissions
2.4. Each domain can have subdomains that correspond to different parts of the organization

3.0 Users will be able to upload and download files
3.1. A user will be able to create a thread where the file will be uploaded
3.2. Threads will be part of a subdomain
3.3. A user that is part of the subdomain may choose to view or download the file

4.0 Each thread will support feedback/comments
4.1. A thread can be upvoted/downvoted to show popularity within a domain
4.2. Users can leave comments on the thread
4.3. Other users can respond to a previous user's comment
4.4. Users will be able to upvote/downvote other user's comments

5.0 A search will be provided to look up specific information in the system
5.1. Users can look up specific files/threads in the domain
5.2. Users can search for other users and view their profile
5.3. Users can search for domains or subdomains to join

6.2 System Capability Requirements

6.2.1 Capabilities

Each of the use cases correspond to each bubble (or combination of bubbles) in the use case diagram in section 5.1.

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	An authorized user can access a domain or subdomain, they can create

subdomain	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.

6.3 User Interface Requirements

The user interface will be simple but effective. On the top navigation bar will be all the domains that the user is in. Entering a domain will have the top navigation bar show the subdomains of that domain and so on and so forth for each level. To the left side of the page and below the navigation bar will be the menu, allowing access to the login, account settings, site options, etc. To the right of the menu and below the top navigation bar is the main page content. This is where the thread postings will be. Each posting will have a title, creator, date of creation, rating, the number of comments, and the number of attachments shown. There will also be upvote and downvote buttons next to the postings. Accessing a post will result in the thread post appearing the top portion of the main content view with all comments below.

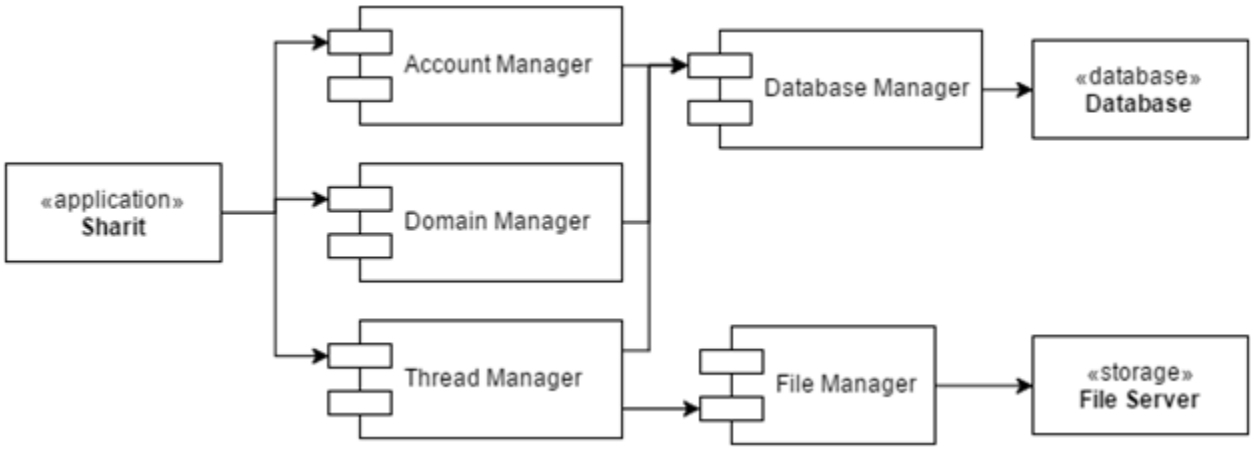
The main content page will be mainly white with black text. The top navigation bar will be a blue color and the menu will be a black color. These colors will be changeable in the settings.

6.4 Component Architecture

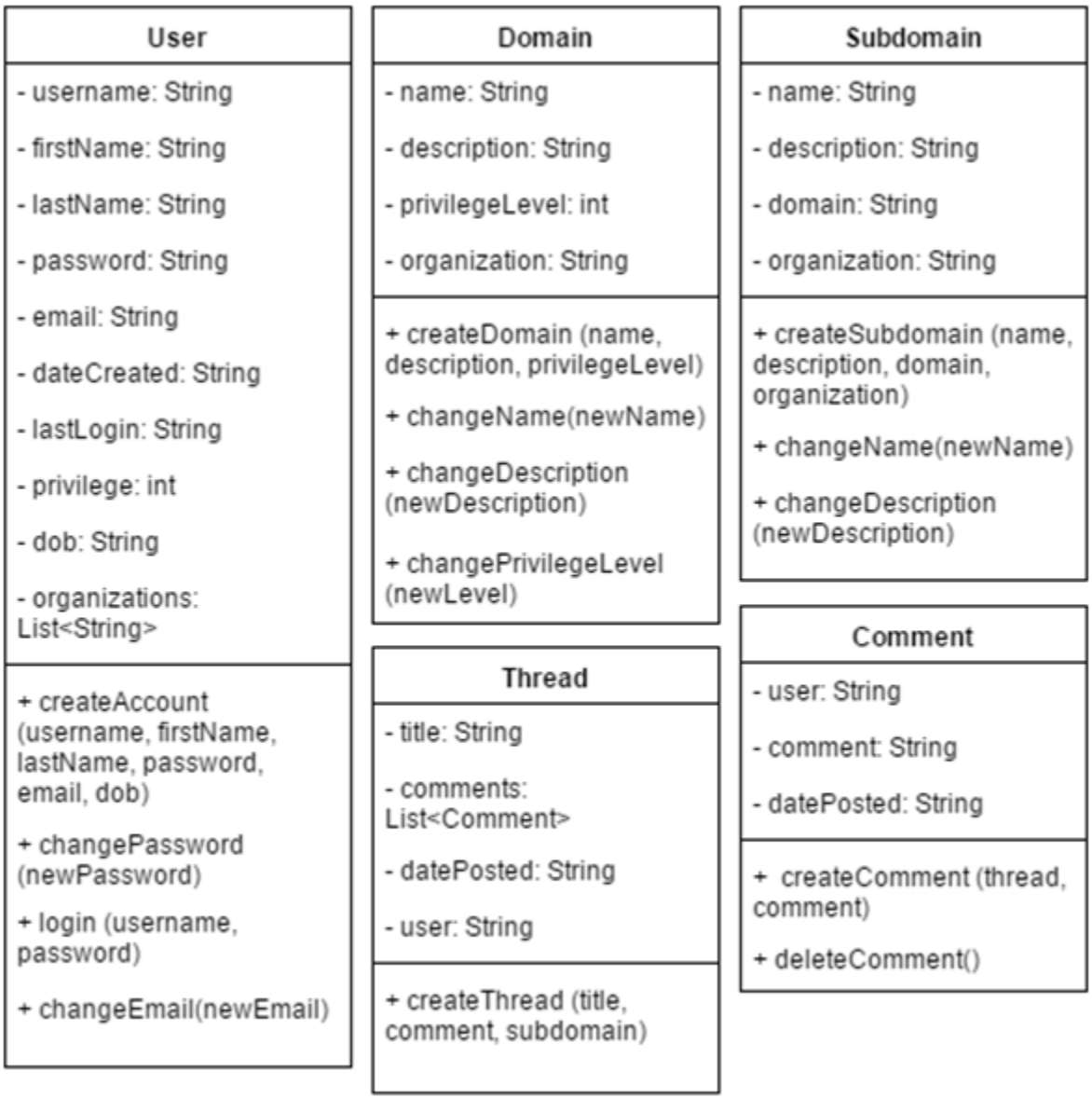
6.4.1 Component Descriptions

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.

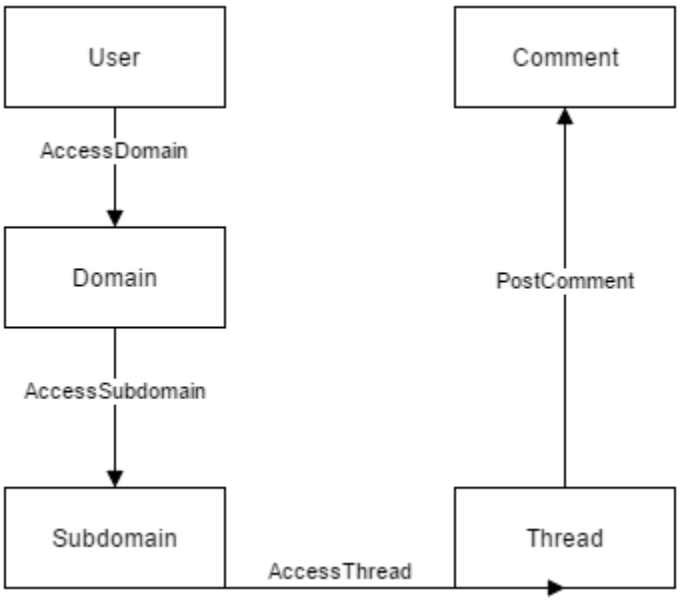
6.4.2 Component Architecture Diagram



6.5 Class Diagrams



6.6 Class Relationship/Interaction Diagrams



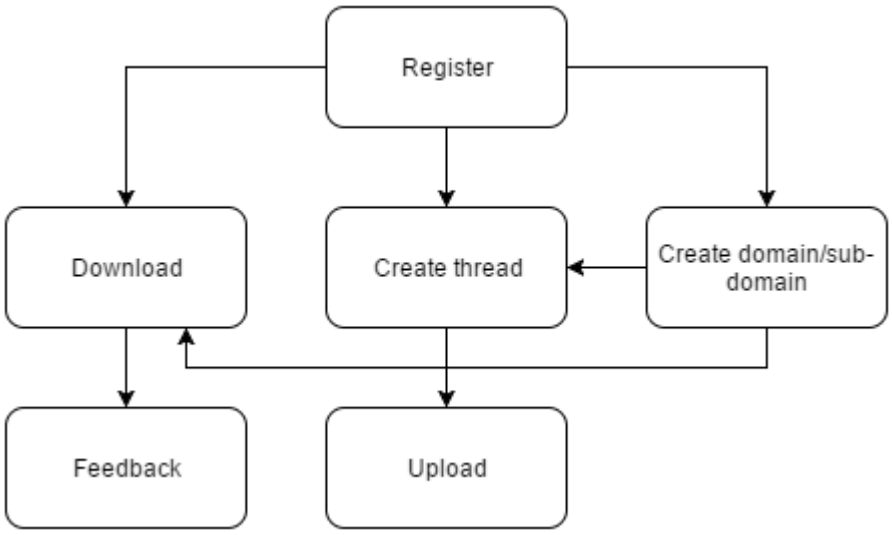
6.7 Event Section

6.7.1 Event Dictionary

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,	This allows users to engage

	upvote/downvote	with each other

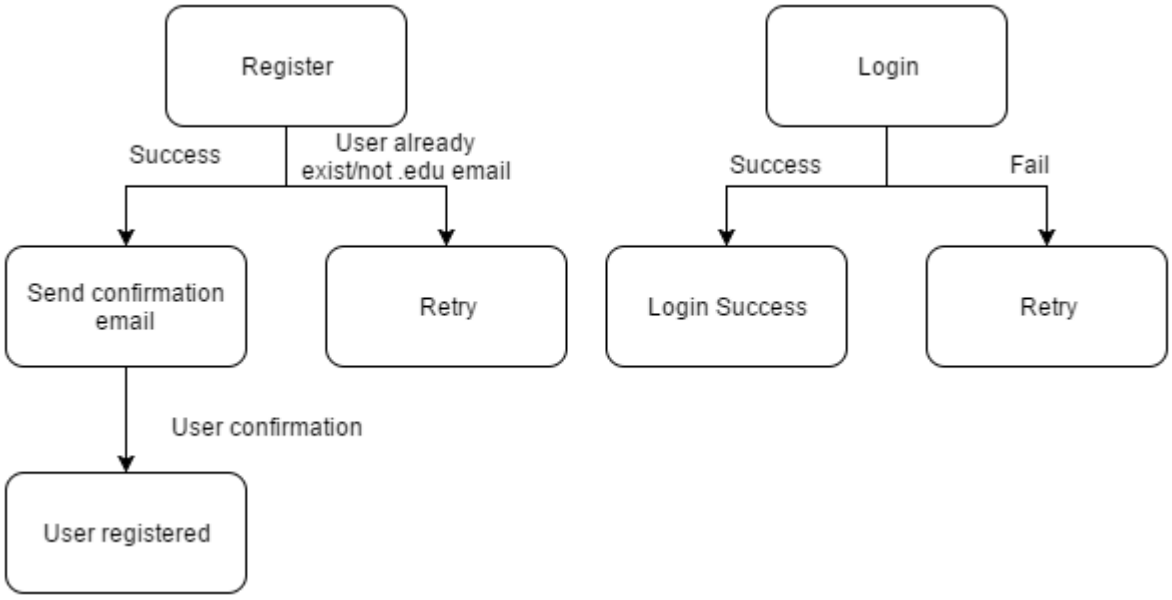
6.7.2 Event Diagrams



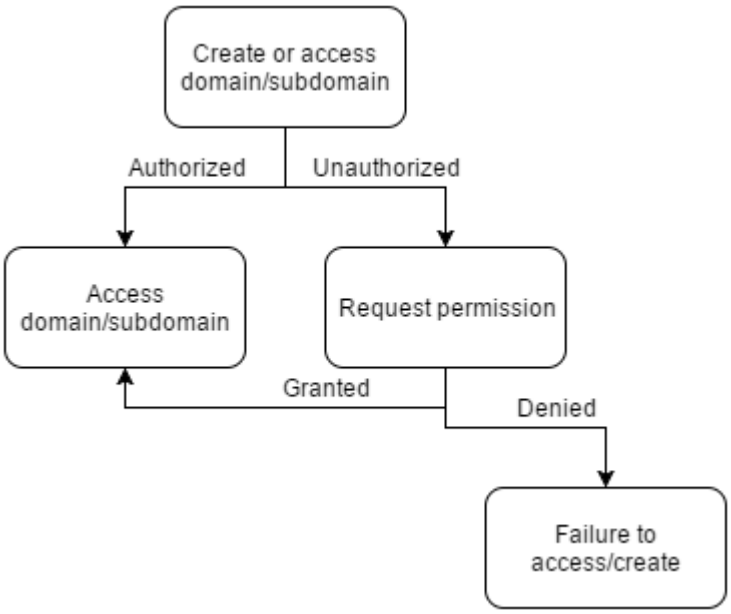
6.8 Activity/State (Scenario) Section

6.8.1 Activity (Scenario) Diagrams

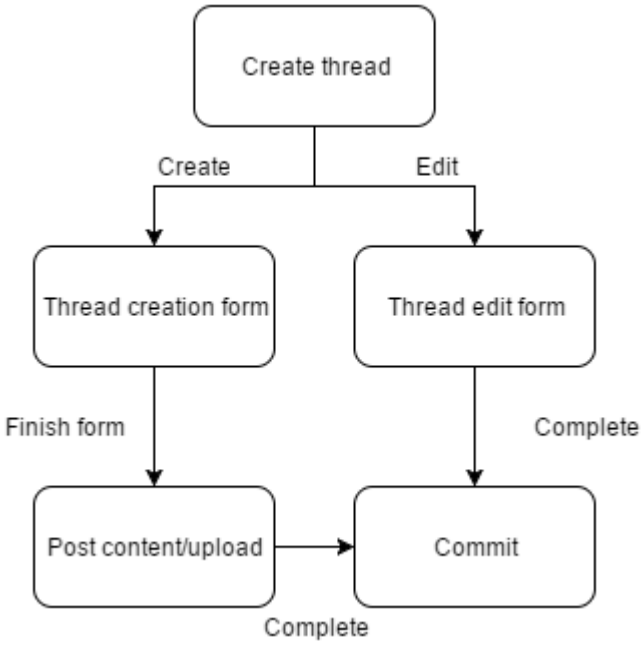
The user tries to login or register into Sharit.



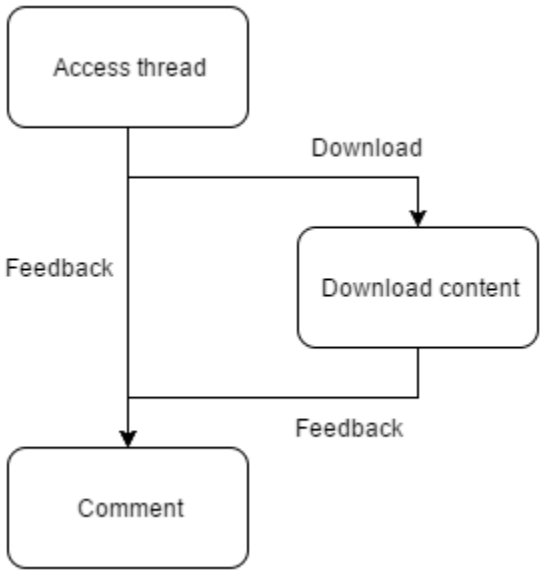
The user tries to access or create a domain or a subdomain.



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

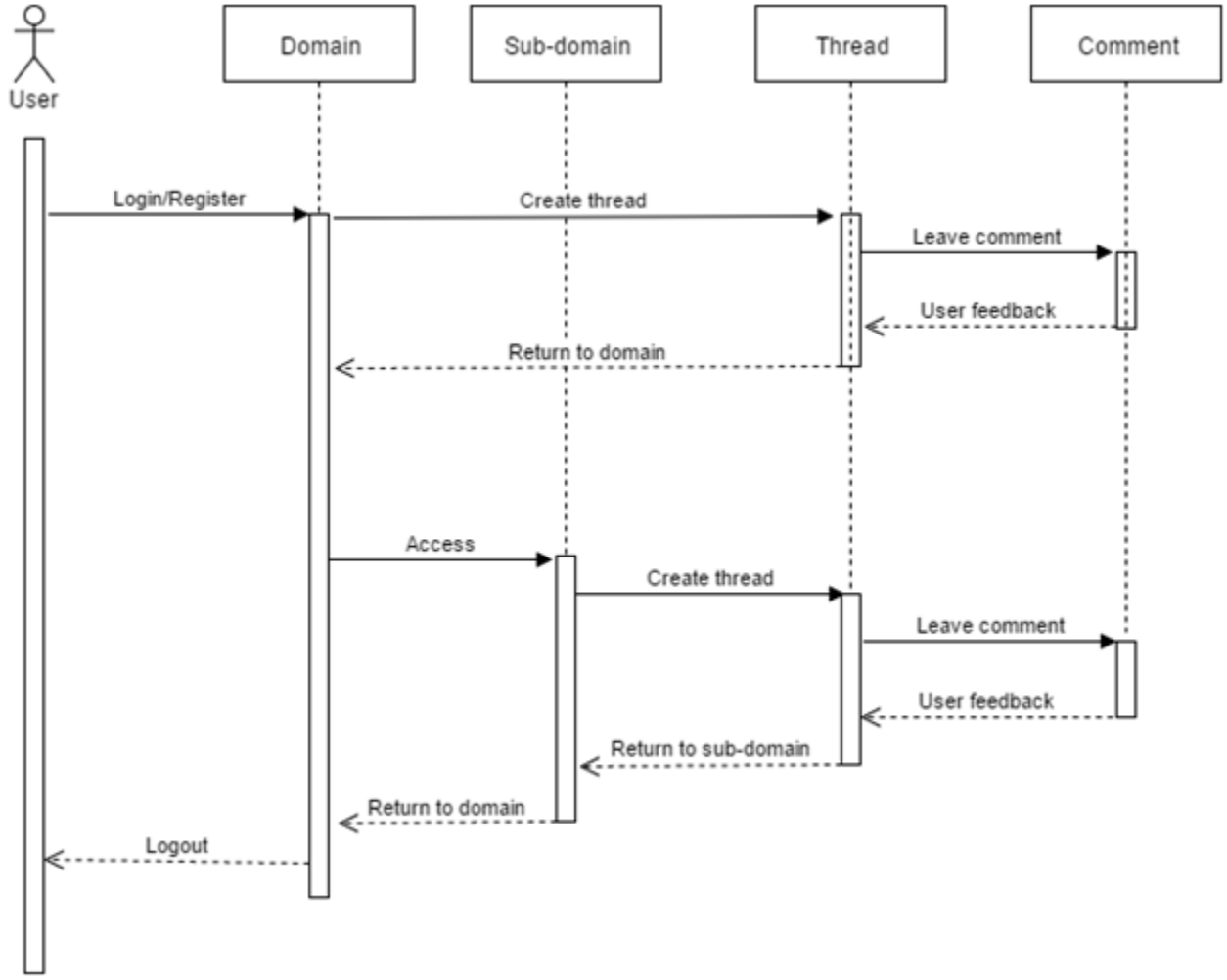


The user wants to download content from a thread or leave feedback


Access thread
Download
Feedback
Download content
Feedback
Comment

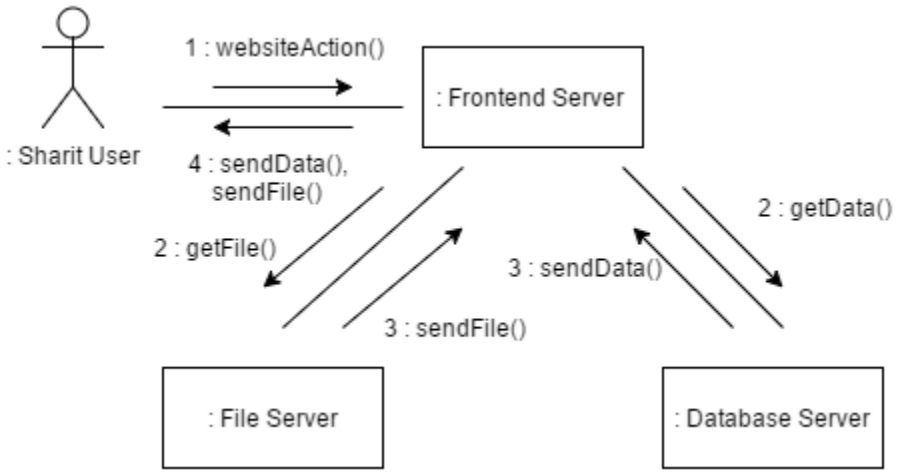
6.8.2 Activity (Scenario) Specification

6.9 Sequence Diagrams



6.10 Collaboration Diagrams

This is a very simplified collaboration diagram as there would be too many actions to fit everything on one clean diagram.



6.11 Dictionaries

See appendix 13.1 for the relevant dictionary information

7. NON-FUNCTIONAL/OPERATIONAL REQUIREMENTS

7.1 System External Interface Requirements

Sharit will use an external server for our data files. The backend interface will communicate with this server to request file reads and writes. A separate server for files will lessen the load on the other servers. The requirement will therefore be the capability to connect to the external server.

7.2 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.

7.3 Security and Privacy Requirements

The credentials maintained by users is a simple username and password verification system. The system maintains that each new user has a unique username

that is not in the system already. Privacy is equivalent to access in our system. Files can only be downloaded and uploaded by a logged in a user. Domains given to groups will be monitored monthly, but contents will not be released. Files uploaded will likewise be checked for legitimacy, but not released. Files deemed to be dangerous will be made inaccessible to users.

7.4 System Environment Requirements

The product runs in every environment with a mouse, keyboard, and a working internet connection. A hard drive is optional if the user needs to store files offline. The user can access their account assuming they do not forget their login information.

7.5 Computer Resource Requirements

7.5.1 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

7.5.2 Computer Hardware Resource Requirements

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

7.5.3 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)

7.5.4 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.

7.6 System Quality Factors

In order to have a high quality system, the goal is have the lowest possible number of faults and defects. To achieve this, programmers will partake in peer code reviews, walkthroughs, and inspections. This will greatly reduce the number of faults found in a system, thus increasing quality.

7.7 Design and Construction Constraints

The life cycle model used will be the Iteration and Incremental Life Cycle Model. This model is flexible and adaptable. During the design of the website, there will be many changes in its appearance and feel. Increments will be used to handle these changes, while the inherent Waterfall Model will provide a structured basis for development.

Coding will be done primarily using the MEAN stack: MongoDB, Node, Express, and AngularJS. These technologies will be used to create the front-end, the website.

Git will be used for source-control during the development.

7.8 Personnel-Related Requirements

The head administrator will periodically check the content and files uploaded for all the domains and subdomains. This check will be quick and cursory. It is up to the domain administrator and subdomain administrators to conduct more thorough checks on the content in their assigned domain.

7.9 Training-Related Requirements

The accessibility of the website is very simple. Users will register for an account that is aided by a step-by-step form process. Functionality of website will be self-explanatory by a friendly interface.

7.10 Logistics-Related Requirements

All crashes and errors will be logged in the database. Data usage rates will be noted to determine the rate at which space will need to be added to the servers. Git will log code history.

7.11 Packaging Requirements

There are no packaging requirements for Sharit.

7.12 Precedence and Criticality Requirements

The main requirement is to ensure that Sharit does not turn into a site where illegal files (copyrighted material) are shared. File security is also very important; users who do not have access to a domain should not have access to the files inside.

7.13 Other Requirements

The Sharit system will be made scalable so that if the user base gets large, the system can adjust accordingly to accommodate.

8. 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

9. 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.

10. REQUIREMENTS TRACEABILITY

The functional and non-functional requirements will be thoroughly traced throughout the life of the software engineering process. Each specification will be corresponded with an artifact in each of the workflows and be traced. Requirements tracing is done to avoid litigation and also mitigates liabilities. Keeping track of requirements will make sure everything is implemented and meets the specifications.

11. RATIONALE

Finding information relevant to coursework can drastically improve one's grades. NYU-Poly has historically incurred an usually low graduation rate (50~60%). Information, pertaining previous semester student notes, tests, and possibly homeworks, should be upheld as samples for students currently taking the course to help them review. In the requirements, we aim to answer general questions posted by students that not even google.com, NYUClasses, and instructors could not provide. This is most often phrased "what's the format" or "is there a sample midterm?". To uphold integrity of information, we provide and limit information and registration to students with a nyu.edu email address.

In the future, if the project expands outside the NYU community, Sharit can be a creative solution for an easier file sharing system while also fostering a sense of community within a group. It 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.

12. NOTES

Currently, 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.

13. APPENDICES

13.1 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	int

	domain	
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	Changes the subdomain's	void

(newDescription)	description	

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

13.2 UML Diagrams

UML diagrams can be found in section 6.

13.3 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 hr	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

Initial 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
SPMP 2.0	Allen Zheng	2 hr	1 hr	1 hr
	Hui Huang	2 hr	2 hr	0 hr
	Kenneth Liang	2 hr	1 hr	1 hr
	Warlon Zeng	2 hr	2 hr	0 hr
	Summary	8 hr	6 hr	2 hr
Initial 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
Initial RAS	Allen Zheng	3 hr	3 hr	0 hr
	Hui Huang	3 hr	4 hr	1 hr
	Kenneth Liang	3 hr	5 hr	2 hr
	Warlon Zeng	3 hr	3 hr	0 hr
	Summary	12 hr	15 hr	3 hr

Cumulative

Whom	Estimated	Actual	Difference
Allen Zheng	16 hr	17 hr	1 hr

Hui Huang	16 hr	25 hr	9 hr
Kenneth Liang	16 hr	23 hr	7 hr
Warlon Zeng	16 hr	20 hr	4 hr
Summary	64 hr	85 hr	21 hr

13.4 Defect Tracking

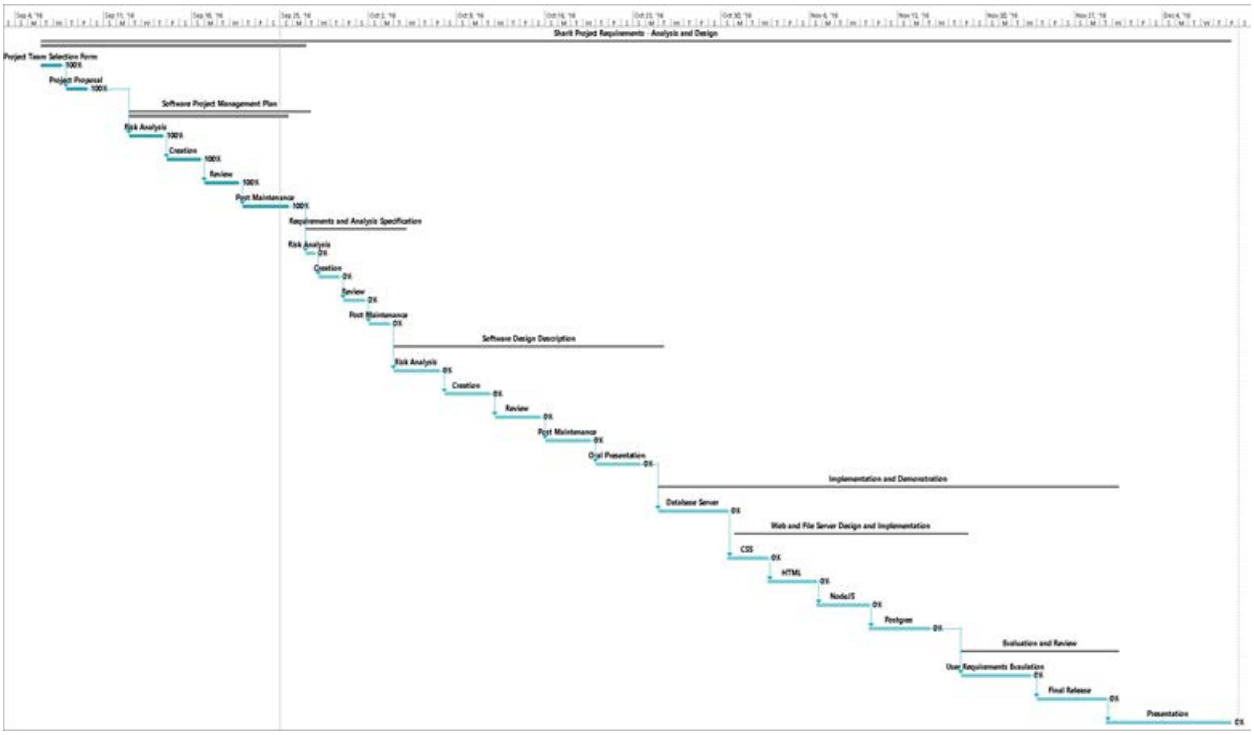
Artifact or Deliverable	Whom	Estimated	Actual	Difference
Initial SRS	Allen Zheng	5	3	2
	Hui Huang	5	2	3
	Kenneth Liang	5	4	1
	Warlon Zeng	5	2	3
	Summary	20	11	9
SRS 2.0	Allen Zheng	4	2	2
	Hui Huang	4	5	1
	Kenneth Liang	4	3	1
	Warlon Zeng	4	2	2
	Summary	16	12	6
Initial SPMP	Allen Zheng	5	4	1
	Hui Huang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

SPMP 2.0	Allen Zheng	3	4	1
	Hui Huang	3	3	0
	Kenneth Liang	3	4	1
	Warlon Zeng	3	2	1
	Summary	12	13	3
Initial 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
Initial RAS	Allen Zheng	6	5	1
	Hui Huang	6	7	1
	Kenneth Liang	6	9	3
	Warlon Zeng	6	4	2
	Summary	24	25	7

Cumulative

Whom	Estimated	Actual	Difference
Allen Zheng	30	21	9
Hui Huang	30	30	0
Kenneth Liang	30	31	1
Warlon Zeng	30	17	13
Summary	120	99	23

13.5 Gantt Chart/Microsoft Project Schedule



ID		Task Name	Duration	Start	Finish	Resource Initials
1		**Sharit Project Requirements - Analysis and Design**	**106 days**	**Tue 9/6/16**	**Fri 12/9/16**	**KL,HH,WZ,AZ**
2	✓	Project Team Selection Form	2 days	Tue 9/6/16	Wed 9/7/16	KL,HH,WZ,AZ
3	✓	Project Proposal	2 days	Thu 9/8/16	Fri 9/9/16	KL,HH,WZ,AZ
4	✓	**Software Project Management Plan**	**16 days**	**Tue 9/13/16**	**Tue 9/27/16**	**KL,HH,WZ,AZ**
5	✓	Risk Analysis	3 days	Tue 9/13/16	Thu 9/15/16	KL,HH,WZ,AZ
6	✓	Creation	3 days	Fri 9/16/16	Sun 9/18/16	KL,HH,WZ,AZ
7	✓	Review	3 days	Mon 9/19/16	Wed 9/21/16	KL,HH,WZ,AZ
8	✓	Post Maintenance	4 days	Thu 9/22/16	Sun 9/25/16	KL,HH,WZ,AZ
9		**Requirements and Analysis Specification**	**9 days**	**Tue 9/27/16**	**Tue 10/4/16**	**KL,HH,WZ,AZ**
10		Risk Analysis	1 day	Tue 9/27/16	Tue 9/27/16	KL,HH,WZ,AZ
11		Creation	2 days	Wed 9/28/16	Thu 9/29/16	KL,HH,WZ,AZ
12		Review	2 days	Fri 9/30/16	Sat 10/1/16	KL,HH,WZ,AZ
13		Post Maintenance	2 days	Sun 10/2/16	Mon 10/3/16	KL,HH,WZ,AZ
14		**Software Design Description**	**24 days**	**Tue 10/4/16**	**Tue 10/25/16**	**KL,HH,WZ,AZ**
15		Risk Analysis	4 days	Tue 10/4/16	Fri 10/7/16	KL,HH,WZ,AZ
16		Creation	4 days	Sat 10/8/16	Tue 10/11/16	KL,HH,WZ,AZ
17		Review	4 days	Wed 10/12/16	Sat 10/15/16	KL,HH,WZ,AZ
18		Post Maintenance	4 days	Sun 10/16/16	Wed 10/19/16	KL,HH,WZ,AZ
19		Oral Presentation	4 days	Thu 10/20/16	Sun 10/23/16	KL,HH,WZ,AZ
20		**Implementation and Demonstration**	**41 days**	**Tue 10/25/16**	**Wed 11/30/16**	**KL,HH,WZ,AZ**
21		Database Server	6 days	Tue 10/25/16	Sun 10/30/16	KL,HH,WZ,AZ
22		**Web and File Server Design and Implementation**	**21 days**	**Mon 10/31/16**	**Fri 11/18/16**	**KL,HH,WZ,AZ**
23		CSS	4 days	Sun 10/30/16	Wed 11/2/16	AZ
24		HTML	4 days	Wed 11/2/16	Sun 11/6/16	HH
25		NodeJS	5 days	Sun 11/6/16	Thu 11/10/16	WZ
26		Postgres	5 days	Thu 11/10/16	Tue 11/15/16	KL
27		**Evaluation and Review**	**14 days**	**Fri 11/18/16**	**Wed 11/30/16**	**KL,HH,WZ,AZ**
28		User Requirements Evaulation	6 days	Fri 11/18/16	Wed 11/23/16	KL,HH,WZ,AZ
29		Final Release	6 days	Thu 11/24/16	Tue 11/29/16	KL,HH,WZ,AZ
30		Presentation	11 days	Tue 11/29/16	Fri 12/9/16	KL,HH,WZ,AZ