NYU | TANDON SCHOOL OF ENGINEERING

Computer Science and Engineering

Sharit

Software Project Management Plan

Version 1.0

Document Number: SPMP-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 6, 2016	Allen Zheng
Hui Huang	Author	April 6, 2016	Hui Huang
Kenneth Liang	Author	April 6, 2016	Kenneth Liang
Warlon Zeng	Author	April 6, 2016	Warlon Zeng

REVISION LEVEL

Date	Revision Number	Purpose
April 6, 2016	Version 1.0	Initial Release

Table of Contents

1. OVERVIEW

1.1 Project Summary

The purpose of this document is to specify the management for the website Sharit. This document will outline details, such as budget and management plans. This document is intended for the designers and the developers.

1.2 Purpose, Scope, and Objectives

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.

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.

Upon release, students that are part of the NYU can access the site with the email and creating a password. Teachers can also access the site with their NYU credentials.

1.3 Assumptions and Constraints

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.

This product will be programmed in C++ and Postgres. Any changes will not impact the product largely.

1.4 Project Deliverables

The priority is to release the Software Analysis Specifications (SAS) by April 11, 2016. The project will be utilizing an incremental life cycle with the deliverables listed below:

Software Project Management Plan	April 6, 2016
Software Analysis Specification	April 11, 2016
Software Design Document	April 25, 2016

1.5 Schedule and Budget Summary

The summary of the project schedule was made with Microsoft Project. The summary and Gantt Chart can be found in section 12.3. The project budget will be available in the next iteration of the document.

1.6 Evolution of the Plan

The Software Project Management Plan is a document that will be continuously updated throughout the software development process as requirements and other specifications change. The initial version of this document will be used as a foundation for future changes. Future changes to this baseline will be tracked and recorded by incrementing the version number of the document.

2. REFERENCES

Project Sharit, Team A6 Project Proposal, Version 1.0, March 7, 2016.
Project Sharit SRS-002, Version 2.0, March 23, 2016.

3. DEFINITIONS

Term	Definition
Artifact	A component of the software process
Audit	A software review of by auditor(s) who are/is independent (not part) of the team.
Backup Programmer	A programmer with similar skill to the chief programmer and capable of taking up the chief programmer's job as a reserve.
Baseline	A standard of which the product has upheld in approval.
Chief Programmer	Highly skilled programmer that reviews other programmers and directs them with his deep architectural understanding of the project.
Fault	A bug in a line of code or error in documentation.
Gantt Chart	A bar chart that projects the project schedule and tasks.
Inspection	A formal review of software code and documentation.
Milestone	An action or event marking a significant change or stage in development.
Programmer	A person that writes code for several components of the project.
Programming Secretary	A person that keeps the documentations and libraries of the project in order.
Project Deliverable	Product or document specified and produced to be delivered to the client.
Requirements	Description of function and non-functional features that may include a set of use cases to describe user interaction.
Risk	Foreseen or unforeseen unfortunate events that may hinder software development progress, productivity, and/or success.

Traceability Matrix	A table that corresponds the requirements to baseline documents
Walkthrough	An informal review of software code and documentation.

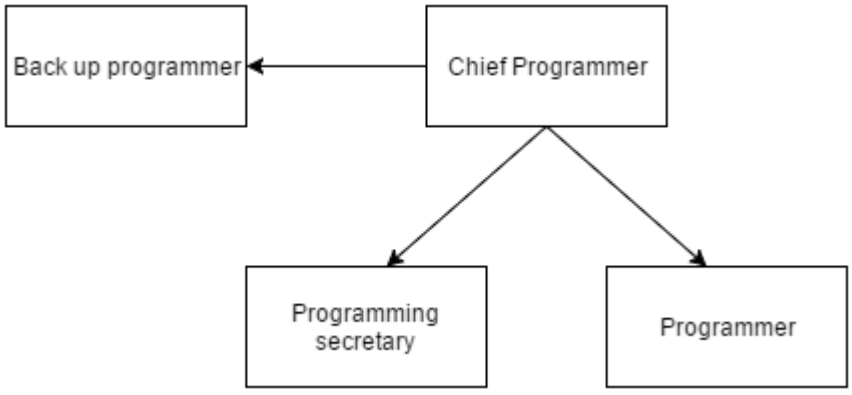
4. PROJECT ORGANIZATION

4.1 External Interfaces

The system uses no third-party software. All interfaces are internal and managed by the team.

4.2 Internal Structure

The development team will be following the chief programming model. Team members will be reporting to the Chief Programmer with their progress and any updates or to provide suggestions. Further details of team organization is provided in section 4.3.



4.3 Roles and Responsibilities

Member	Role

Kenneth Liang	Chief Programmer, reviewer, posts document
Hui Huang	Backup Programmer, document writer
Warlon Zeng	Programming Secretary, document writer
Allen Zheng	Programmer, document writer

The chief programmer is a highly skilled programmer that manages the team and understands the whole structure of the software project; this programmer is responsible for every single line of code written. A backup programmer is someone who can take over if the chief programmer is not available. This person has just as much understanding of the project as the chief programmer. A programming secretary is responsible for maintaining and updating project libraries and documentation. A programmer's only job is to code various components of a software project.

5. MANAGEMENT PROCESS

5.1 Start Up Plan

5.1.1 Estimation Plan

The estimation plan will be available in the next iteration of this document.

5.1.2 Staffing Plan

The staffing plan will be available in the next iteration of this document.

5.1.3 Resources Acquisition Plan

The resource acquisition plan will be available in the next iteration of this document.

5.1.4 Training plan

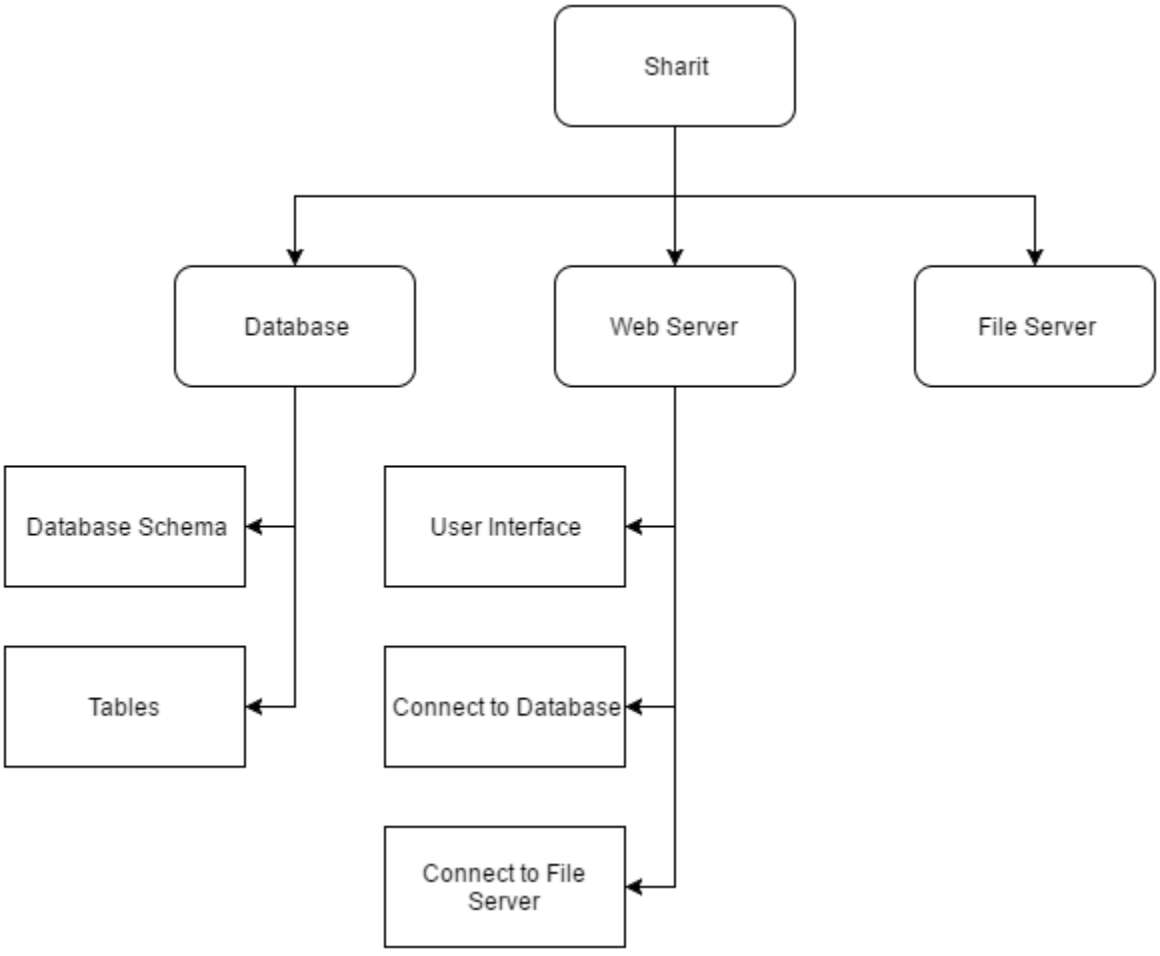
The training that is required for the members of the team are relational database design and web design. Two members of the team are already adequate in both these

things and the skills can be taught to the other two members. The skills that need to be taught include: constructing relational schema for the database, server side programming, and client side programming. Each team member will be following the lead of the chief programmer who will make the managerial and technical decisions.

5.2 Work Plan

5.2.1 Work activities

The Work Breakdown Structure depicts the work activities required to construct Sharit. There are three levels to the structure. The first level is Sharit, the entire project. The second level is the main components making up Sharit. The third level is the breakdown of each component into parts that are essential to its function.



Work Package Level 2	Resources	Estimated Duration	Products	Acceptance Criteria	Predecessor Activity	Successor Activity
Database	2 people	6 hours	Database	Consistent, holds all user information	Sharit	Database schema and tables
Web server	4 people	30 hours	Web server	Functional under load and simplistic	Sharit	User interface and connection to database and file server
File server	2 people	10 hours	File server	Able to store and retrieve files	Sharit	NA

Work Package Level 3	Resources	Estimated Duration	Products	Acceptance Criteria	Predecessor Activity	Successor Activity
Database schema	2 people	5 hours	Database	Boyce Codd Normal Form	Database	NA
Tables	4 people	1 hours	Database tables	Follows schema and insertable into database	Database	NA
User Interface	2 people	25 hours	Website	Easy to understand and addresses all necessary functionality	Web server	NA
Connect to database	1 person	2 hours	Connection to database	Functional	Web server	NA

Connect to file server	2 people	3 hours	Connection to file server	Functional, consistent connection during download and upload	Web server	NA

5.2.2 Schedule Allocation

The time sequencing constraints for the software project are as follows: the database must be completed before the server side code is written and the web pages must be constructed before the client side code is written. All the other work can be done concurrently by each programmer.

Each completed document for a project deliverable is a milestone. The deliverable documents are: project proposal, SRS, SPMP, SAS, and SDD. After all documents are completed, there will be an oral presentation. Following the oral presentation will be when the database construction milestone will be completed followed by the rest of the coding. A more detailed schedule can be found in section 12.3.

5.2.3 Resource Allocation

The following table shows the resource allocation breakdown for the third level of the structure. The third level is an expansion of the second and first levels.

Work Package (level 3)	# of Personnel	Required Skill Level	Software tools	Special test environments
Tables	2	Knowledge of SQL and normal forms	draw.io	--
Relational Schemas	2	Knowledge of SQL and relations	draw.io	--
User Interface	2	Knowledge of HTML, CSS, and JavaScript	A modern text editor	--
Connect to	2	Knowledge of C++ and	A modern	--

Database		PostgreSQL	text editor	
Connect to File Server	1	Knowledge of C++ and PostgreSQL	A modern text editor	Machine that can simulate a file server

5.2.4 Budget Allocation

The budget allocation will be available in the next iteration of the document.

5.3 Control Plan

5.3.1 Requirement Control and Traceability

All the requirements as stated in section 7.1 of the Software Requirements Specification (SRS) will the trace throughout the life of the software development process. Each specification will be corresponded with an artifact in each of the workflows and be traced in a traceability matrix. Requirements tracing is done to avoid litigation and also mitigates liabilities. Keeping track of requirements will make sure everything is implemented and meets the requirements as described in section 7.1.

Requirement changes will be controlled by traceability, impact analysis, and reviews. Changes that occur in the requirements needs to be assessed to determine its impact on the overall project. Reviews will be performed by all members of the team to ensure everything is in order. If a requirement changes, all subsequent documents will be updated from their current baseline.

5.3.2 Schedule Tracking and Adjustment

The software that is used for schedule tracking and adjustment is Microsoft Project. Microsoft Project will track milestones and other minor changes as well as the predicted time of completion and team members assigned to the task. If the project gets delayed by any unforeseen circumstance, the schedule will show the delay. Workload and performance will have to be adjusted so that the project will be completed in a timely manner. A Gantt Chart will be created from the schedule that visualizes each milestone of the project. The schedule and Gantt Chart can be found in section 12.3.

5.3.3 Budget Tracking and Adjustment

Budget tracking and adjustment will be available in the next iteration of the document.

5.3.4 Quality Control

The quality for all phases of the software project will be monitored and recorded throughout the whole life of the development process. The methods that will be used for quality control include: having a SQA group, qualification, and audits by each team member. All faults that are found will be documented along with its resolution for traceability purposes. Walkthroughs and inspections will be performed at regular intervals to ensure all the software meets expectations and is fault-free. Verification will be performed when each workflow concludes and validation will be performed when the software is ready to be released.

5.3.5 Reporting Mechanisms

Reporting mechanisms will be available in the next iteration of the document.

5.3.6 Metrics Collection Plan

Software metrics that will be documented throughout the project's development are: effort in person months, duration in months, quality based on number of faults, and the size of the project via lines of code. Each team member's hours will be also be documented in order to construct an estimate for the project duration and effort in person months. After each walkthrough or inspection of the software, another set of metrics will be recorded: fault density, inspection rate, fault detection rate, and fault detection efficiency. Every single metric will be recorded and updated once new data is available.

5.4 Risk Management Plan

A table will be used to label risks that may be encountered throughout the lifecycle model. Risks will be labeled, categorized by type, and given a contingency plan if encountered.

Risk	Type	Description	Detection/Prevention	Contingency Plan

Team member change	Personnel	A team member leaves or joins	Members are required to report if they are unavailable or planning to leave.	A meeting will be held with all members to discuss work division.
Behind schedule	Software	The project is not progressing on time.	During meetings, difficulties will be discussed.	Alternative solutions will be explored. The team will work overtime.
Database crash	Software	The database encounters an error	Database loads will be monitored daily.. Weekly backups will be made.	In the case of an inconsistent database, a backup will used to restore the data.
File server crash	Software	The file server becomes offline.	Statuses will be monitored daily. There will be more than one file server.	Fix the crashed server and restore its data with a backup server.
Overbudget	Budget	The costs are exceeded.	Weekly projection of costs will be made. Minimize resource usage when costs near budget limit.	Contact the client to discuss the next available course of action.

5.5 Post Implementation Plan

Post implementation plan will be available in the next iteration of the document.

6. TECHNICAL PROCESSES

6.1 Process Model

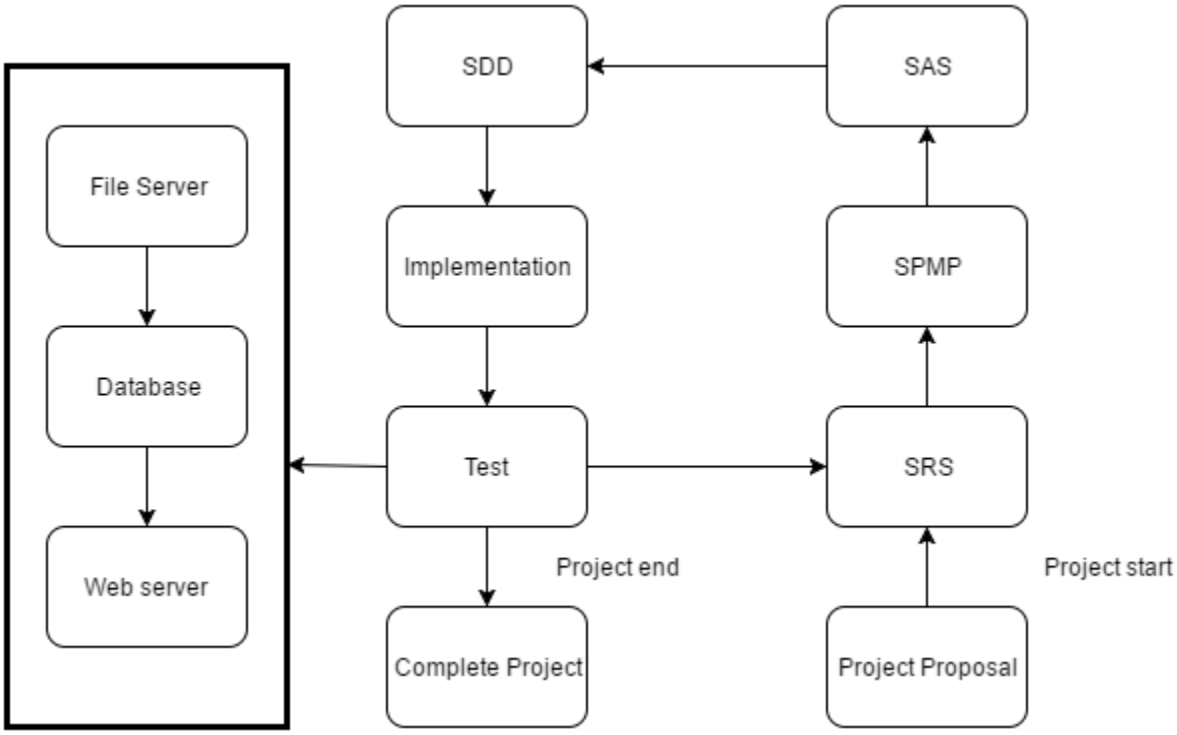
The project uses the iterative and incremental development model to deliver the software products. We chose this model due to its reflex nature and constant ability to revisit the five workflows: requirements, analysis, design, implementation, and test.

Each increment will be result of reaching a major milestone. These milestones include completion of the deliverables, completion of the database, completion of the file server, and finally the completion of the web server. The deliverables in this project are: the project proposal, Software Requirements Specification (SRS), Software Project Management Plan (SPMP), Software Analysis Specification (SAS), and Software Design Document (SDD).

During each increment, each workflow will be touched upon in an iterative process to keep everything up-to-date. Some workflows will be visited more than others depending on the increment. Initial increments will see more documentation, while later increments will see more implementation and testing. The diagram below shows a depiction of an increment with the various workflows adapted to fit the project.

The project is initiated by the client and terminated when the project is complete or management decides to scrap the project due to unforeseen circumstances. After the completion of the deliverables implementation and testing will begin. Thus, the first increment is completely documentation-oriented.

Reviews and walkthroughs will be conducted on a biweekly basis. The purpose of these sessions will be to discuss problems to mitigate risks during the process.



6.2 Methods, Tools, and Techniques

Documents are delivered in Microsoft Word format, but edited using Google Docs. Google Docs allows for team members to edit simultaneously. When the current version of the document is finalized, it is downloaded from the Google servers where it can be distributed to clients in the Word format. Diagrams are drawn using the website draw.io. Diagrams are saved in case of future edits and converted to PNG format to be used in the documents. Document distribution is provided with the use of Google Docs. As for schedule planning and tracking, Microsoft Project is used as the most efficient way to plan and track the projects. Training will be done verbally with the use of practical examples as well as a programmer that is on the current team.

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.

6.3 Infrastructure Plan

The project will be controlled by Git. As such there is no need for people to be limited to work in any specific workspace. There will be regular meetings, either in person or online in order to discuss and analyze the product. Meeting places will be located within the New York Area that is quiet and can be accessed by all members of the team.

The server is stored online and is hosted by Chief Programmer. It can be accessed from anywhere. Upon deliverance, the server will either be moved to a premium server or the NYU servers.

All documents are stored on Google Docs marked with revision numbers while project files are stored on Github for easy access.

6.4 Product Acceptance and Migration Plans

The product acceptance and migration plans will be available in the next iteration of the document.

7. SUPPORTING PROCESSES PLANS

7.1 Configuration Management Plan

For configuration identification method, items that will be under configuration control will be given a unique ID number for tracking and placed into categories based on relevance and impact on the whole project. A change to a configuration control artifact will go through an evaluation process to identify the impact that the change will have on the entire project. If the requested change will benefit the project and deemed valuable, the configuration control item will be changed to the new proposed item.

Evaluation method - each member of the project team will evaluate all the proposed changes and identify whether the proposed change will produce desired outcomes.

Status accounting method - configuration control and implementation status of the most recent version of each configuration item will be documented and tracked.

Release management method - current version will be released when all of its configuration control items are updated, completed, and ready to be released.

Process for baselining work products:

1. The manager should label the baseline version.
2. The new baseline version should be announced to the team
3. Should be announced using email
4. Should identify if it is a new baseline version or an updated baseline version of existing configuration items

Process for logging and analysis of change requests:

1. A change request should be entered into a configuration management system where it will be pending approval.

2. The request will undergo an evaluation process by the team members to identify the need for the change.
3. When the request is approved or denied, it will update the status of the request from pending to either approved or denied.

7.2 Qualification (Verification and Validation) Plan

Verification and validation will be performed on the following aspects:

- Software architecture design
- Software requirements
- User interface design
- Database design

All of the software requirements will be verified the client's expectations and requirements in order to make sure that the system that is built will do what the clients require. The requirements are thoroughly traced and checked throughout the life of the project to ensure the project remains in line with the client's requirements.

Each team member is responsible for their own work as well as the works of each team member. The chief programmer will perform the final verification and validation checks. Inspections will be performed at regular intervals to make sure all project team members are up to date with the status of the project as well as for fault detection.

7.3 Documentation (Library) Plan

This section describes the documentation plan for the project's documentation deliverables. The names of the person with each role can be found in section 4.3.

Document	Preparer	Reviewer	Date for Baseline Version	Distribution List
Software Requirements	-Chief Programmer -Backup Programmer	-Chief Programmer -Backup Programmer	03/23/16	-Git repository -Google docs

Specification (SRS)	-Programming Secretary -Programmer	-Programming Secretary -Programmer		-Preparer -Reviewer -Approver
Software Project Management Plan (SPMP)	-Chief Programmer -Backup Programmer -Programming Secretary -Programmer	-Chief Programmer -Backup Programmer -Programming Secretary -Programmer	04/06/16	-Git repository -Google docs -Preparer -Reviewer -Approver
Software Analysis Specification (SAS)	-Chief Programmer -Backup Programmer -Programming Secretary -Programmer	-Chief Programmer -Backup Programmer -Programming Secretary -Programmer	04/11/16	-Git repository -Google docs -Preparer -Reviewer -Approver
Software Design Document (SDD)	-Chief Programmer -Backup Programmer -Programming Secretary -Programmer	-Chief Programmer -Backup Programmer -Programming Secretary -Programmer	04/25/16	-Git repository -Google docs -Preparer -Reviewer -Approver
Architectural Descriptions	-Chief Programmer -Backup Programmer	-Chief Programmer -Backup Programmer -Programming Secretary -Programmer	05/1/16	-Git repository -Google docs -Reviewer
Design Specifications	-Chief Programmer -Backup Programmer	-Chief Programmer -Backup Programmer -Programming Secretary -Programmer	05/1/16	-Git repository -Google docs -Reviewer
Interface Specifications	-Chief Programmer -Backup Programmer	-Chief Programmer -Backup Programmer -Programming Secretary -Programmer	05/1/16	-Git repository -Google docs -Reviewer
Test Plans	-Chief Programmer -Backup Programmer	-Chief Programmer -Backup Programmer	05/1/16	-Git repository -Google docs

		-Programming Secretary -Programmer		-Reviewer
Meeting Minutes	-Chief Programmer -Backup Programmer	-Chief Programmer -Backup Programmer -Programming Secretary -Programmer	24 hours after meeting	-Git repository -Google docs -Reviewer
Review Reports	-Chief Programmer -Backup Programmer	-Chief Programmer -Backup Programmer -Programming Secretary -Programmer	24 hours after review	-Git repository -Google docs -Reviewer

7.4 Quality Assurance Plan

After completing a document, an inspection will be held. Any faults that persist after completion will be identified and resolved by the members of the quality assurance team using a five-step process. 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.

7.5 Reviews and Audits

Review	Schedule	Resources	Methods	Procedures
NYU Tandon management reviews	Biweekly	NYU Tandon reviewer	-Review works in progress	- Review works in progress - Identify faults

Developer peer reviews	Biweekly	Team members	-Review works in progress -Report and document any problems	- Meet with team members and review works in progress - Identify faults and prompt fixes - Implement fixes and document changes
Technical reviews	Biweekly	Team members	-Review works in progress	- Meet with team members and review works in progress - Identify faults and prompt fixes - Implement fixes and document changes
Walkthroughs	Weekly	Team members	-Briefly review works in progress	- Meet with team members and review works in progress - Identify faults and prompt fixes - Implement fixes
Inspections	After completing a document	Team members	-Extensively review works in progress -Report and document any problems	- Meet with team members and review works in progress - Present copies of completed document to members of review team - Each team member reads document in detail - Identify faults and prompt fixes - Implement fixes and document changes - Review again to see if any faults persist
Audits	After completing a document	Auditors	-Review works in progress -Report and	- Meet with auditors and review works in progress - Identify faults and prompt fixes - Implement fixes and document

			document any problems	changes

7.6 Problem Resolution Plans

Depending on the type of problem, if it is non-technical, it will be reported to the team manager. Else, it will be reported to the chief programmer. After this problem is documented, the fault(s) causing this problem, document or code, will be inspected.

In the technical side, 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.

7.7 Environment Management Plans

To develop code for Sharit, access to the tools required are all freely available online. A simple text editor is required to create and modify all code. Use of other coding software is up to the discretion of the developers. The testing environment only requires a browser to access the website to test its functionality. To test the website's ability to function under load, code will be written that will create multiple connections accessing the site.

7.8 Process Improvement Plan

The process improvement plan will be available in the next iteration of the document.

8. ADDITIONAL PLANS

There are no additional plans at this time.

9. INDEX

10. RATIONALE

Sharit is created with the goal of an easier file sharing system that fosters 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.

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 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

Cumulative

Whom	Estimated	Actual	Difference
Allen Zheng	8 hr	9 hr	1 hr
Hui Huang	8 hr	14 hr	6 hr
Kenneth Liang	8 hr	14 hr	6 hr
Warlon Zeng	8 hr	13 hr	5 hr
Summary	32 hr	50 hr	18 hr

12.2 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

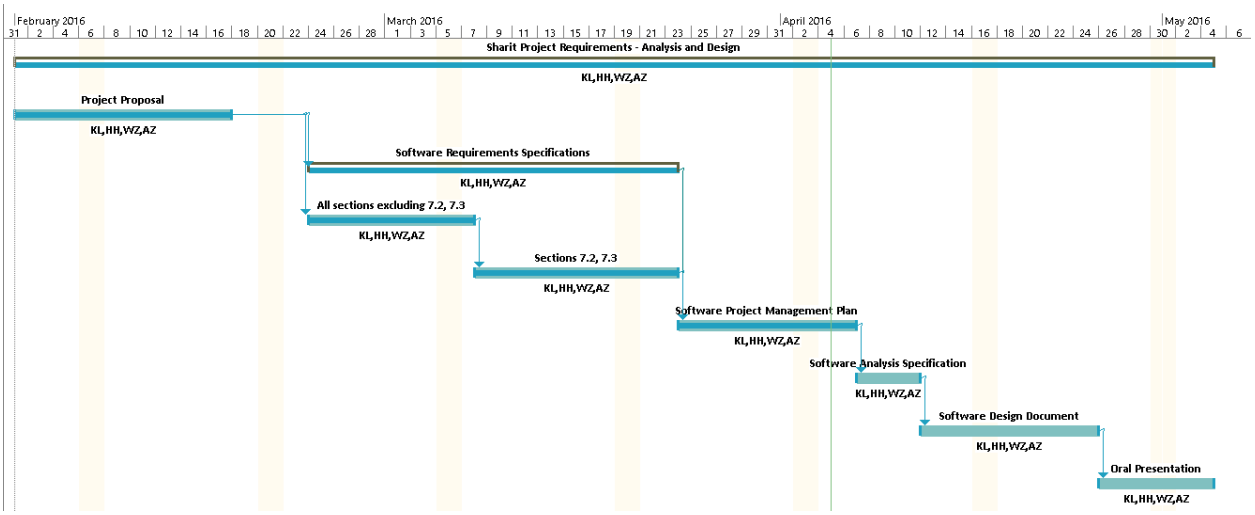
	Warlon Zeng	5	2	3
	Summary	20	11	9
SRS 2.0	Allen Zheng	4	2	2
	Hui Huang	4	5	1
	Kenneth Liang	4	3	1
	Warlon Zeng	4	2	2
	Summary	16	12	6
SPMP	Allen Zheng	5	4	1
	Hui Huang	5	5	0
	Kenneth Liang	5	6	1
	Warlon Zeng	5	3	2
	Summary	20	18	4

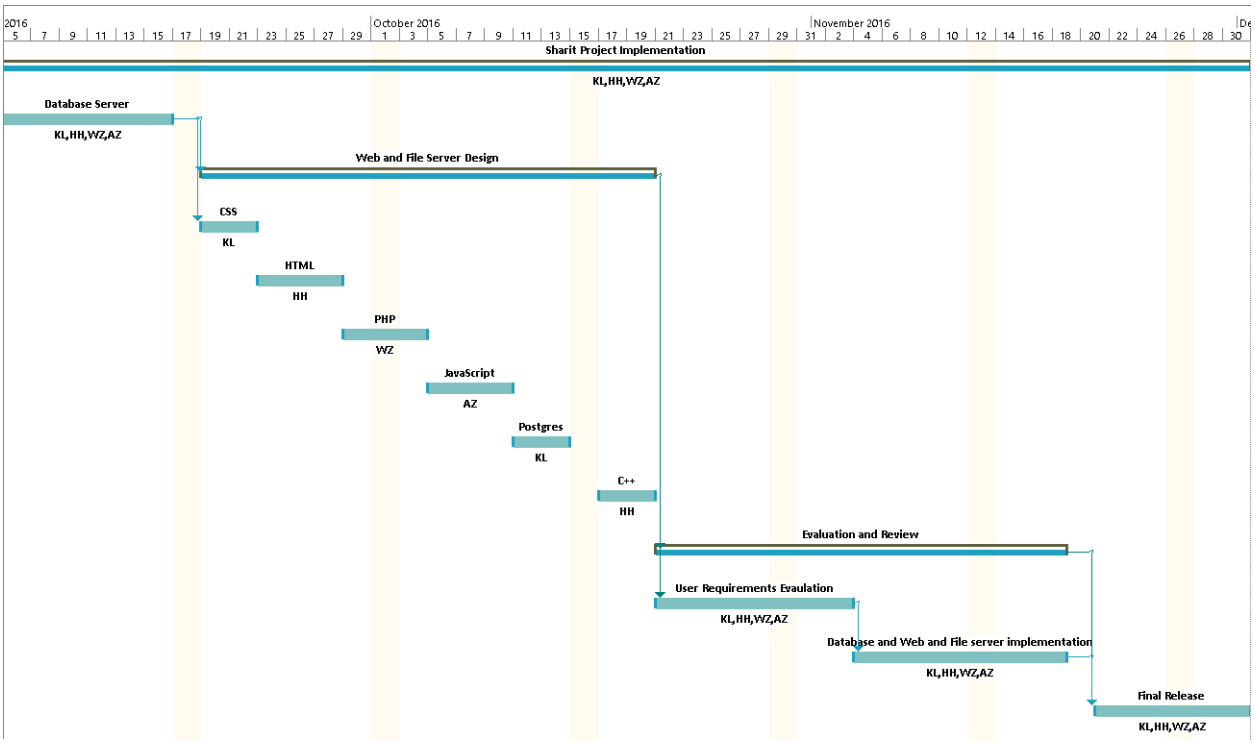
Cumulative

Whom	Estimated	Actual	Difference
Allen Zheng	14	9	5
Hui Huang	14	12	2
Kenneth Liang	14	13	1
Warlon Zeng	14	7	7
Summary	56	41	15

12.3 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			**Software Analysis Specification**	**3 days**	**Thu 4/7/16**	**Mon 4/11/16**	**KL,HH,WZ,AZ**
8			**Software Design Document**	**10 days**	**Tue 4/12/16**	**Mon 4/25/16**	**KL,HH,WZ,AZ**
9			**Oral Presentation**	**7 days**	**Tue 4/26/16**	**Wed 5/4/16**	**KL,HH,WZ,AZ**
10							
11			**Sharit Project Implementation**	**64 days**	**Mon 9/5/16**	**Thu 12/1/16**	**KL,HH,WZ,AZ**
12			**Database Server**	**10 days**	**Mon 9/5/16**	**Fri 9/16/16**	**KL,HH,WZ,AZ**
13			**Web and File Server Design**	**24 days**	**Mon 9/19/16**	**Thu 10/20/16**	
14			CSS	4 days	Mon 9/19/16	Thu 9/22/16	KL
15			HTML	4 days	Fri 9/23/16	Wed 9/28/16	HH
16			PHP	4 days	Thu 9/29/16	Tue 10/4/16	WZ
17			JavaScript	4 days	Wed 10/5/16	Mon 10/10/16	AZ
18			Postgres	4 days	Tue 10/11/16	Fri 10/14/16	KL
19			C++	4 days	Mon 10/17/16	Thu 10/20/16	HH
20			**Evaluation and Review**	**21 days**	**Fri 10/21/16**	**Fri 11/18/16**	
21			User Requirements Evaulation	10 days	Fri 10/21/16	Thu 11/3/16	**KL,HH,WZ,AZ**
22			Database and Web and File server	11 days	Fri 11/4/16	Fri 11/18/16	**KL,HH,WZ,AZ**
23			**Final Release**	**9 days**	**Mon 11/21/16**	**Thu 12/1/16**	**KL,HH,WZ,AZ**


February 2016
March 2016
April 2016
May 2016
Sharit Project Requirements - Analysis and Design
KL,HH,WZ,AZ
Project Proposal
KL,HH,WZ,AZ
Software Requirements Specifications
KL,HH,WZ,AZ
All sections excluding 7.2, 7.3
KL,HH,WZ,AZ
Sections 7.2, 7.3
KL,HH,WZ,AZ
Software Project Management Plan
KL,HH,WZ,AZ
Software Analysis Specification
KL,HH,WZ,AZ
Software Design Document
KL,HH,WZ,AZ
Oral Presentation
KL,HH,WZ,AZ


2016
October 2016
November 2016
Sharit Project Implementation
KL,HH,WZ,AZ
Database Server
KL,HH,WZ,AZ
Web and File Server Design
CSS
KL
HTML
HH
PHP
WZ
JavaScript
AZ
Postgres
KL
C++
HH
Evaluation and Review
User Requirements Evaulation
KL,HH,WZ,AZ
Database and Web and File server implementation
KL,HH,WZ,AZ
Final Release
KL,HH,WZ,AZ