Computer Science and Engineering

Chores App

System Requirements Specification (SRS)

Version 3.0

Document Number: SRS-003

Project Team Number: A08

Project Team Members: Helen Xu (hjx201), Kevin Grajeda (kag622), Alwyn Zhang (az1436)

REVIEW AND APPROVALS

<Team Members>	Function (Author, Reviewer, Approval)	Date	Signature
Alwyn Zhang	Author	10/07/2020	Alwyn Zhang
Kevin Grajeda	Author	10/07/2020	Kevin Grajeda
Helen Xu	Author	10/07/2020	Helen Xu
Kevin Grajeda	Author	10/19/2020	Kevin Grajeda
Helen Xu	Author	10/19/2020	Helen Xu
Alwyn Zhang	Author	10/19/2020	Alwyn Zhang
Kevin Grajeda	Author	11/17/2020	Kevin Grajeda
Alwyn Zhang	Author	11/17/2020	Alwyn Zhang
Helen Xu	Author	11/17/2020	Helen Xu

REVISION LEVEL

Date	Revision Number	Purpose
10/08/2020	Version 1.0	Initial Release
10/20/2020	Version 2.0	SRS Requirements
11/17/2020	Version 3.0	SRS Analysis

TABLE OF CONTENTS

1. Document Purpose

1.1 Purpose

The purpose of this software requirements specification document is to be a guiding tool for the developers to use to create the Chores App. It clarifies the system's specifications and function/non-functional and business requirements. This document is intended for the development team, testing team, and the clients of the system.

2. Introduction

2.1 Scope

The system will give users the ability to manage their shared tasks and responsibilities with their roommates/housemates. Since people often put off responsibilities or forgot about them, this application aims to help users stay on top of their tasks and to communicate with roommates/housemates effectively. Users can create tasks that can be seen by other users within the application and they can be modified by the user based on their status. The system is not intended to help the user complete specific tasks/chores, but to be a management tool resource for users.

2.2 Identification

Identification: Chores App SRS-001, Version 3.0

2.3 Bounds

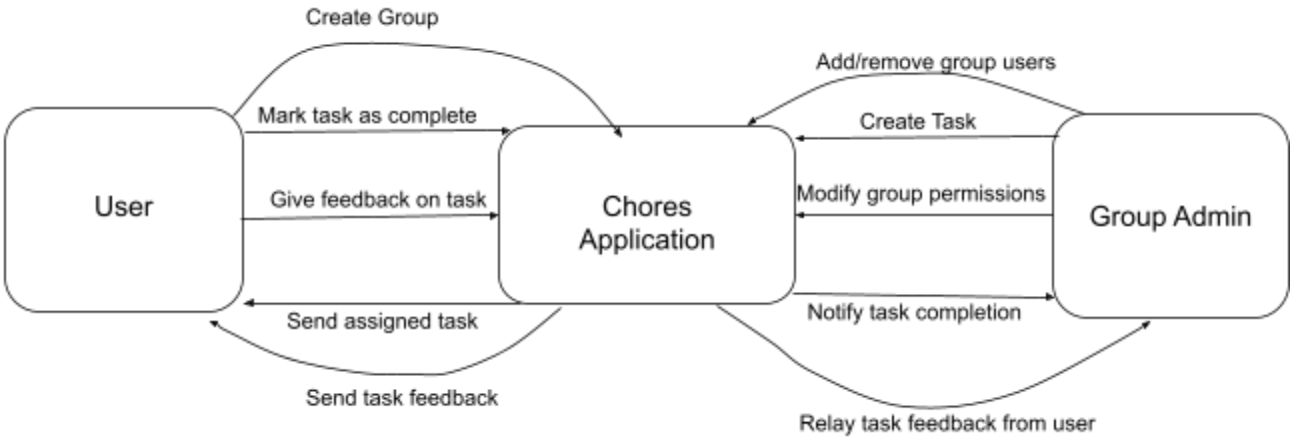
The system will require an internet connection in order for users to access it. It will also require that users create accounts using their email address and to set up a username and a password.

2.4 Objectives

The system will be developed using a combination of the Agile development model and a plan-driven model. The project's development started with the project proposal. It will continue with the development of the SRS, SPMP, and a presentation of the project itself. As for the system's development, the main priority is to create the task creation and management system. The next priorities are implementing group interaction and user account creation. The UI design has a low priority. The life cycle of the system is incremental, with the highest priority parts of the system being delivered first. The initial milestone dates are:

Deliverable	Delivery Date
Project Proposal	September 22, 2020
Software Specifications Requirement(SRS) - Domain Definition	October 8, 2020
SRS - Project Requirements	October 20, 2020
Software Project Management Plan (SPMP)	November 2, 2020
SRS - Project Analysis	December 1, 2020
Presentation	Final Two Weeks of Semester

2.5 Context Diagram



Users become group administrators when they create a group or are appointed by an administrator of an existing group. Depending on the group permissions set by the group admin, users with the group may anonymously submit feedback on tasks completed by other users, which can be sent to the admin or directly to the user responsible for the task.

2.6 Additional Descriptive Items

Product Functions:

-Allow users to create accounts

-Group creation and moderation

-Task/chore creation

-Task feedback

-Task history

-Homepage

User Characteristics:

-Regular web and mobile application experience

Constraints:

-Compatibility between web and mobile interfaces

Assumptions and Dependencies:

-Modern web browser

-Modern cell phone with internet connection

3. Glossary

None at this time. May be added in a future release.

4. Reference Documents

Project Proposal, Grajeda, Xu, Zhang, September 2020

5. Business Requirements

5.1 Technology

The application supports the business goals and objectives by appealing to households who want to organize and assign chores more efficiently and less directly.

5.2 Economics

The demand for the services provided by the application comes from families and roommates who seek to streamline managing their chores. Assigning chores is often found awkward and can feel confrontational, and the application helps users avoid such situations.

5.3 Regulatory and Legal

The regulatory business requirement we have is maintaining the privacy of users. We want to ensure that the important user information such as email address and password are protected. The application should also be secure and free from any possible exploits.

5.4 Market Considerations

Currently there exist several task management softwares made for roommates/households. While they provide intuitive interfaces to organize and schedule tasks, our app also facilitates communication by providing a system for easier communication amongst groups and the ability for users to provide feedback on tasks completed by other group members.

5.5 Risks and Alternatives

No.	Title	Est. Likelihood of Occurrence	Est. Impact	Est. Cost of Mngmt.	Priority No.	Retirement Plan	Responsible Person	Target Completion Date
1.	User falsely reports task as complete	5	7	6	1	Completed tasks show up in a history log	Kevin Grajeda	Rolling basis
2.	User falsely or accidentally deletes task	3	7	6	2	Deleted tasks show up in a history log	Alwyn Zhang	Rolling basis
3.	User creates same chore twice	4	5	3	3	Check if a similar tasks exists	Helen Xu	Rolling basis

5.6 Human Resources and Training

The engineering team will have sufficient training to be qualified to develop all the components of the system. The entire team must be familiar with all of the project's documentation.

6. User Requirements (Descriptive Functional and Non-functional requirements)

6.1 Functional Descriptive Detailed Requirements

1. System will allow users to create an account

   1.1 Users will be able to create a username and password using an email address. Their username will be the main form of identification within the system.

   1.2 Creation of account will allow users to access all the features of the system

2. System will have a GUI for navigation

   2.1 Users will be able to easily manage tasks and join groups through the GUI.

   2.2 Includes a home page for quickly accessing different parts of the application.

3. Ability to create, join, and moderate groups

   3.1 A user can create a group for other people to join. In the group, people can view their tasks and the tasks of other group members.

   3.2 A user can be invited to a group through their username by other users already within the group.

   3.3 One user or multiple users can moderate the group's tasks. They can also manage group settings such as permissions to create and modify tasks. They can also review completed tasks.

   3.4 Group moderator can give other users permissions to create and modify tasks

4. Ability to create and view tasks/chores

   4.1 Group moderators can create tasks. The tasks can be named, given start/end dates, given duration, given descriptions, marked as complete, deleted, or set to be recurring. Other users can view the tasks created and the information related to the task. Users can view a list of all the group's tasks or a list of just their tasks.

5. Ability to give task feedback

   5.1 Users can give feedback to the group telling them that their tasks is complete or if there are any complications such as the user not being able to complete the task on time or at all

   5.2 Users can also notify others to remind the other uses to complete their task

6. Ability to view task history

   6.1 System will keep track of the tasks created and their modifications in a log for the group's reference

6.2 Non-Functional Descriptive Detailed Requirements

1. Product Requirement - Availability

   1.1 The system should be available at all times and should never be down for more than an hour.

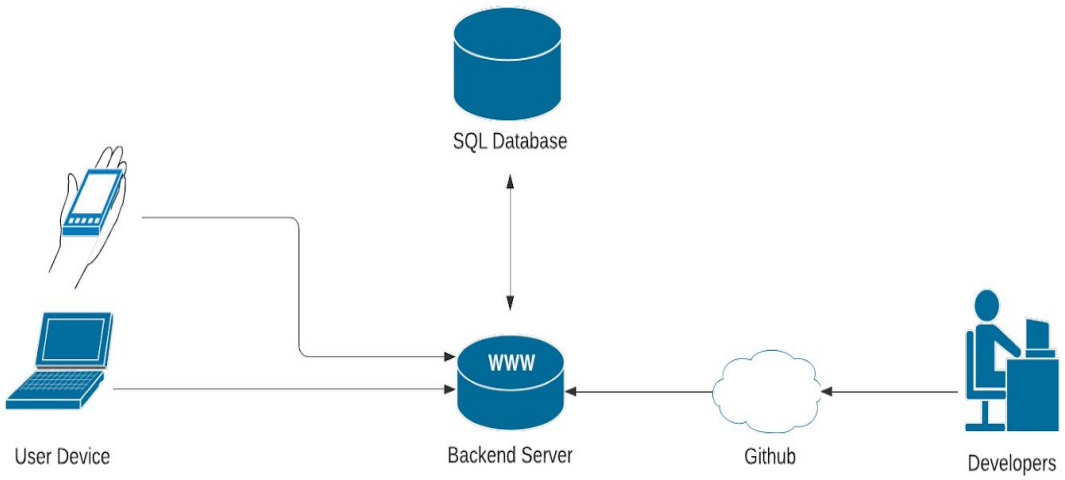
2. Ethical requirements - Privacy and safety

   2.1 The application should not compromise any user info that is stored on the website.

7. System Architecture

The system will consist of 3 architectural components:

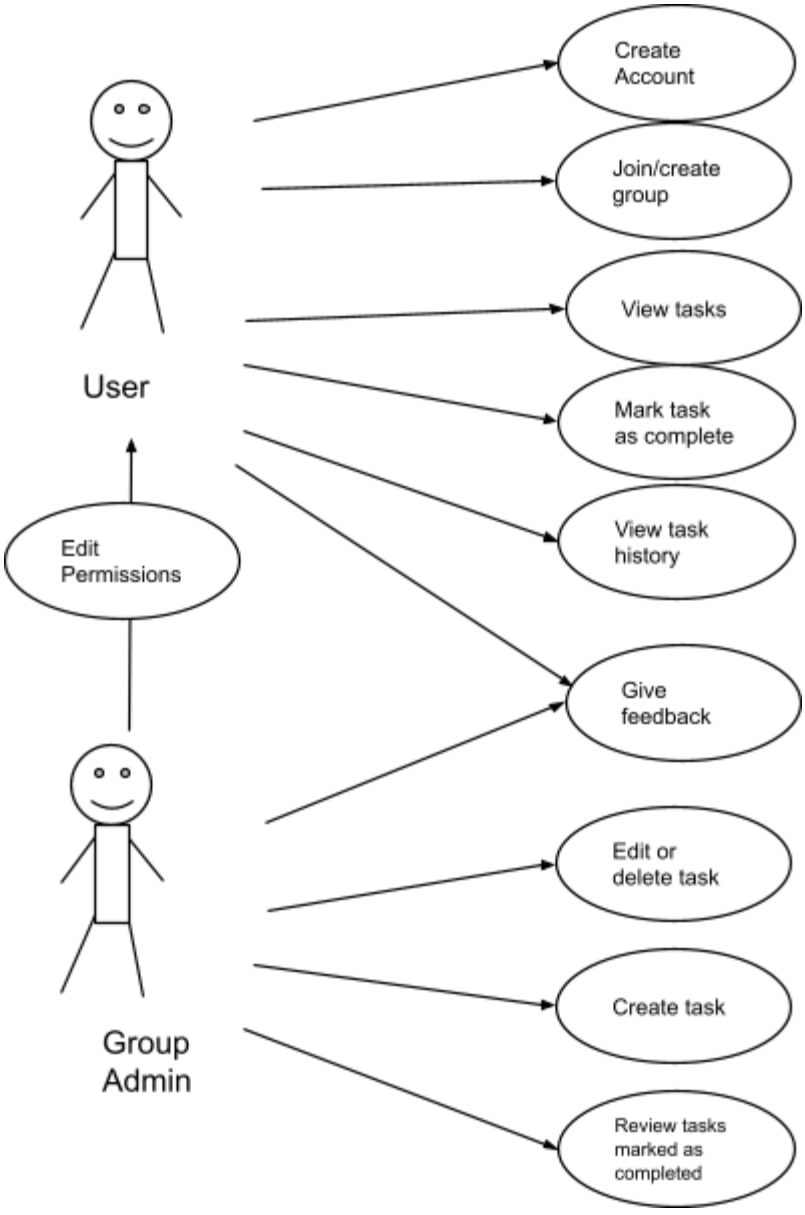
1. Website: The website is the user's interface and provides the user with the means to interact with the server.
2. Server: The server runs the system's logic and will retrieve anything from the database that the user requests.
3. Database: The database stores any user information and other information that the application needs to function.



8. Detailed System requirements – Use Cases

8.1 Requirement Use Cases

8.1.1 Use Case Diagrams



8.1.2 Use Case Descriptions

Create Account		
Description	The user can use their email and enter a password to register for an account.	
Pre-Conditions	None	
Flows	**Basic or Normal Flows**	1. User navigates to account creation page 2. The system prompts the user to input a valid email address and a password that they will use to log in. 3. The user provides the necessary input.
	Alternative Flows	1. If there is no internet connection the user will be shown an error message and encouraged to retry.
Post Conditions	The user will have an account in the system.	
Special Requirements	The user must be connected to the internet.	
Extension Points		

Join/Create Group		
Description	The user can create a new group or join one that currently exists.	
Pre-Conditions	User has registered for an account.	
Flows	**Basic or Normal Flows**	1. User chooses whether to join or create a group. 2. The system prompts the user to input a group code or shows a page for group creation depending on the user's choice. 3. The user provides the necessary input.
	Alternative Flows	1. If there is no internet connection the user will be shown an error message and encouraged to retry.
Post Conditions	The user will be in a group.	
Special Requirements	The user must be connected to the internet.	
Extension Points		

View Tasks		
Description	The user can view a list of the group's tasks or their tasks.	
Pre-Conditions	User must be in a group	
Flows	**Basic or Normal Flows**	1. The user wants to see their tasks. 2. They open up a list of the group's tasks. 3. They select a specific task and view its information.
	Alternative Flows	1. If the user wants to view their own tasks, they can filter the list to just show their tasks.
Post Conditions	The user is aware of the group's task and/or their own tasks.	
Special Requirements	The user must be connected to the internet.	
Extension Points		

Mark Task As Complete		
Description	Once a user finish's their task, they can mark it as complete or incomplete for other users to see.	
Pre-Conditions	At least one task must be created and the user must be in a group.	
Flows	**Basic or Normal Flows**	1. After the user finishes their tasks, they select their completed task in the application in their task list. 2. The user marks their tasks as completed. 3. System asks the user if they are sure . 4. User selects that they are and the system will mark the task as complete. If not, the user is sent back to their task list.
	Alternative Flows	1. If the user accidentally marked the task as complete, they can undo it by selecting the task again. 2. System asks the user if they are sure. 3. User says that they are and the system will mark the task as incomplete. If not, the user is sent back to their task list.

Post Conditions	The task is marked as complete or incomplete.
Special Requirements	The user must be connected to the internet.
Extension Points	

View Task History		
Description	Users can view a task log which keeps track of the tasks being created, edited, and marked as completed and verified.	
Pre-Conditions	At least one task must be created and the user must be in a group.	
Flows	**Basic or Normal Flows**	1. User opens the task log in the application. 2. The user can scroll through the list of tasks.
	Alternative Flows	1. If the user wants to see a specific task's information they can tap on a task and view its information.
Post Conditions	The user is presented with all the information about the group's tasks.	
Special Requirements	The user must be connected to the internet.	
Extension Points		

Give Feedback		
Description	Users can give feedback to the group on whether the status of their task.	
Pre-Conditions	At least one task must be created and the user must be in a group.	
Flows	**Basic or Normal Flows**	1. The user completes their task 2. The user can open up the task on the system and open up the feedback section 3. User can give notification that they're task is complete
	Alternative Flows	1. The user did not/could not complete their task. 2. The user can open up the task on the system and open up the feedback section. 3. They can give the notification that they could not complete the task on time along with an optional small message as to why.

Post Conditions	Other users in the group are aware of the status of a specific task
Special Requirements	The user must be connected to the internet.
Extension Points	

Edit or Delete Task		
Description	If a moderator wants to change an already existing task, they can choose to edit information related to a task, or they can delete it all together.	
Pre-Conditions	At least one task must be created and the user must be in a group.	
Flows	**Basic or Normal Flows**	1. The user opens up the list of existing tasks. 2. The user selects the tasks they want to edit. 3. The task's information is shown to the user and the user can select which piece of information they want to change (name, duration, description, completion, recurring status). 4. The user makes their changes.
	Alternative Flows	1. The user wants to delete the task. 2. They open up the task's information and select to delete the task. 3. System asks the user if they want to delete the task. 4. If they select yes, it is deleted. If not, they are sent back to the task information screen.
Post Conditions	Task is edited to the user's desires.	
Special Requirements	The user must be connected to the internet.	
Extension Points		

Create Task		
Description	The group administrator can create new tasks to assign to users.	
Pre-Conditions	User must be group admin, or have permission to create tasks	
Flows	**Basic or Normal Flows**	1. User opens up the task creation menu.

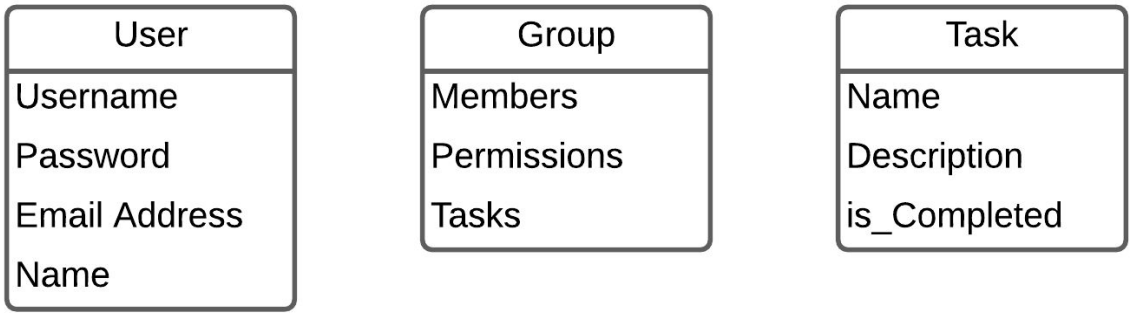
		2. User gives the task a name and any descriptions it may need. 3. User assigns the tasks to users within the group.
	Alternative Flows	1. If the user does not have permission to create tasks, they will be shown a message that tells the user to seek out the group admin to modify their permissions or create tasks for them.
Post Conditions	The new task will have been created and visible to all users.	
Special Requirements	The user must be connected to the internet.	
Extension Points		

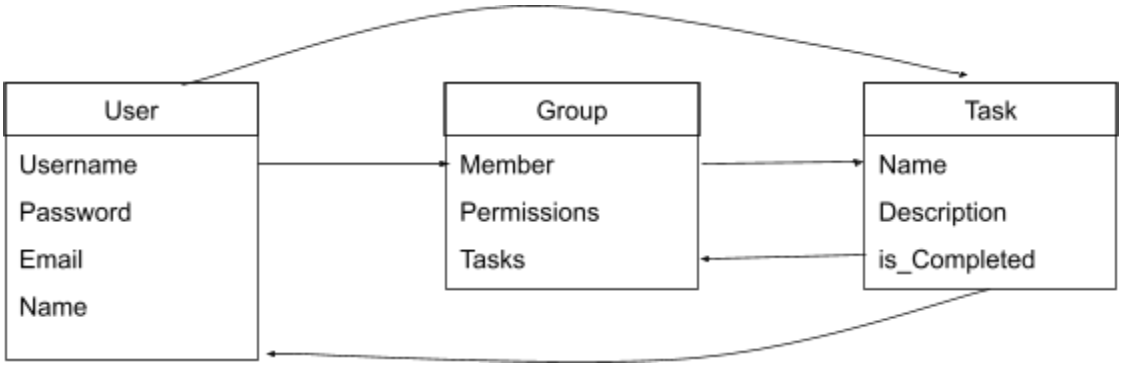
Review Task Marked as Completed		
Description	Group moderators can check the tasks of other users within the application and then check in real life that it has been completed. Then the moderator can verify that the task has been completed.	
Pre-Conditions	Another user must have marked a task as completed.	
Flows	**Basic or Normal Flows**	1. Group moderator sees that a task has been marked completed. 2. After the moderator has seen that the task has been completed in person, they can verify the task as completed.
	Alternative Flows	1. Group moderator sees that a task has been marked completed. 2. The moderator sees that the task was not actually completed. 3. Moderator can undo task completion and give a notification to the user who did not complete the task.
Post Conditions	Moderator will have verified that a given task is complete.	
Special Requirements	Moderator must be connected to the internet.	
Extension Points		

9. System Model (UML)

9.1 Static - Class Diagrams

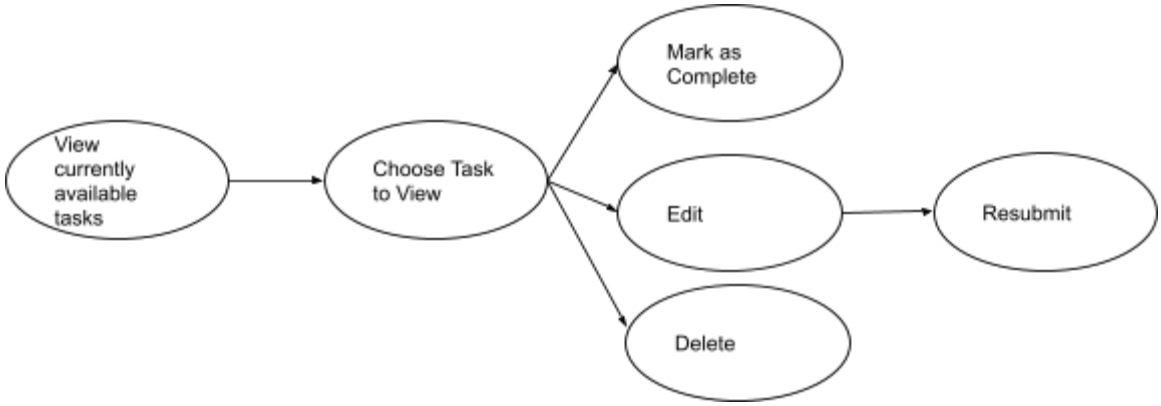
1. User
2. Group
3. Task

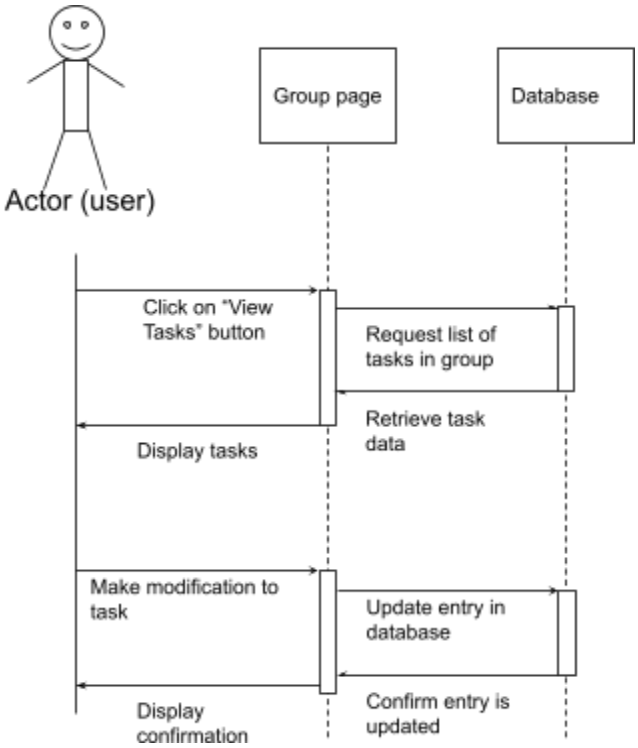




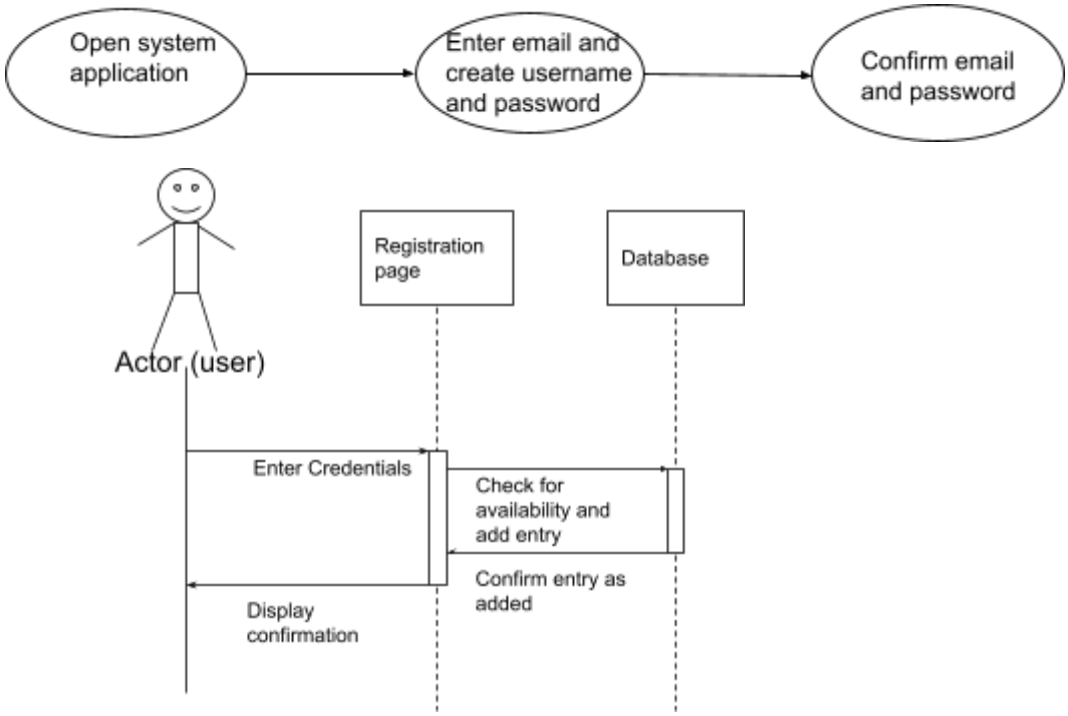
9.2 Dynamic - Behavioral Models

1. Edit, delete or mark existing tasks as complete

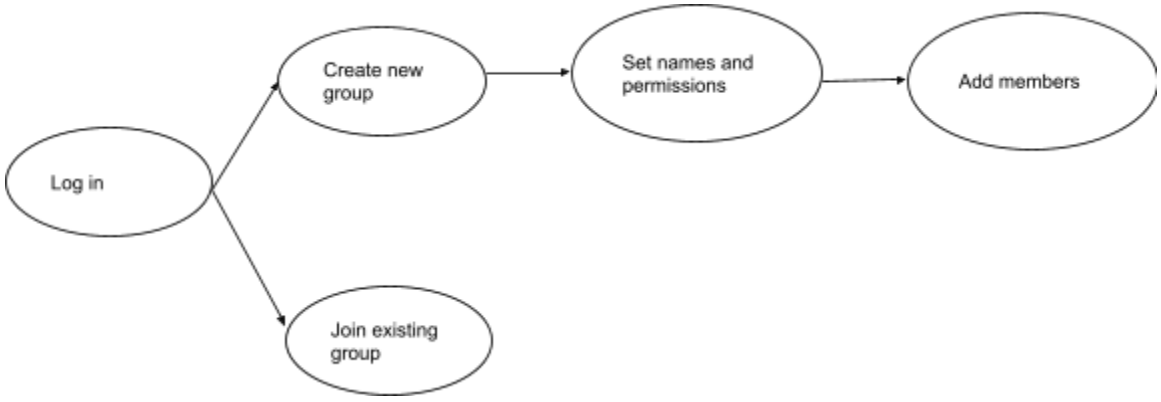


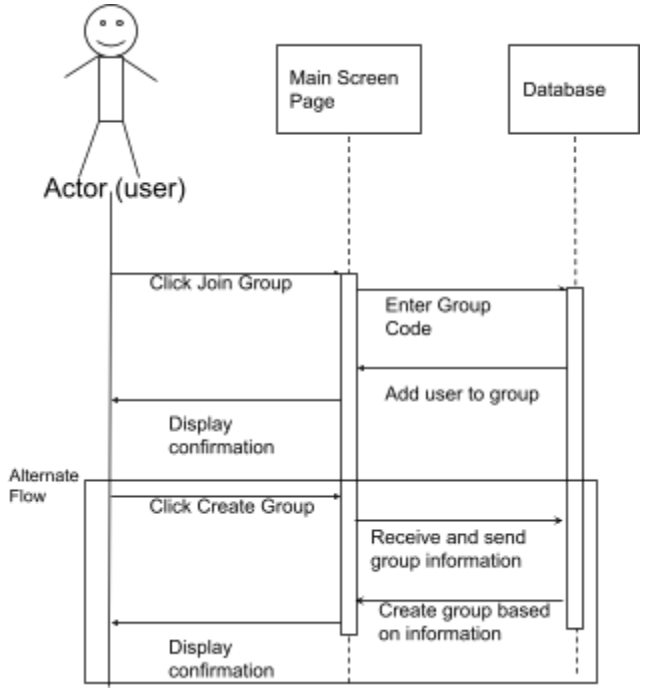


2. Create an account

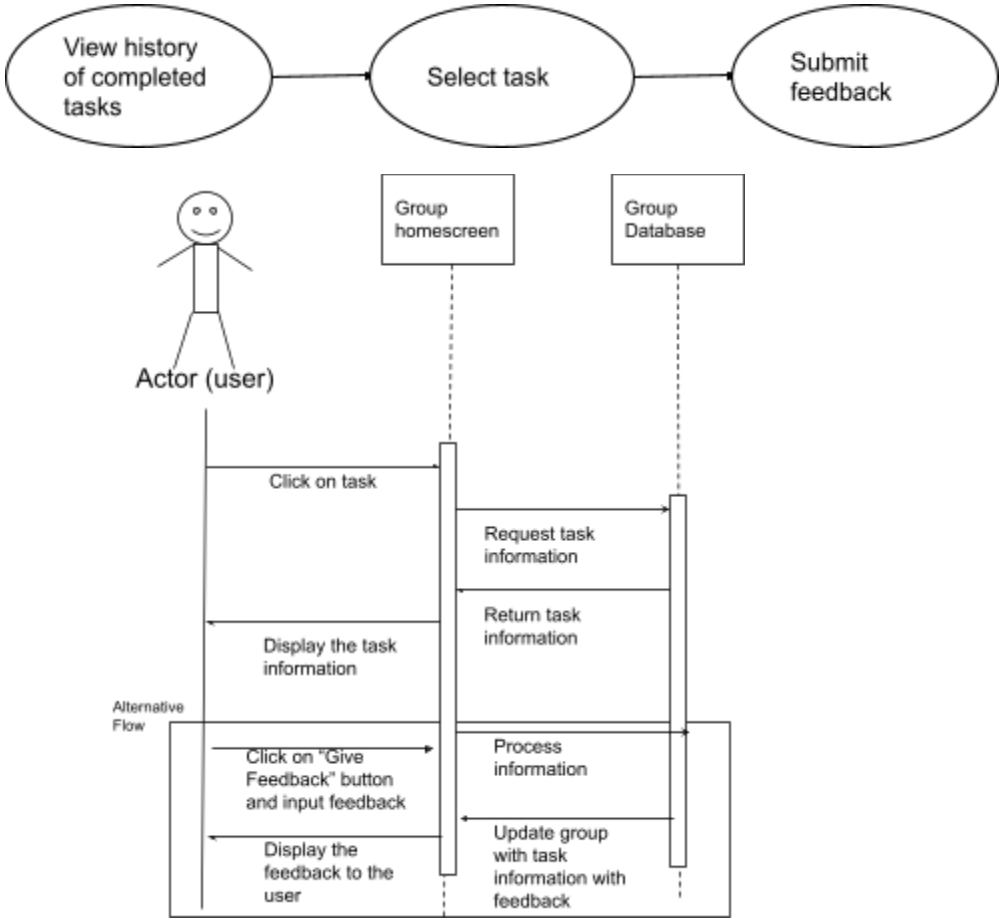


3. Create/join group

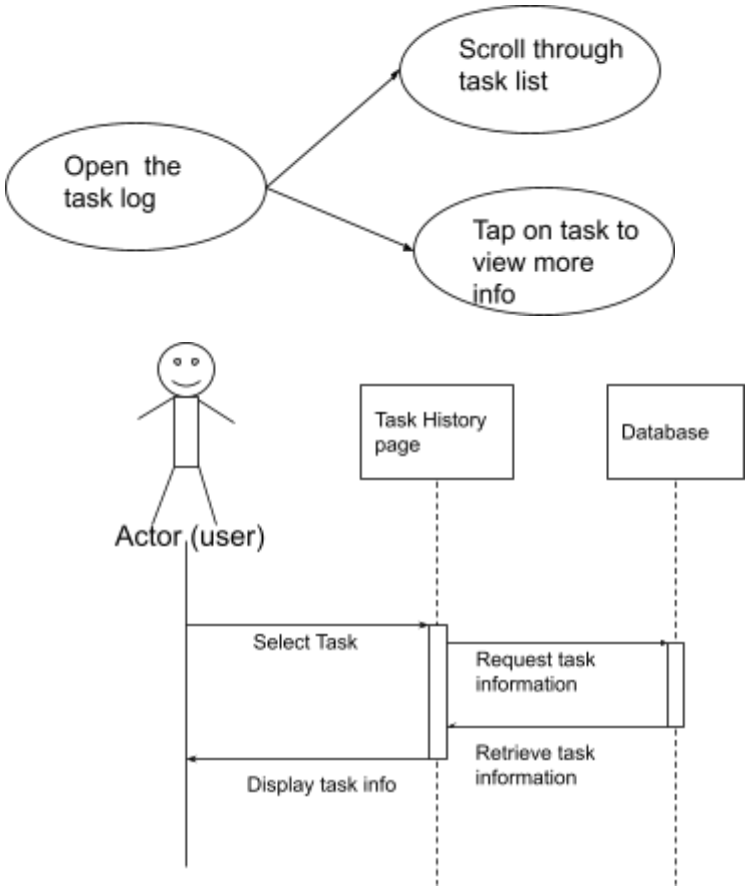




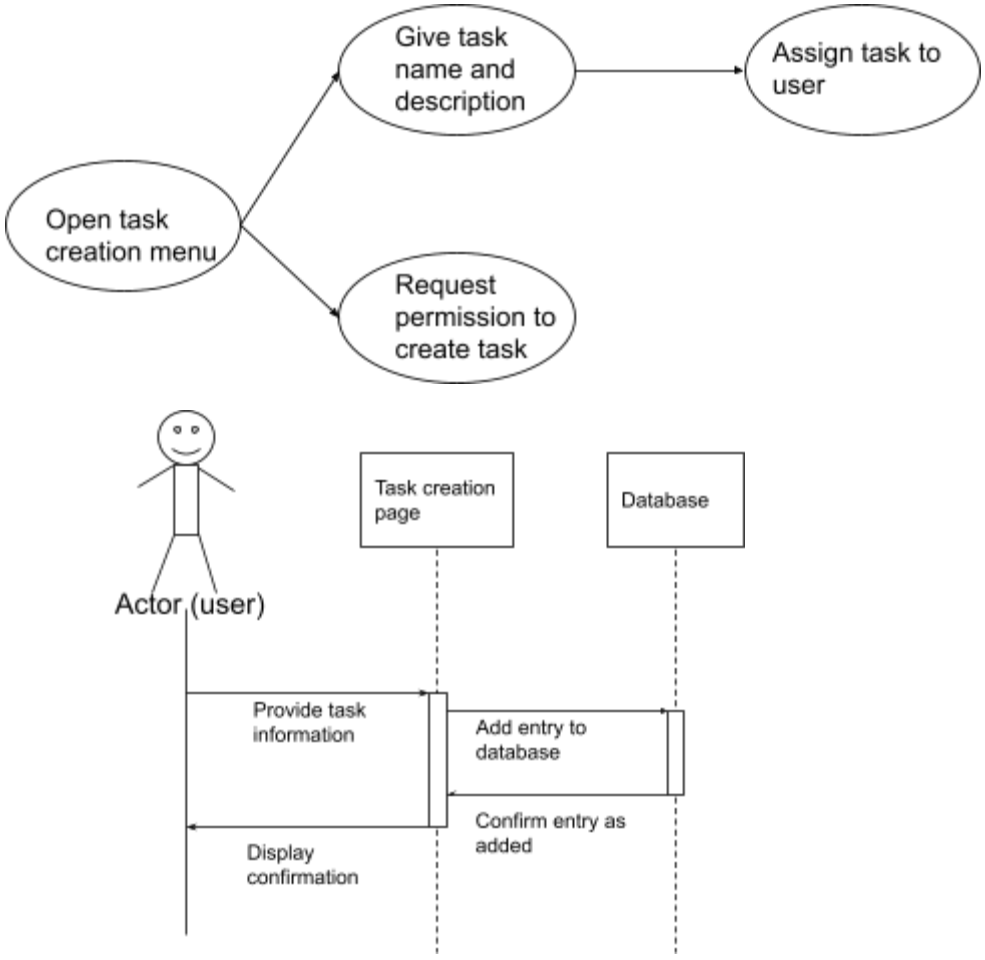
4. Review Completed tasks + give feedback



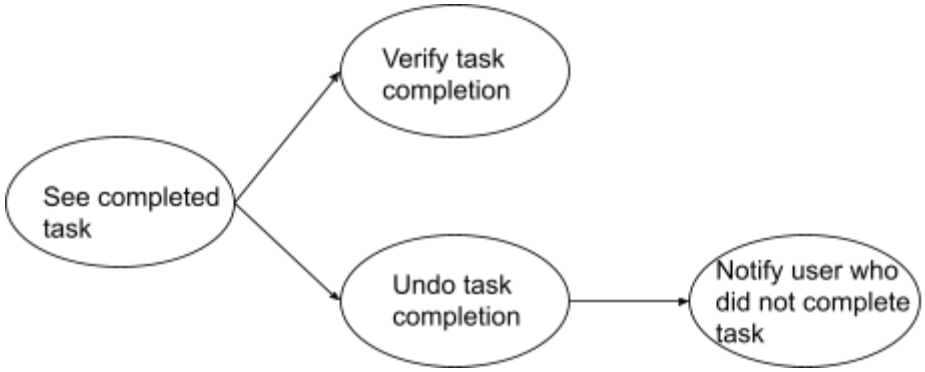
5. View history

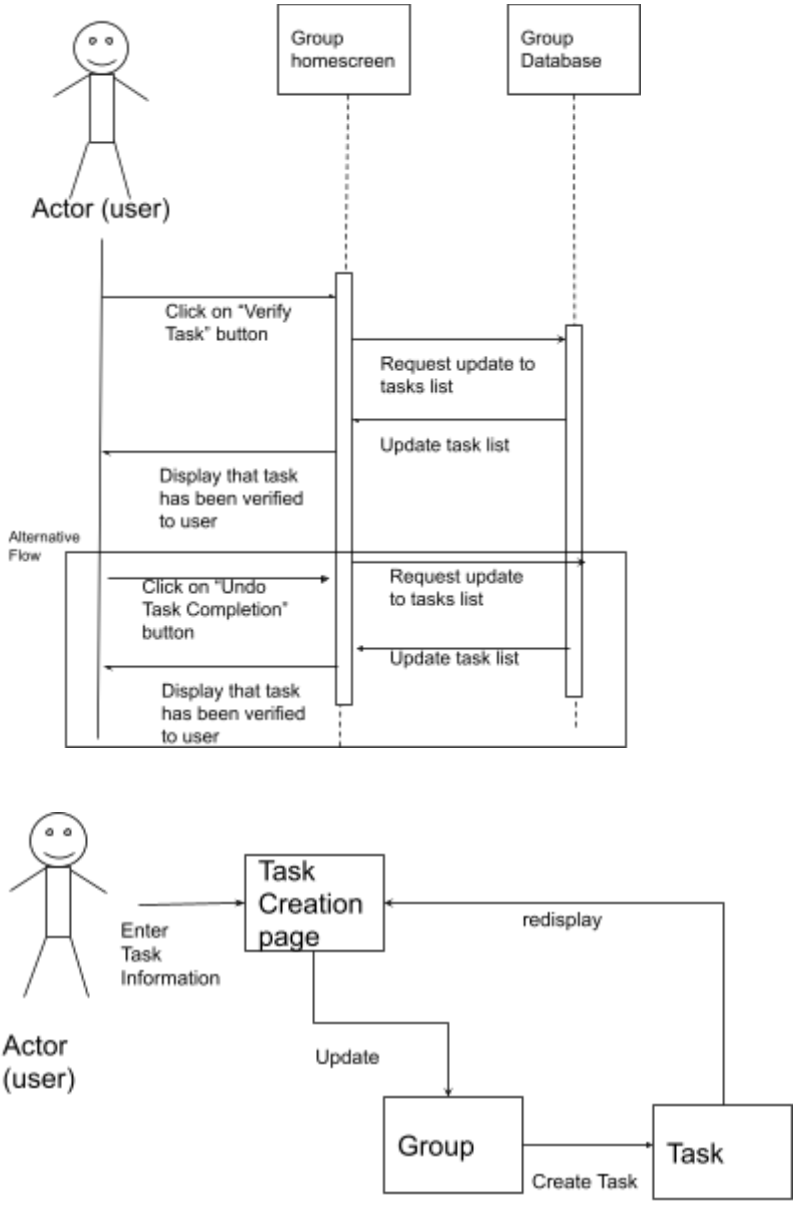


6. Create task



7. Review completed tasks





10. Evolution of the SRS

The SRS will be updated when:

1. Requirements are changed, added or removed
2. If inaccurate information or shortcomings are found
3. When deliverables are completed

Any changes to the SRS will be reported; the changes will be checked and approved by everyone in the group and the version number updated when there is a change.

11. Rationale

None currently. May be added in a future release.

12. Notes

None currently. May be added in a future release.

13. APPENDICES

13.1 System Test Plan Requirements

The SQA testing process will be used to evaluate the system to ensure that the requirements specified in this document are being met. It will also evaluate the system's security and reliability.

To test each requirement, the system will be run through a test scenario similar to a use case involving that requirement. The results of the test will be compared to a standard set for that requirement before the test.

In order for the software to be reliable, it needs to be able to have a 99% uptime with an average to high amount of traffic. This will be tested by running the application with simulated traffic.

It must also be tested for security, as the software should not be easily hacked. The application will be tested for any vulnerabilities through ethical hacking.

13.2 Qualification Provisions

This document will be reviewed for correctness, completeness, and consistency. It will be reviewed both individually and as a group to ensure that it is unambiguous, stable, modifiable, verifiable, and traceable.

If a defect is discovered during a review, it will be brought to the group for verification and a group member will be assigned to fix the defect. Once it has been fixed, the other group members will review the document to affirm the defect has been resolved. If the group determines that the defect has not been fixed, the group will discuss how to best resolve the issue and modify the document accordingly upon coming to an agreement.

13.2.1 Requirements Traceability

The type of requirements traceability being used is backward. Each requirement for the system has the Project Proposal, with specification being in the SRS document as its origin. Non-functional requirements can also be traced to the SRS document. Requirements can be traced to the SRS from any point of development by referencing the User Requirements section. Any future documentation mentioning the system requirements can also be traced similarly.

13.3 Schedule Tracking

Artifact or Deliverable	Who (individual or Team)	Estimated	Actual	Difference
SRS Domain	Alwyn Zhang	3 hours	2 hours	1 hour
SRS Domain	Kevin Grajeda	4 hours	2 hours	2 hours
SRS Domain	Helen Xu	3 hours	1 hour	2 hours
	Entire Team	10 Hours	5 hours	5 hours

Artifact or Deliverable	Who (individual or Team)	Estimated	Actual	Difference
SRS Requirements	Alwyn Zhang	2 hours	2 hours	0 hours
SRS Requirements	Kevin Grajeda	3 hours	2 hours	1 hour
SRS Requirements	Helen Xu	3 hours	2 hours	1 hour
	Entire Team	8 hours	6 hours	2 hours

Artifact or Deliverable	Who (individual or Team)	Estimated	Actual	Difference
SRS Analysis	Alwyn Zhang	2 hours	2 hours	0 hours
SRS Analysis	Kevin Grajeda	2 hours	2 hours	0 hours
SRS Analysis	Helen Xu	3 hours	1 hour	2 hours
	Entire Team	7 hours	5 hours	2 hours

Cumulative

Who (individual or Team)	Estimated	Actual	Difference
Alwyn Zhang	7 hours	6 hours	1 hour
Kevin Grajeda	9 Hours	6 Hours	3 hours
Helen Xu	9 hours	4 hours	5 hours
Entire Team	25 Hours	16 hours	9 hours

13.5 Defect Tracking

Artifact or Deliverable	Who (individual or Team)	Estimated	Actual	Difference
SRS Domain	Alwyn Zhang	5	6	1
SRS Domain	Kevin Grajeda	12	8	4
SRS Domain	Helen Xu	7	5	2
	Entire Team	24	17	7

Artifact or Deliverable	Who (individual or Team)	Estimated	Actual	Difference
SRS Requirements	Alwyn Zhang	2	4	2
SRS Requirements	Kevin Grajeda	10	7	3
SRS Requirements	Helen Xu	5	6	1
	Entire Team	17	17	6

Artifact or Deliverable	Who (individual or Team)	Estimated	Actual	Difference
SRS Analysis	Alwyn Zhang	1	5	4
SRS Analysis	Kevin Grajeda	6	5	1
SRS Analysis	Helen Xu	2	5	3
	Entire Team	7	15	8

Cumulative

Who (individual or Team)	Estimated	Actual	Difference
Alwyn Zhang	8	15	7
Kevin Grajeda	28	20	8
Helen Xu	14	16	2

Entire Team	48	49	1

13.6 Dictionary

Classes

Name	Description	Methods	Attributes
User	Stores the data of any user of the app including login credentials.	CreateUser() Login()	Username Password Email Address OwnedGroups
Group	Stores data on any group, including user permissions and a data structure of tasks.	CreateTask() EditPermissions() DisplayTasks()	Name Members Permissions Tasks
Task	Stores data on a task created by a group admin, including the task name, a description, and the users assigned.	EditTask() DeleteTask()	Name Description Is_completed Assigned

Methods

Name	Description	Class	Arguments
CreateUser()	Creates and stores a user on the app database	User	username password email
Login()	Takes the user to their dashboard both arguments are valid	User	email password
CreateTask()	Creates a task object within the group	Group	name description assigned

EditPermissions()	Changes a certain user's permissions within a group or grants admin status	Group	user permissions
DisplayTasks()	Displays every task in the group's task list on the user's screen	Group	
EditTask()	Modifies the passed task object's name or description	Task	name description
DeleteTask()	Removes the task object associated with the task selected by the user from the task list	Task	name

Attributes

Name	**Description**	**C/S**			**R/W**
User					
Username	Username of user (login and identification)	Simple	VARCHAR	20	R
Password	Password of user (for login)	Simple	VARCHAR	20	R/W
Email Address	Email address of user	Simple	VARCHAR	30	R/W
OwnedGroups	Groups owned/created by this user	Complex	Group.Names		R/W
Group					
Name	Name of group	Simple	VARCHAR	20	R
Members	List of members in group	Complex	User.Usernames		R/W

Permissions	Permissions for members in group	Complex	Permissions		R/W (only by admin)
Tasks	Data structure of tasks for that group	Complex	Tasks		R/W
Task					
Name	Name of task	Simple	VARCHAR	20	R/W
Description	Description of task	Simple	VARCHAR	100	R/W
Is_completed	Whether task is completed or not	Simple	Boolean		R/W
Assigned	People assigned to task	Complex	User.Usernames		R/W

Relationship

Name	Description	From Class	To Class	Optional	Cardinality
owns	Users create groups	User	Group		one-to-many
is in	Users can be in groups	User	Group		many-to-many
contains	groups contain tasks assigned to their members	Group	Task		one-to-many
is assigned to	tasks can be assigned to users	Task	User		many-to-many

Key Events

Name	Description	Motive	Action	Pre-Conditions	Post Conditions	State Change
Create Account	Users create	Gain access to app	Enter login credentials	Must be connected to	Account is created	New entry in database

	accounts			internet		
Create Group	User creates group	Create a space to interact with groupmates	Enter group name and establish permissions	Must be logged in	Group is created and user is in group	New entry in database
Create Task	User creates task	Assign work to groupmates	Enter task information	Must be logged in and in a group	Task is created and assigned to at least one user	New entry in database

14. Index

None currently. May be added in a future release.