

**Computer Science and Engineering**

**ItWorks**

**Requirements Analysis Specification**

**Version 1.0**

Document Number: RAS-001

Project Team Number: B28

Project Team Members:

|  |  |
| --- | --- |
| Rong Fang | rsf288 |
| Tian Lin | tcl344 |
| Lisa Frankel | lhf238 |

**REVIEW AND APPROVALS**

|  |  |  |  |
| --- | --- | --- | --- |
| **Printed Name and Title** | **Function (Author, Reviewer, Approval)** | **Date** | **Signature** |
| Tian Lin | Author | February 7, 2018 |  |
| Rong Fang | Author | February 7, 2018 |  |
| Lisa Frankel | Author | February 7, 2018 |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**REVISION LEVEL**

|  |  |  |
| --- | --- | --- |
| **Date** | **Revision Number** | **Purpose** |
| February 7, 2018 | Version 1.0 | Initial Release |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**TABLE OF CONTENTS**

1. **INTRODUCTION……………………………………………………………………………..…..……..6**
   1. PURPOSE
2. **SCOPE…………………………………………………………………………………………...……....6**
   1. IDENTIFICATION
   2. BOUNDS
   3. OBJECTIVES
   4. SYSTEM OVERVIEW
   5. DOCUMENT OVERVIEW
3. **REFERENCE DOCUMENT…………………………………………………………………….……...7**
4. **BUSINESS REQUIREMENTS…………………………………………………………………….…..7**
   1. TECHNOLOGY
   2. *ECONOMICS*
   3. *REGULATORY AND LEGAL*
   4. *MARKET CONSIDERATION*
   5. *RISKS AND ALTERNATIVES*
   6. *HUMAN RESOURCES AND TRAINING*
5. **LOGICAL ARCHITECTURAL SPECIFICATION……………………………………………..……..9**
   1. CONTEXT DIAGRAM
6. **FUNCTIONAL REQUIREMENTS ANALYSIS SPECIFICATION…………………………………9**
   1. FUNCTIONAL DESCRIPTIVE DETAILED REQUIREMENTS
   2. SYSTEM CAPABILITY REQUIREMENTS
      1. *Capabilities*
      2. *Use Case Diagrams*
      3. *Use Case Descriptions*
   3. USER INTERFACE REQUIREMENTS
   4. COMPONENT ARCHITECTURE
   5. CLASS DIAGRAMS
   6. CLASS RELATIONSHIP/INTERACTION DIAGRAMS
   7. EVENT SECTION
      1. *Event Dictionary*
      2. *Event Diagrams*
   8. ACTIVITY/STATE (SCENARIO) SECTION (TO BE COMPLETED IN DESIGN)
      1. *Activity (Scenario) Diagrams*
      2. *Activity (Scenario) Specification*
   9. SEQUENCE DIAGRAMS
   10. COLLABORATION DIAGRAMS
   11. DICTIONARIES
7. **NON-FUNCTIONAL/OPERATIONAL SPECIFICATIONS………………………………………..27**
8. **SYSTEM TEST PLAN REQUIREMENTS…………………………………………………………..28**
9. **QUALIFICATION PROVISIONS……………………………………………………………………..28**
10. **REQUIREMENTS TRACEABILITY………………………………………………………………..28**
11. **RATIONALE.. …………….…………….…………….………………….…………….…………….28**
12. **NOTES…………….……………………….…………….…………….….…………….…………….28**
13. **APPENDICES…………….…………….…………….…………….…………….…………….……29**
    1. DICTIONARIES
    2. UML DIAGRAMS, IFNOT INCLUDED IN THE BODY OF THE DOCUMENT
    3. SCHEDULE TRACKING
    4. DEFECT TRACKING

**1. INTRODUCTION**

**1.1 Purpose**

The purpose of this RAS is to provide a detailed description of the requirements and analysis of ItWorks. This RAS will explain the purpose, as well as scope, business requirements, context diagram, function requirements, non-functional requirements, system test plan requirements, qualification provisions, and requirements traceability for the project.

The intended audience of this RAS is the developers of the system, project management, corporate executives, and the SQA team.

**2. SCOPE**

**2.1 Identification**

Requirements Analysis Specification: Version 1.0, Document Number: RAS – 001

**2.2 Bounds**

System boundaries - The app itself enables users to post and see posts from others. Database system and server are external to the project. Boundaries includes user interfaces with application, and API used to access the external database system.

**2.3 Objectives**

Project Priority: Quality and on schedule

Type of delivery: evolutionary

Initial deliverable:

1. Project proposal - 1/31/18
2. Requirements Analysis Specification (RAS) – 2/7/18
3. Software Project Management Plan (SPMP) - 2/14/18
4. Software Design Document (SDD) – 3/7/18
5. Software Design Document Final (SDD) – 4/25/18

**2.4 System Overview**

ItWorks is a mobile/web based application that values user input. When conventional solutions don’t work, there is no good way to seek alternative solutions. Users can post advice and search for answers. ItWorks will not only enable its users to look for answers, but also use a feedback system that lets users know if the answers they find are effective. Additionally, there will be a reputation system, in which other users vote on whether a piece of advice is useful or not. This makes the application more engaging.

It is a crowdsourcing platform that allows users to access tried-and-tested, yet unconventional methods of solving problems that come about in day-to-day life.

**2.5 Document Overview**

* Section three will contain reference documents.
* Section four will contain business requirements and explain the technology, economic, legal drivers, market considerations. The value that is provided for the person who proposed the project.
* Section five will contain logical architectural specifications, including a context diagram, user interface requirements, component architecture, class diagrams, class relationship/interaction diagrams, events, behavior, and dictionaries.
* Section six will contain non functional/operational specifications, which includes relevant requirements and constraints.
* Section seven contains the system test plan requirements.
* Section eight contains the qualification provisions.
* Section nine contains the requirements traceability.
* Section ten contains rationale.
* Section eleven contains notes.
* Section twelve is the appendix which contains our dictionaries, UML diagrams, and schedule / defect tracking.

**3. REFERENCE DOCUMENTS**

Team A28, ItWorks Project, V 1.0, 9/21/2017

Team A28, ItWorks Initial SRS, V 001, 10/17/2017

Team A28, ItWorks Final SRS, V 002, 10/28/2017

Team A28, ItWorks SPMP, V 1.0, 10/14/2017

Team A28, ItWorks SAS, V 1.0, 11/21/2017

**4. Business Requirements**

**4.1 Technology**

The popularity and convenience of mobile apps is a large driver in our decision to make the app mobile. ItWorks builds a mobile/web community interface to connects supplies of solutions to people who need them anytime anywhere.

**4.2 Economics**

The earning revenue from ads on an app is cost efficient. The app utilizes most relevant user search terms to match sponsored ads to generate the most revenue possible.

**4.3 Regulatory and Legal**

None

**4.4 Market Considerations**

Everyone who can use a device that can run apps are a potential market source. The app is built to be crowd sourced by the people who have solutions and for the people who is looking for solution.

**4.5 Risks and Alternatives**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No. | Title | Estimated likelihood of occurring (L:1-10 with 1 lowest likelihood | Estimated impact (I: 1-10 with 1 lowest impact) | Estimated cost of managing (M:1-10 with 1 lowest cost) | Priority number (lowest number handled first) (11-L) \*(11-I) \*M | Retirement plan | Responsible person | Target completion date |
| 1 | Database is corrupted or down | 2 | 9 | 4 | 72 | See Note 1 | Rong | 5/1/2018 |
| 2 | Team member unable to meet schedule due to emergencies | 8 | 5 | 3 | 54 | See Note 2 | Lisa | Continual |
| 3 | Unable to handle amount of users because of incorrect size estimate | 6 | 8 | 3 | 45 | See Note 3 | Tian | 5/1/2018 |
| 4 | Product deliverables produced behind schedule | 5 | 10 | 6 | 36 | See Note 4 | Lisa | Continual |
| 5 | Product does not meet customer requirement | 3 | 10 | 5 | 40 | See Note 5 | Rong | Continual |

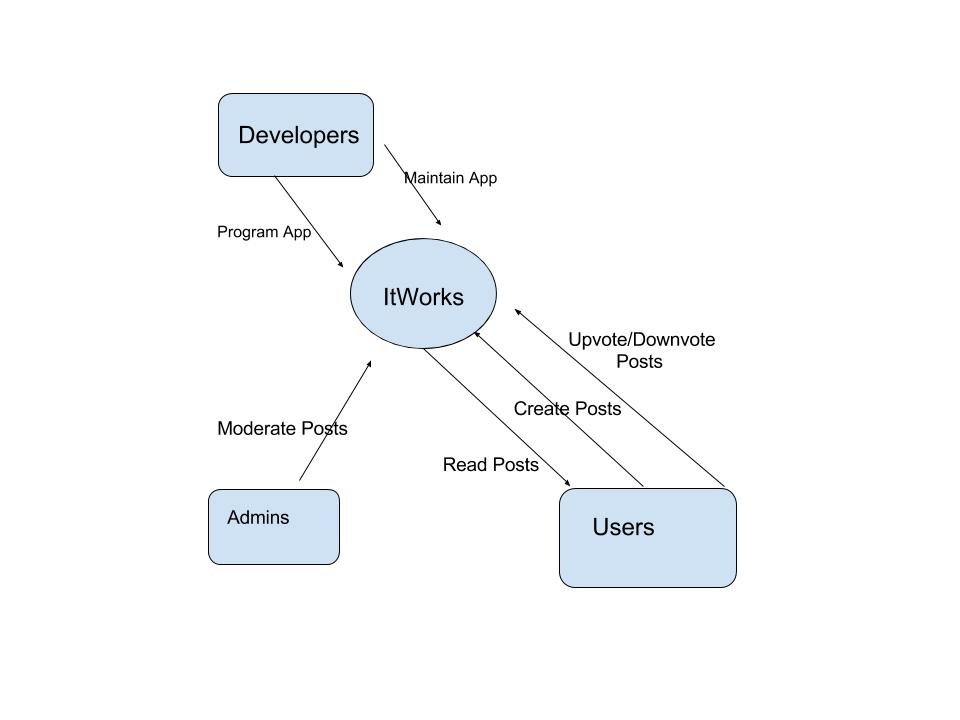
1. Insure our data will be stored redundantly.
2. Reschedule to deliver on time.
3. Have plan in place for expanding server and bandwidth so that it's easy to quickly retire risk.
4. Track metrics on progress so that product manager may intervene before it’s too late.
5. Agile process ensures constant communication with customer to ensure product meets their requirements.

**4.6 Human Resources and Training**

Business (project) resources and training required.

**5. Logical Architectural Specification**

**5.1 Context Diagram**



**6. Functional Requirements Analysis Specification**

**6.1 Function Descriptive Detailed Requirements**

**6.2 System Capabilities Requirements**

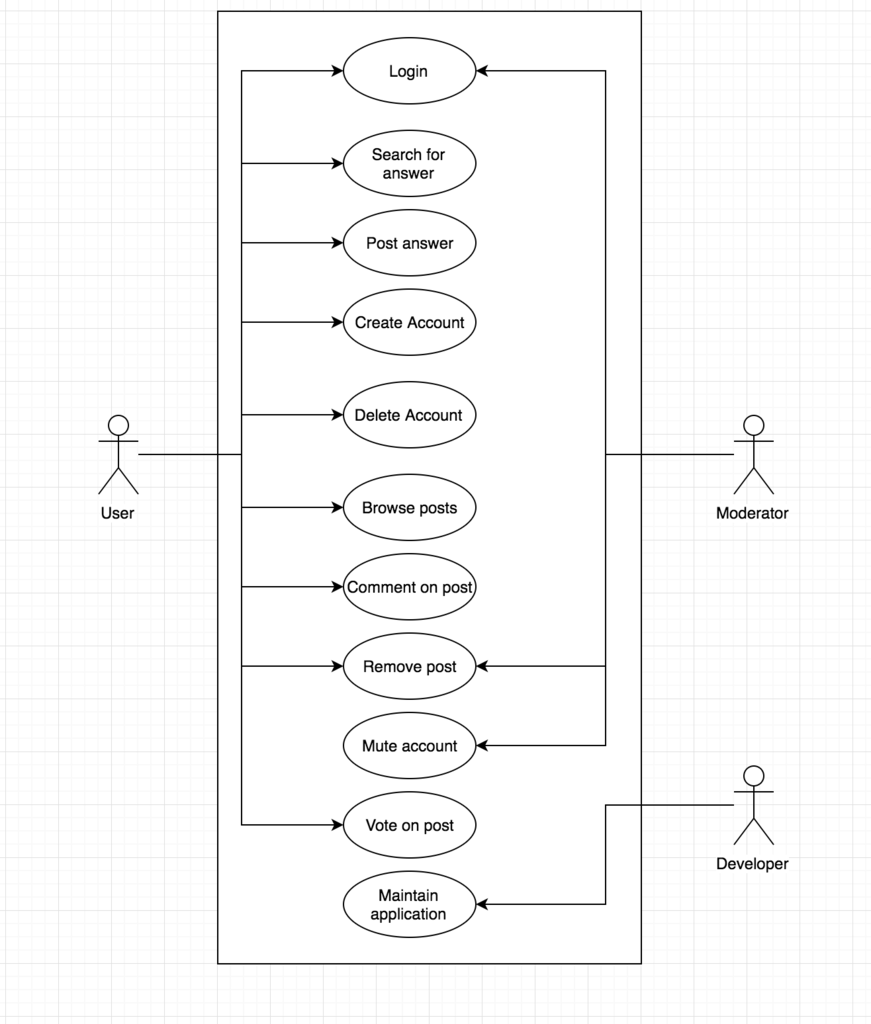
**6.2.1 Capabilities**

|  |  |
| --- | --- |
| **Use Cases** | **Description** |
| Login | User logs into their account |
| Search for answer | User searches for answer through the application |
| Post answer | User posts a solution to a problem |
| Create account | User creates an account |
| Delete account | User deletes account and all associated data |
| Browsing posts | Search through all existing posts |
| Comment on post | User comments on a post |
| Remove post | Remove a post from the site |
| Mute account | Users with frequent flagged posts will only be allow to read posts |
| Vote on post | User up-votes or down-votes post depending on how good they think the solution is |

**6.2.2 Use Case Diagrams**

Provided in this section is the use case diagram for ItWorks, as well as a list of the actors and the use cases involved.

Use case diagram:



Actors

* User
* Moderator
* Developer

**6.2.3 Use Case Descriptions**

|  |  |  |
| --- | --- | --- |
| Login | | |
| Description | User logs into their account | |
| Pre-conditions | User or moderator is not already logged in | |
| Flows | Basic or Normal Flows | Successful login:   1. Go to login page → 2. Serve login page ← 3. Enter name and password → 4. Submit form → 5. Log user in ← 6. Redirect to main page ← |
| Alternative Flows | Unsuccessful login:   1. Go to login page → 2. Serve login page ← 3. Enter name and password → 4. Submit form → 5. Report unsuccessful login → 6. Return to step 3 → |
| Post conditions | User or moderator is logged in | |
| Special Requirements | None | |
| Extension Points | None | |

|  |  |  |
| --- | --- | --- |
| Search for answer | | |
| Description | User searches for answer through the application | |
| Pre-conditions | User is Logged in | |
| Flows | Basic or Normal Flows | 1. Go to search page → 2. Serve search page ← 3. Input question into search bar on page → 4. Submit form → 5. Load relevant data ← |
| Alternative Flows |  |
| Post conditions | User is given search results | |
| Special Requirements | None | |
| Extension Points | None | |

|  |  |  |
| --- | --- | --- |
| Post answer | | |
| Description | User posts a solution to a problem | |
| Pre-conditions | User is logged in | |
| Flows | Basic or Normal Flows | 1. Go to post page → 2. Serve post page ← 3. Input solution into post form on page → 4. Submit form → 5. Post stored in database ← 6. Post displayed on post page ← |
| Alternative Flows |  |
| Post conditions | The solution is posted to the problem | |
| Special Requirements | None | |
| Extension Points | None | |

|  |  |  |
| --- | --- | --- |
| Create account | | |
| Description | User creates an account | |
| Pre-conditions | User does not have an account | |
| Flows | Basic or Normal Flows | 1. Go to landing page → 2. Click create account → 3. Server create account page ← 4. Enter username, name, password, and phone number into form → 5. Submit form → 6. Receive text confirmation code ← 7. Request text confirmation code ← 8. Enter text confirmation code → |
| Alternative Flows | 1. Go to landing page → 2. Click create account → 3. Serve create account page ← 4. Enter name, password, and phone number into form → 5. Submit form → 6. Inform username or phone number already associated with account ← 7. Repeat step 4 → |
| Post conditions | User has a functional account | |
| Special Requirements | None | |
| Extension Points | None | |

|  |  |  |
| --- | --- | --- |
| Delete account | | |
| Description | User deletes account and all associated data | |
| Pre-conditions | User is logged in | |
| Flows | Basic or Normal Flows | 1. Go to delete account page → 2. Serve delete account page ← 3. Select delete account button → 4. Submit form → 5. Delete user account and all associated data ← 6. Redirect to landing page ← |
| Alternative Flows |  |
| Post conditions | User’s account is deleted | |
| Special Requirements | None | |
| Extension Points | None | |

|  |  |  |
| --- | --- | --- |
| Browsing posts | | |
| Description | Search through all existing posts | |
| Pre-conditions | User is logged in | |
| Flows | Basic or Normal Flows | 1. Go to browse page → 2. Serve browse page ← 3. Enter browse keywords in form → 4. Submit form → 5. Load relevant data ← |
| Alternative Flows |  |
| Post conditions | User is landed on posts he/she browse for | |
| Special Requirements | None | |
| Extension Points | None | |

|  |  |  |
| --- | --- | --- |
| Comment on post | | |
| Description | User comments on a post | |
| Pre-conditions | User is logged in and looking at the post | |
| Flows | Basic or Normal Flows | 1. Select “comment on post” → 2. Input comment string → 3. Submit form → 4. Store comment in database ← 5. Load comment onto post ← |
| Alternative Flows |  |
| Post conditions | Comment appears on post | |
| Special Requirements | None | |
| Extension Points | None | |

|  |  |  |
| --- | --- | --- |
| Remove post | | |
| Description | Remove a post from the site | |
| Pre-conditions | User or moderator is logged in | |
| Flows | Basic or Normal Flows | User removes post:   1. Select remove this post → 2. Remove post from database ← 3. Stop displaying post ← |
| Alternative Flows | Moderator removes post:   1. Look at flagged posts → 2. Determine post needs to be removed → 3. Select remove this post → 4. Remove post from database ← 5. Stop displaying post ← |
| Post conditions | Post is removed from database | |
| Special Requirements | None | |
| Extension Points | None | |

|  |  |  |
| --- | --- | --- |
| Mute account | | |
| Description | Users with frequent flagged posts will only be allow to read posts | |
| Pre-conditions | Moderator is logged in | |
| Flows | Basic or Normal Flows | 1. Look for flagged users → 2. Review past user activities ← 3. Select the flagged user → 4. Edit account privilege to the applied user account to read post only → 5. Account database gets updated → |
| Alternative Flows |  |
| Post conditions | The selected user will be disallowed from posting, voting and commenting on posts | |
| Special Requirements | None | |
| Extension Points | None | |

|  |  |  |
| --- | --- | --- |
| Vote on post | | |
| Description | User up-votes or down-votes post depending on how good they think the solution is | |
| Pre-conditions | User is logged in and looking at the post | |
| Flows | Basic or Normal Flows | 1. Click up or down arrow next to post → 2. Show effect of up or down vote ← |
| Alternative Flows |  |
| Post conditions | Vote affects post’s score | |
| Special Requirements | None | |
| Extension Points | None | |

|  |  |  |
| --- | --- | --- |
| Maintain application | | |
| Description | Developer maintains application, handles any errors and updates site | |
| Pre-conditions | Developer is authorized to work on application | |
| Flows | Basic or Normal Flows | 1. Understand error or update → 2. Implement bug fix or update → 3. Push code → |
| Alternative Flows |  |
| Post conditions | Developer implements change | |
| Special Requirements | None | |
| Extension Points | None | |

**6.3 User Interface Requirements**

Prototyping is not part of the life cycle model used. A user needs access to a browser as this is a web application. This can be accessed on any PC or smart mobile device.

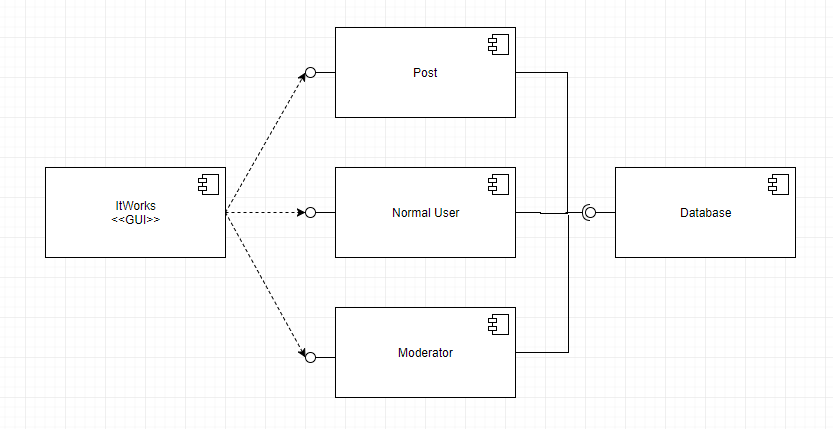
**6.4 Component (Component/Package/Subsystem) Architecture**

**6.4.1 Component Descriptions**

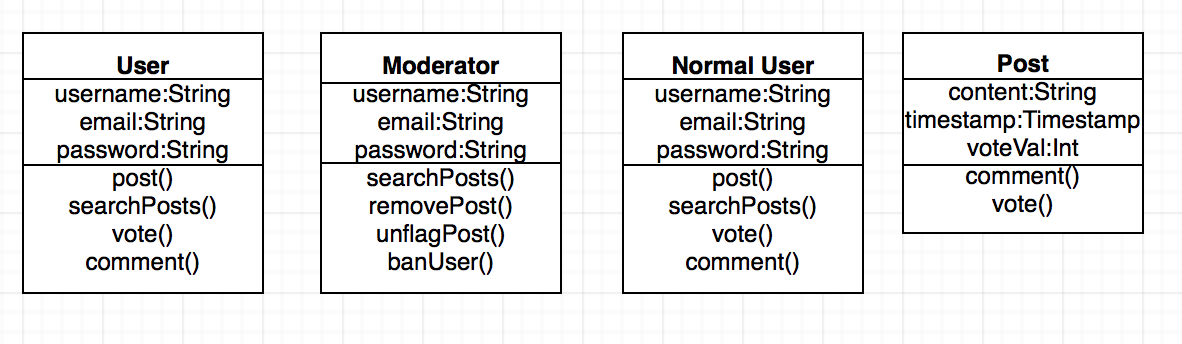
* Post
* User
* Database
* ItWorks GUI

**6.4.2 Component Architecture Diagram**

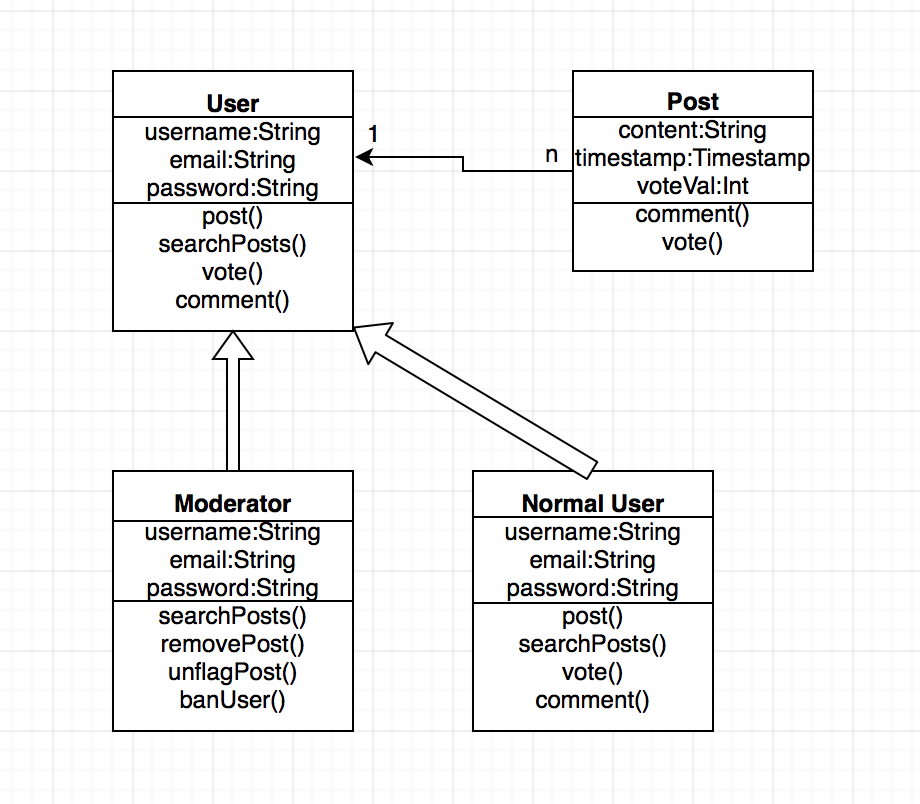
*Component Architecture Diagram*



**6.5 Class Diagrams**



**6.6 Class Relationship / Interaction Diagrams**



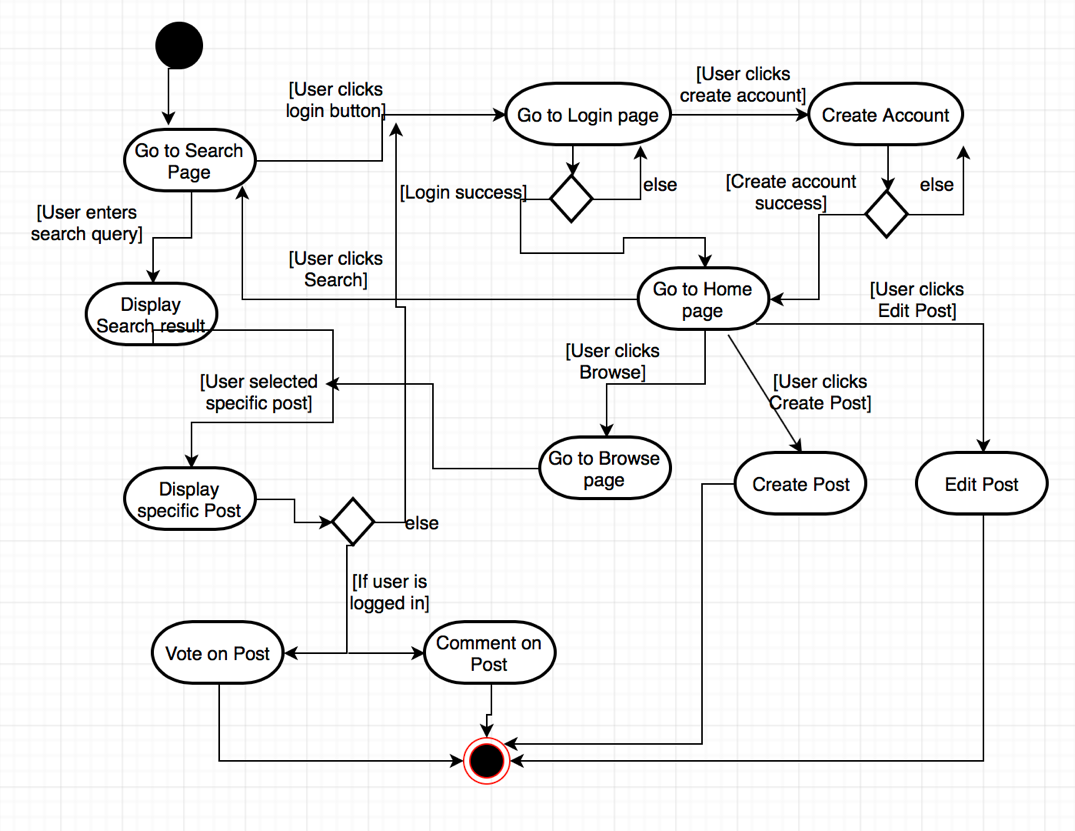
**6.7 Events**

* Login
  + Go to login page
  + Enter name and password
  + Submit form
* Search
  + Go to search page
  + Input question into search bar on page
  + Submit form
* Post
  + Go to post page
  + Input solution into post form on page
  + Submit form
* Create Account
  + Go to landing page
  + Click create account
  + Enter username, name, password, and phone number into form
  + Submit form
  + Enter text confirmation code
* Delete Account
  + Go to delete account page
  + Select delete account button
  + Submit form
* Browse
  + Go to browse page
  + Enter browse keywords in form
  + Submit form
* Comment
  + Select “comment on post”
  + Input comment string
  + Submit form
* Remove Post
  + Select remove this post
* Mute Account
  + Look for flagged users
  + Select the flagged user
  + Edit account privilege to the applied user account to read post only
* Vote on Post
  + Click up or down arrow next to post
* Maintain App
  + Understand error or update
  + Implement bug fix or update
  + Push code
    1. **Event Dictionary**

The key event dictionary table can be found in section 13.1 of the RAS.

**6.7.2 Event Diagrams**

*Event Diagram*



**6.8 Activity / State (Scenario) Section**

A scenario (work flow) diagram will be provided for each scenario. The scenarios are:

* Login
* Logout
* Create Account
* Delete Account
* Vote on Post
* Create Post
* Edit Post
* Comment
* Search
* Browse
* Mute User
* Flag post
* Unflag post

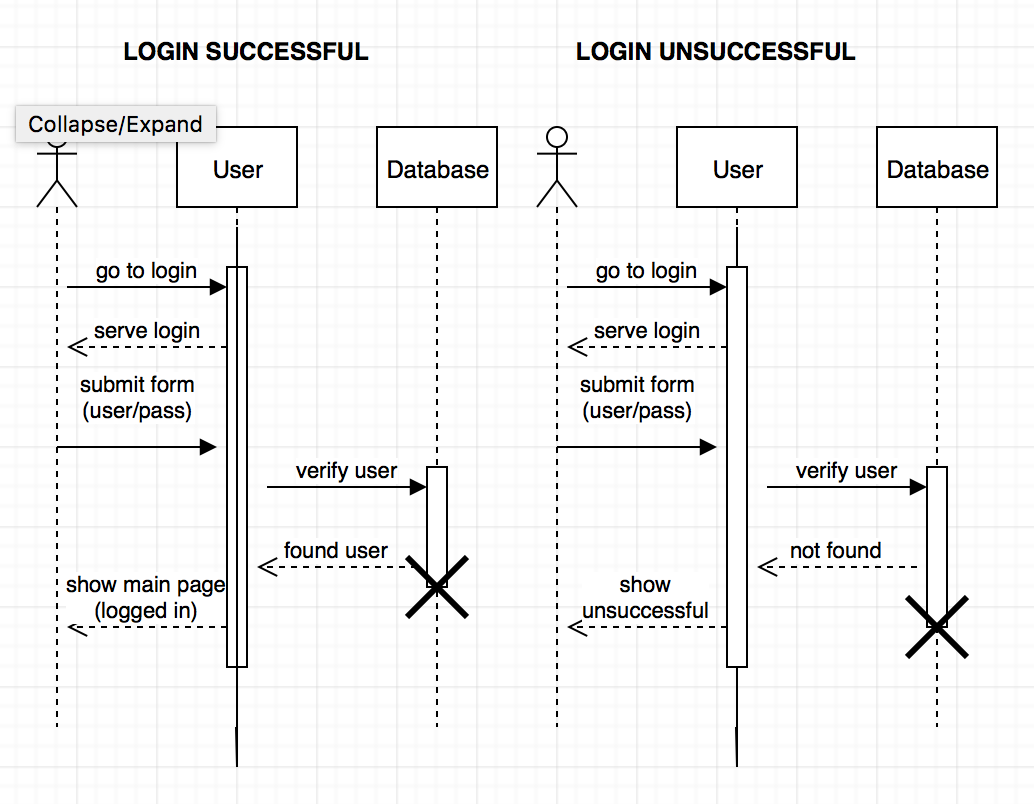
This section will be completed in design.

**6.8.1 Activity (Scenario) Diagrams**

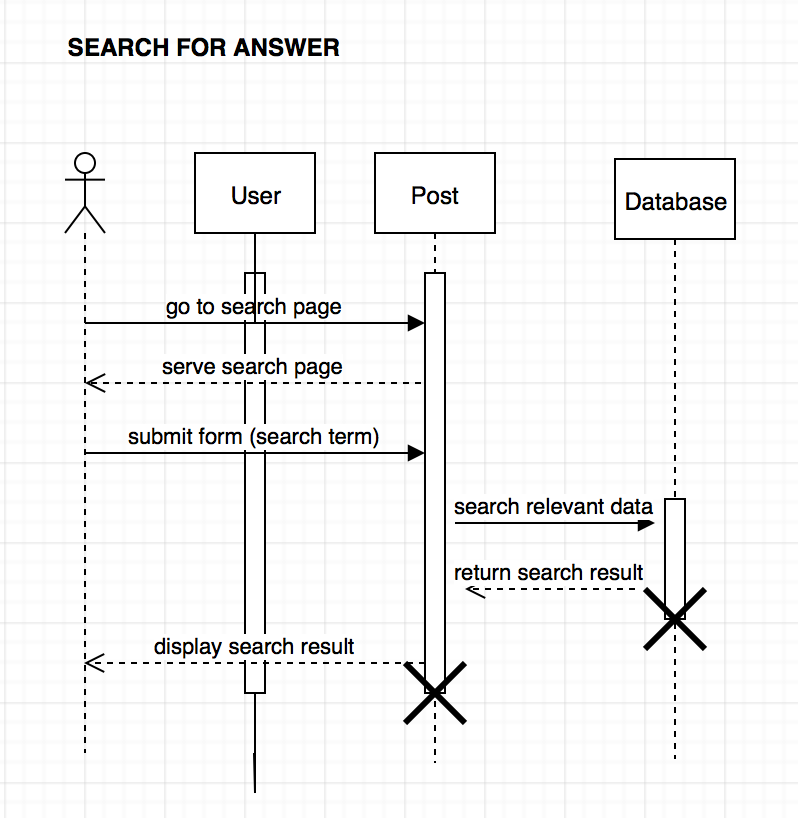
**6.8.2 Activity (Scenario) Specification**

**6.9 Sequence Diagrams**

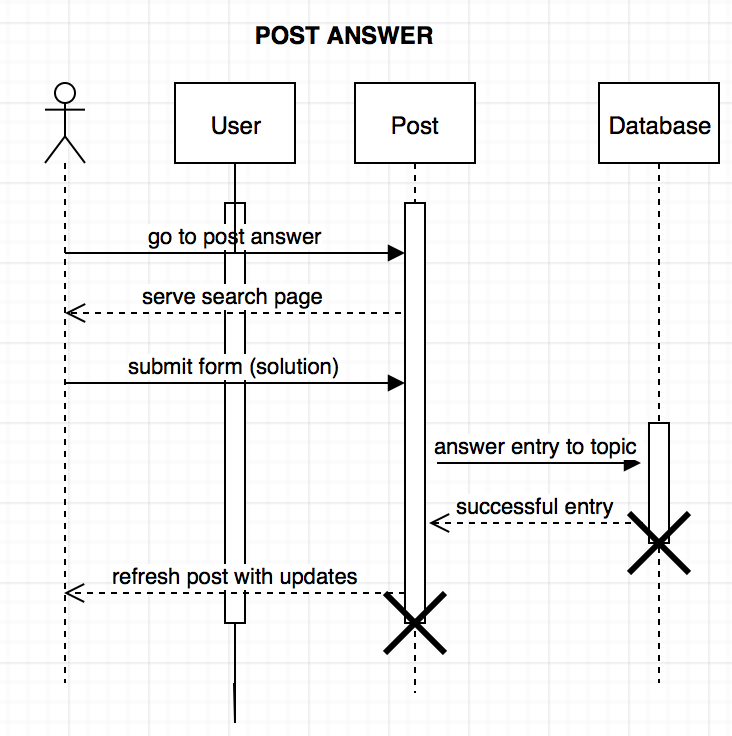
*Login Sequence (successful and unsuccessful)*



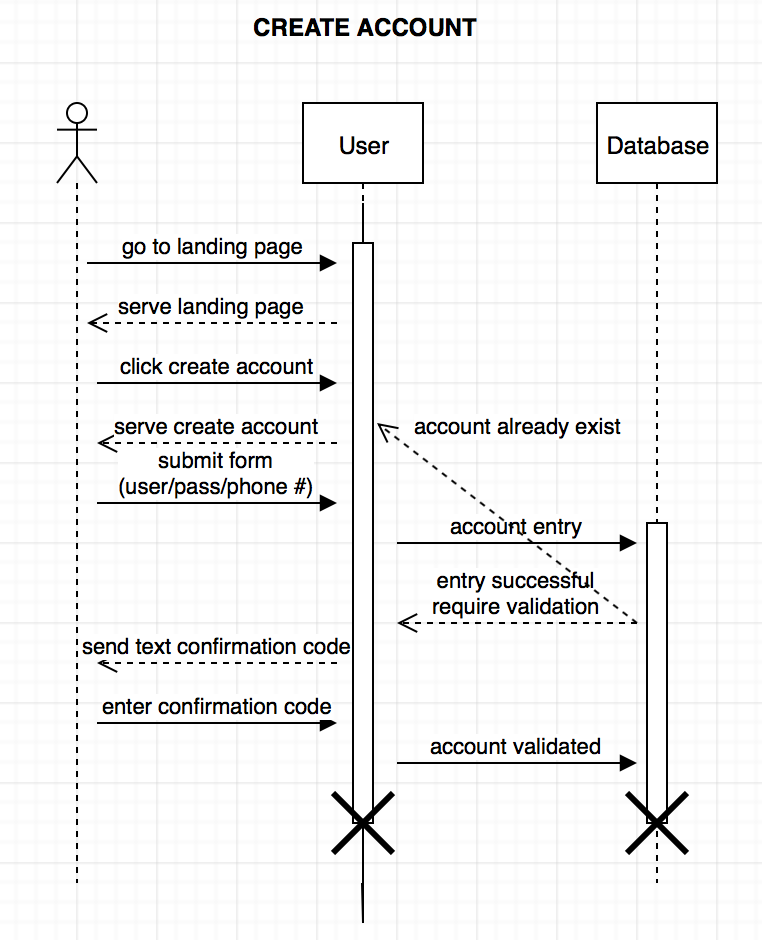
*Search Sequence*

**

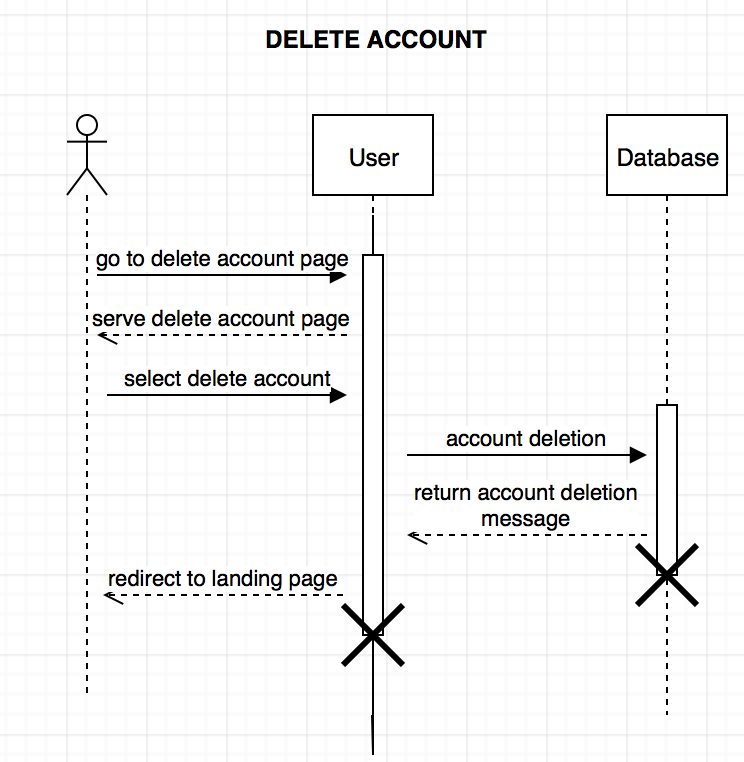
*Post Sequence*

**

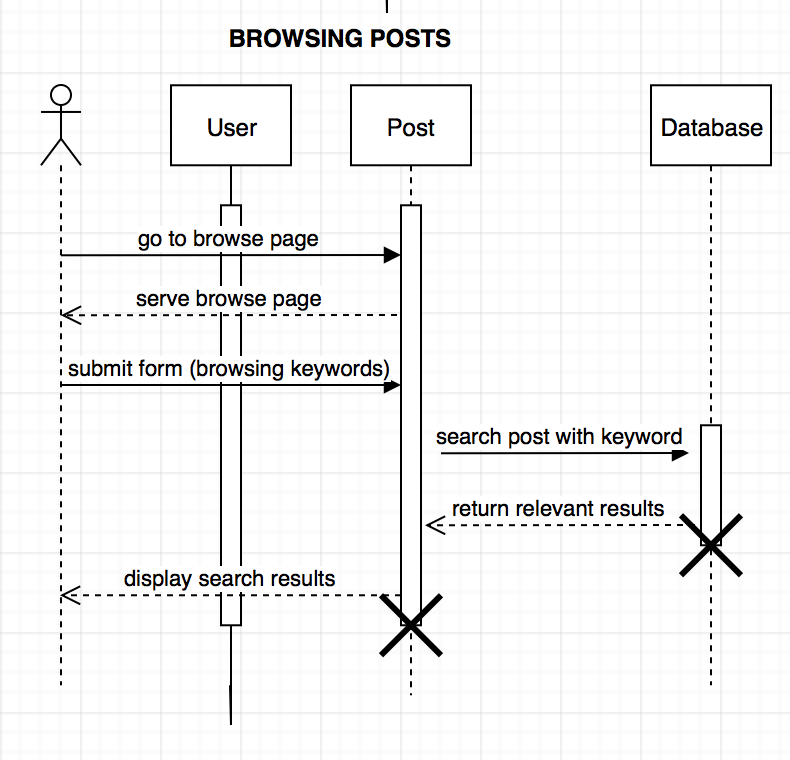
*Create Account Sequence*

**

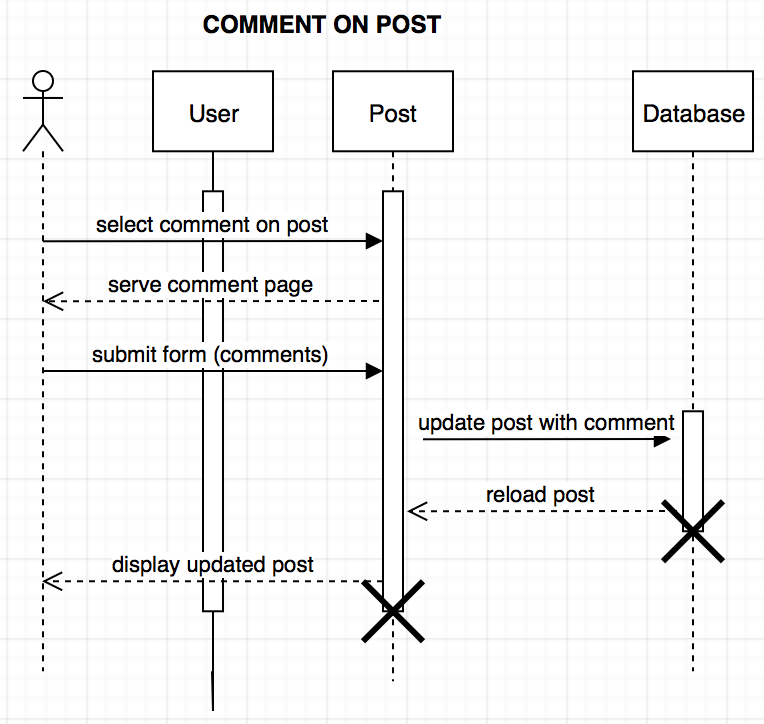
*Delete Account Sequence*

**

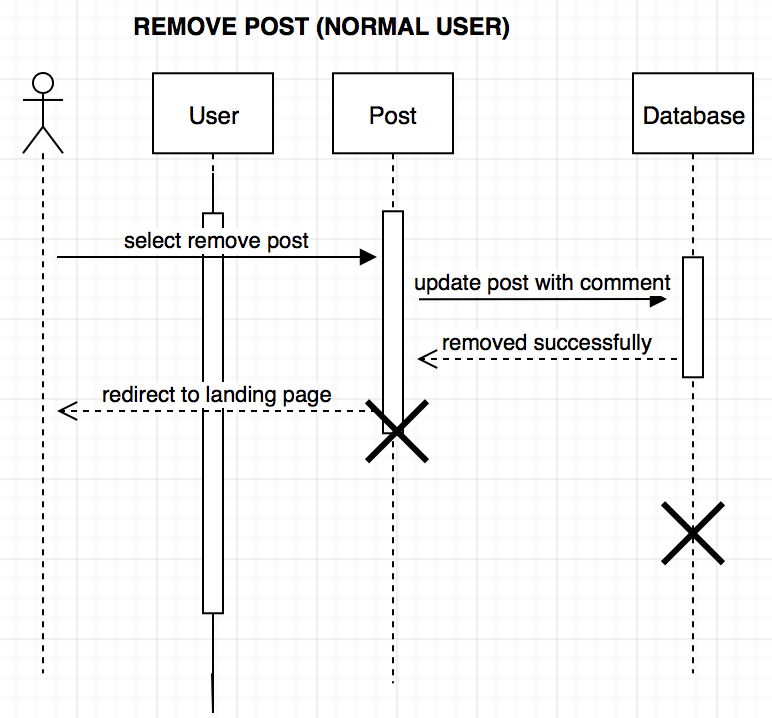
*Browse Sequence*

**

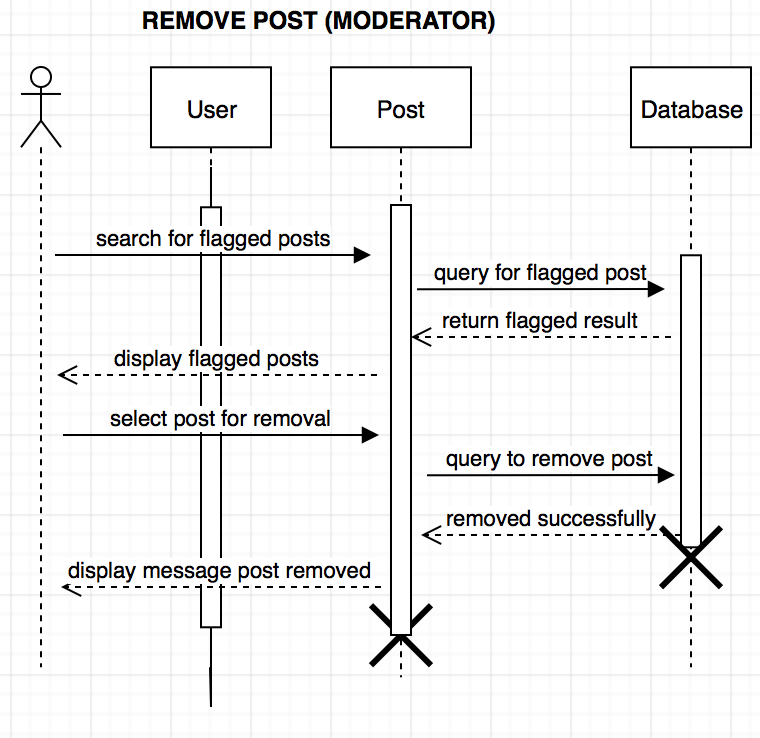
*Comment Sequence*

**

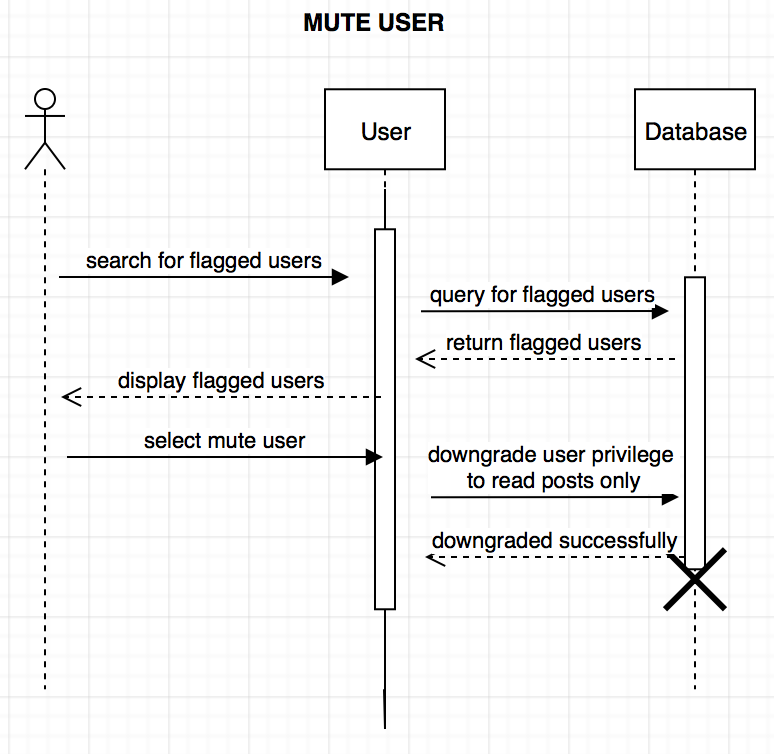
*Remove Post (Normal User) Sequence*

**

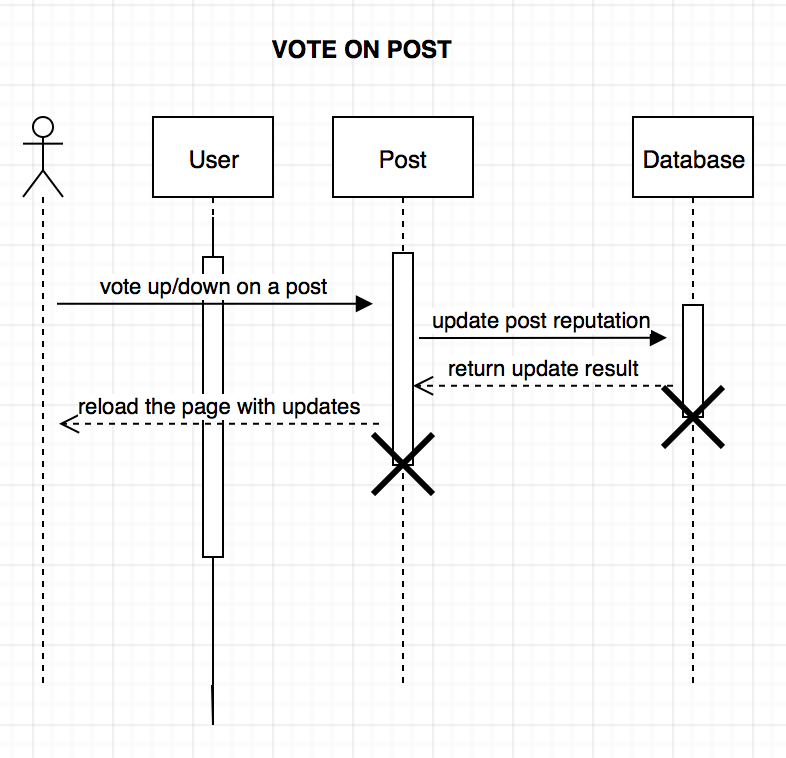
*Remove Post (Moderator) Sequence*

**

*Mute User Sequence*

**

*Vote Sequence*

**

*Maintain Application Sequence*

**6.10 Collaboration Diagrams**

This section will be completed in design.

**6.11 Dictionaries**

The following dictionaries can be found in section 12.1: Classes Dictionary Table, Methods Dictionary Table, Attributes Dictionary Table, Relationship Dictionary Table, and Key Events Dictionary Table.

**7. Non functional / operation specifications**  
  
**7.1 System External Interface Requirements**

**7.2 Safety Requirements  
7.3 Security and Privacy Requirements  
7.4 System Environment Requirements**

**7.5 Computer Resource Requirements**

***7.5.1 Computer Hardware Requirements  
7.5.2 Computer Hardware Resource Requirements***

***7.5.3 Computer Software Requirements  
7.5.4 Computer Communications Requirements***

**7.6 System Quality Factors  
7.7 Design and Construction Constraints  
7.8 Personnel-Related Requirements  
7.9 Training-Related Requirements**

Team members should be taking or have completed a software engineering course at NYU Tandon. All team members who lack programming knowledge in C++ or Go shall spend two week learning the language they need to code in. **7.10 Logistics-Related Requirements  
7.11 Packaging Requirements  
7.12 Precedence and Criticality Requirements**

**7.13 Other Requirements**

**8. SYSTEM TEST PLAN REQUIREMENTS**

The development team will generate inputs to test the system. These inputs will consist of users doing certain actions that will test each our use cases. We have no required simulators. Please refer to section 5.2 to see the use cases that will be tested.

**9. QUALIFICATION PROVISIONS**

There will be various stages of review. There will be self checks, peer review, walkthroughs and inspections to review this document. This document will be reviewed for quality based on the following metrics:

* Correct
* Unambiguous
* Complete
* Consistent
* Stable
* verifiable
* Modifiable (malleable)
* traceable

**10. REQUIREMENTS TRACEABILITY**

Each requirement will be traceable forwards and backwards. In documents, there will be section numbers so that it can be traced in the future. When we code any object we will have in the header, the requirement that it supports, so that it can be traced back to the requirements.

**11. RATIONALE**

**12. NOTES**

**13. Appendix**

**13.1 Dictionaries**

*Classes Dictionary Table*

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Description | Methods | Attributes |
| User | Abstract class for a user of the system | searchPosts()  post()  vote()  comment() | username:String  email:String  password:String |
| Moderator | Subclass of user, moderator is a user of the system with special privileges | searchPosts()  removePost()  unflagPost()  banUser() | username:String  email:String  password:String |
| Normal User | Subclass of user, normal user is a standard user of the system | searchPosts()  post()  vote()  comment() | username:String  email:String  password:String |
| Post | Class for a post | comment()  vote() | content:String  timestamp:Timestamp  voteVal:int |

*Methods Dictionary Table*

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Description | Class | Arguments |
| searchPosts() | Allows user to search through posts in the system that match a keyword or phrase | User, Moderator, Regular User | query:String |
| post() | Allows a user to post content to the site | User, Regular User | content:String  username:String |
| vote() | Allows a user to either up vote or down vote a post | User, Regular User | UpOrDown:bool  post:Post |
| comment() | Allows a user to comment on a post | User, Regular User | comment:String  post:Post |
| removePost() | Allows a moderator to remove a post | Moderator | post:Post |
| unflagPost() | Allows a moderator to unflag a post | Moderator | post:Post |
| banUser() | Allows a moderator to ban a user | Moderator | user:User |

*Attribute Dictionary Table*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Description | Simple/Complex | Type | Size | R/W |
| email | User’s email address | Simple | string | variable |  |
| password | User’s password | Simple | string | variable |  |
| username | User’s username | Simple | string | variable |  |
| content | Post content | Simple | string | variable |  |
| timestamp | A timestamp for a post | Simple | Timestamp | fixed |  |
| voteVal | A post’s score (based on votes) | Simple | int | variable |  |

*Relationship Dictionary Table*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Description | From Class | To class | Optional/Mandatory | Cardinality |
| OwnedBy | Each post is owned by a user | User | Post | Mandatory | A user can have many posts, and a post must have 1 and only 1 owner(user) |

*Key Events Dictionary Table*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Name | Description | Motive | Action | Pre-Conditions | Post-Conditions | State Change |
| Login | user logs into their account | login | Go to login page  Enter name and password  Submit form | User is not logged in | User is logged in | Logged In |
| Search | user searches for answer through the application | Search for answer | Go to search page  Input question into search bar on page  Submit form | User is logged in | User gets results | None |
| Post | user posts a solution to a problem | Post answer | Go to post page  Input solution into post form on page  Submit form | User is logged in | User’s post is added to site | None |
| Create Account | User creates an account | Create an account | Go to landing page  Click create account  Enter username, name, password, and phone number into form  Submit form  Enter text confirmation code | User does not have account | User has an account | Redirected to login page |
| Delete Account | User deletes account and all associated data | Delete an account | Go to delete account page  Select delete account button  Submit form | User has an account | User deletes their account | Redirected to login page |
| Browse | Search through all existing posts | Browse through posts | Go to browse page  Enter browse keywords in form  Submit form | User is logged in | Content appears | None |
| Comment | User comments on a post | Comment on a post | Select “comment on post”  Input comment string  Submit form | User is logged in | User’s comment appears under post | None |
| Remove Post | Remove a post from the site | Remove a post | Select remove this post | User is logged in and post they want to remove is their own OR user is a moderator | Post is removed from the site | None |
| Mute Account | Users with frequent flagged posts will only be allowed to read posts | Mute an account | Look for flagged users  Select the flagged user  Edit account privilege to the applied user account to read post only | User has been flagged | User can no longer post | None |
| Vote on post | User up-votes or down-votes post depending on how good they think the solution is | Vote on a post | Click up or down arrow next to post | User is logged in and viewing a post | User has altered the posts vote value | None |
| Maintain App | Developer maintains application, handles any errors and updates site | Maintain the application | Understand error or update  Implement bug fix or update  Push code | There is an update, error or bug. | New code is implemented | Dependent on code that is implemented |

**13.2 UML Diagrams**

Identified in body documents.

**13.3 Schedule Tracking**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or  Deliverable | Who | Estimated (hrs) | Actual (hrs) | Difference (hrs) |
| Initial SRS | Tian Lin | 12 hrs | 10 hrs | 2 less hrs |
|  | Rong Fang | 12 hrs | 10 hrs | 2 less hrs |
|  | Alex Huang | 12 hrs | 9 hrs | 3 less hrs |
|  | Lisa Frankel | 12 hrs | 9 hrs | 3 less hrs |
|  | Summary for entire team | 48 hrs | 38 hrs | 10 less hrs |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or  Deliverable | Who | Estimated (hrs) | Actual (hrs) | Difference (hrs) |
| Final SRS | Tian Lin | 2 hrs | 2 hrs | 0 hrs |
|  | Rong Fang | 2 hrs | 3 hrs | 1 more hrs |
|  | Alex Huang | 2 hrs | 0 hrs | 2 less hrs |
|  | Lisa Frankel | 2 hrs | 3 hrs | 1 more hrs |
|  | Summary for entire team | 8 hrs | 8 hrs | 0 hrs |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who | Estimated (hrs) | Actual (hrs) | Difference (hrs) |
| SPMP | Tian Lin | 12 | 11 | 1 less |
|  | Rong Fang | 12 | 10 | 2 less |
|  | Alex Huang | 12 | 9 | 3 less |
|  | Lisa Frankel | 12 | 10 | 2 less |
|  | Summary for entire team | 48 | 40 | 8 less |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or  Deliverable | Who | Estimated (hrs) | Actual (hrs) | Difference (hrs) |
| SAS | Tian Lin | 12 hrs | 10 hrs | 2 less hrs |
|  | Rong Fang | 12 hrs | 10 hrs | 2 less hrs |
|  | Alex Huang | 12 hrs | 10 hrs | 2 less hrs |
|  | Lisa Frankel | 12 hrs | 10 hrs | 2 less hrs |
|  | Summary for entire team | 48 hrs | 40 hrs | 8 hrs less |
| Artifact or  Deliverable | Who | Estimated (hrs) | Actual (hrs) | Difference (hrs) |
| Initial RAS | Tian Lin | 4 hrs | 2 hrs | 2 less hrs |
|  | Rong Fang | 4 hrs | 2 hrs | 2 less hrs |
|  | Lisa Frankel | 4 hrs | 2 hrs | 2 less hrs |
|  | Summary for entire team | 12 hrs | 6 hrs | 6 less hrs |

**Cumulative**

|  |  |  |  |
| --- | --- | --- | --- |
| Who | Estimated | Actual (hrs) | Difference (hrs) |
| Tian | 42 hrs | 34 hrs | 8 less hrs |
| Rong | 42 hrs | 35 hrs | 7 less hrs |
| Alex | 38 hrs | 28 hrs | 10 less hrs |
| Lisa | 42 hrs | 35 hrs | 7 less hrs |

**13.4 Defect Tracking**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or  Deliverable | Who | Estimated (# of defects found) | Actual (# of defects found) | Difference (# of defects found) |
| Initial SRS | Tian Lin | 9 | 2 | 7 |
|  | Rong Fang | 9 | 1 | 8 |
|  | Alex Huang | 9 | 2 | 7 |
|  | Lisa Frankel | 9 | 1 | 8 |
|  | Summary for entire team | 36 | 6 | 30 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or  Deliverable | Who | Estimated (# of defects found) | Actual (# of defects found) | Difference (# of defects found) |
| Final SRS | Tian Lin | 5 | 1 | 4 less |
|  | Rong Fang | 5 | 1 | 5 less |
|  | Alex Huang | 5 | 0 | 5 less |
|  | Lisa Frankel | 5 | 0 | 5 less |
|  | Summary for entire team | 30 | 2 | 28 less |
| Artifact or  Deliverable | Who | Estimated (# of defects found) | Actual (# of defects found) | Difference (# of defects found) |
| SPMP | Tian Lin | 9 | 7 | 2 less |
|  | Rong Fang | 9 | 5 | 4 less |
|  | Alex Huang | 9 | 3 | 6 less |
|  | Lisa Frankel | 9 | 3 | 6 less |
|  | Summary for entire team | 36 | 18 | 18 less |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or  Deliverable | Who | Estimated (# of defects found) | Actual (# of defects found) | Difference (# of defects found) |
| SAS | Tian Lin | 9 | 3 | 6 |
|  | Rong Fang | 9 | 2 | 7 |
|  | Alex Huang | 9 | 1 | 8 |
|  | Lisa Frankel | 9 | 2 | 7 |
|  | Summary for entire team | 36 | 8 | 28 |

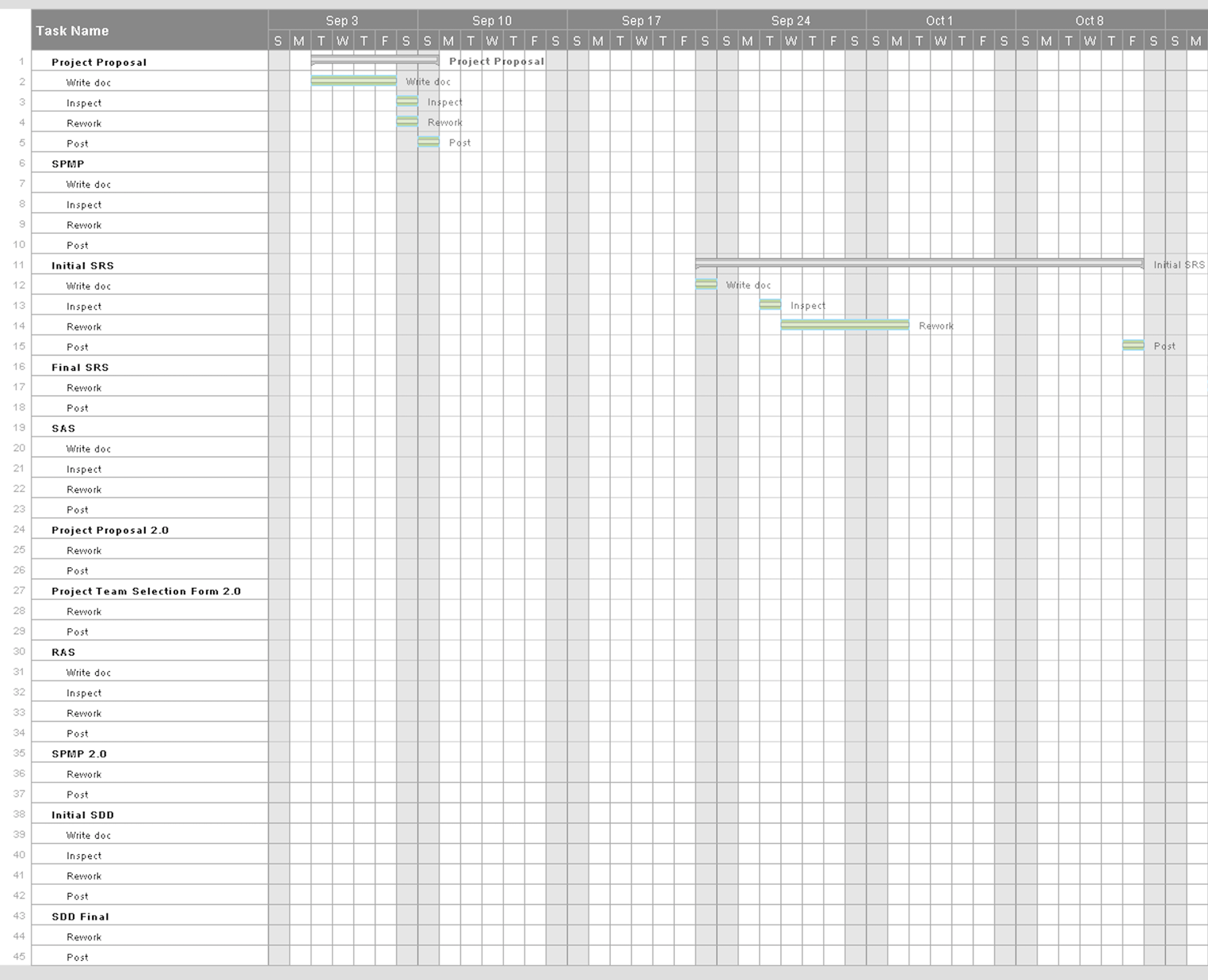
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or  Deliverable | Who | Estimated (# of defects found) | Actual (# of defects found) | Difference (# of defects found) |
| RAS | Tian Lin | 5 | 4 | 1 |
|  | Rong Fang | 5 | 3 | 2 |
|  | Lisa Frankel | 5 | 1 | 4 |
|  | Summary for entire team | 15 | 8 | 7 |

***Cumulative***

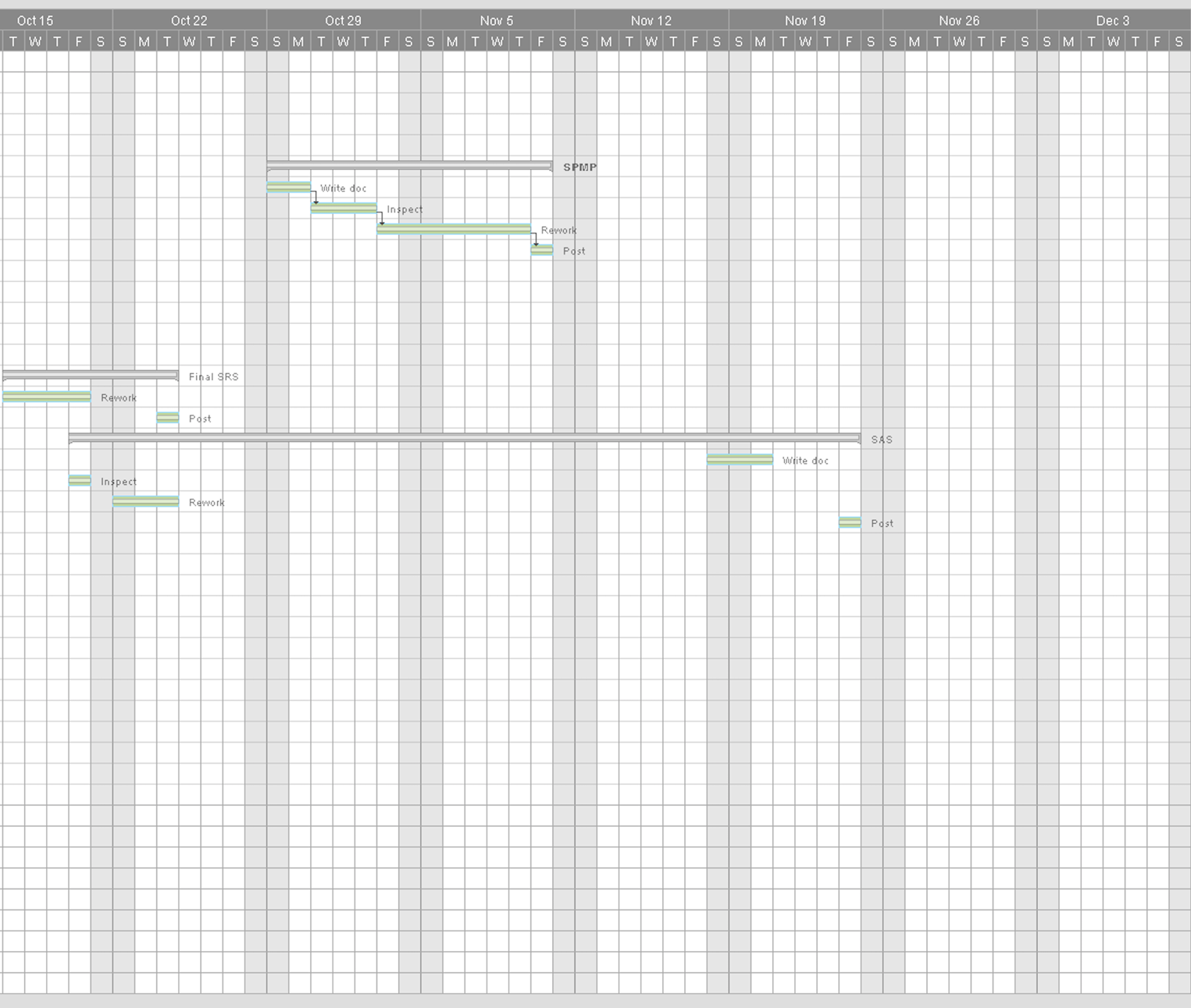
|  |  |  |  |
| --- | --- | --- | --- |
| Who | Estimated (# of defects found) | Actual (# of defects found) | Difference (# of defects found) |
| Tian Lin | 37 | 17 | 20 less |
| Rong Fang | 37 | 12 | 25 less |
| Alex Huang | 32 | 6 | 26 less |
| Lisa Frankel | 37 | 7 | 30 less |
| Team Summary | 128 | 26 | 101 less |

***Gantt Chart***

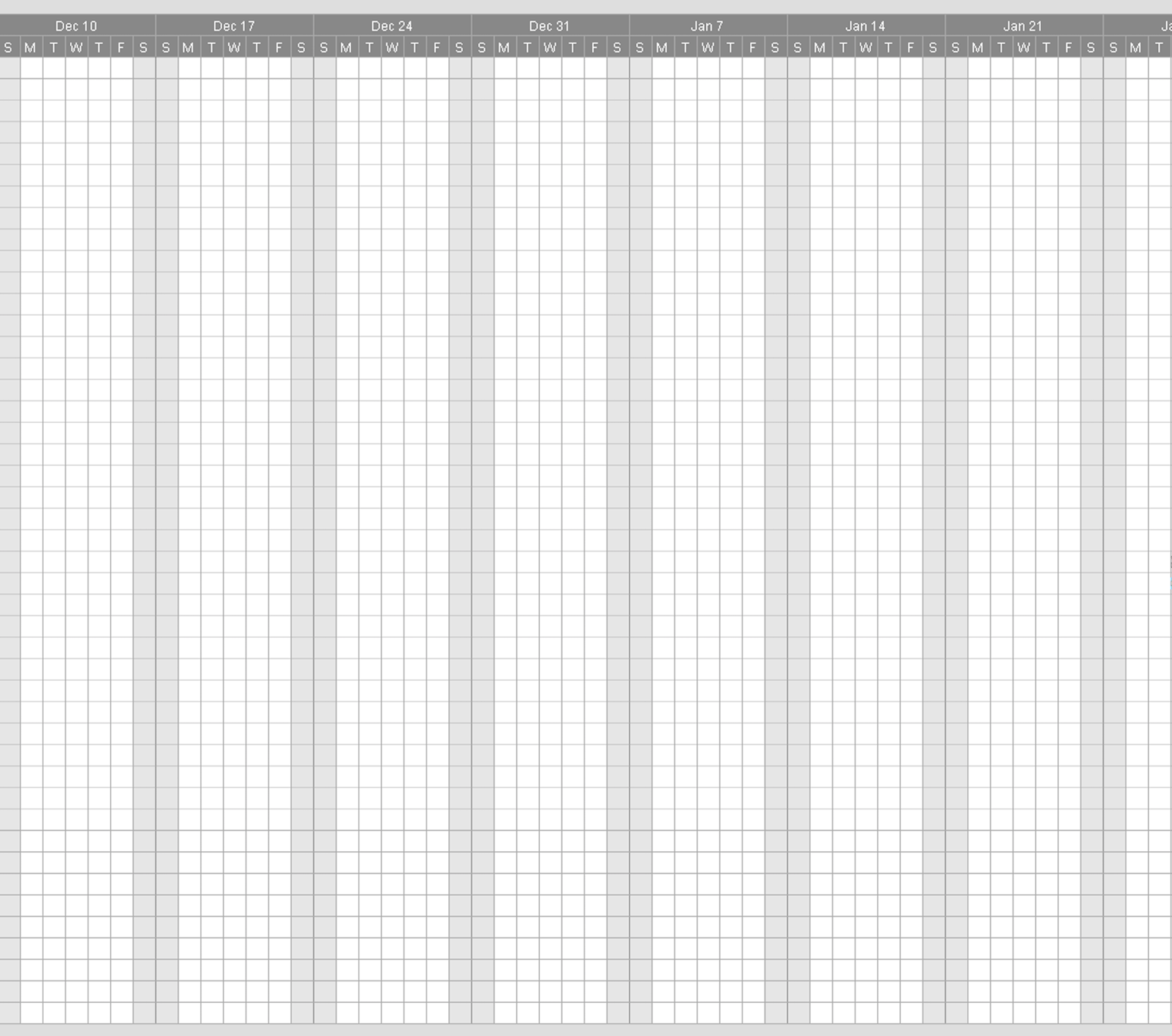
Part 1



Part 2



Part 3



Part 4

