Do More and Prosper (DMP) Software

System Specification (Design)

Prepared for:

Mr. Andy Cameron

Prepared By:

Ammanuel Beyene

April 26, 2022

Table of Contents

1.0 Executive Summary

The DMP is a software that will be designed and developed by Ammanuel Beyene and teammates. The DMP is made for anyone who is above 18 years old. It is a software tool that provides a way for adults to find nearby events, jobs, interest groups, local news, and political decision-making opportunities. People have to sign up as a regular user or poster. Posters can post things like new events, jobs, meet-ups, etc. Users can also post to these sections as posters but will have limited functionality. While all the sections include posts from others, the planner will be a private section for each user.

The DMP's goal is to fix issues such as excessive technology use, time-consuming navigations between different websites, and the effects brought about by the recent covid epidemic. The DMP wants to help people organize their tasks and save time by putting some of their most important tasks in one place. By giving users an opportunity to go to events, public sport meet ups, local politics decision makings, the DMP wants to help people get active, increase their interaction with others, and help them make positive changes to their community. The local politics section was added to the DMP with the idea of encouraging citizens to participate in their local politics and make the changes they want to see in their community.

So far, we have all the proposals, designs, and plans outlined. But the makers of the DMP software do not have all the resources available to implement the DMP as it was hoped to be. There are some risks and constraints, which are mentioned in the problem statement and vision sections. But the MVP should be feasible. After finishing this document, the next steps are to start the design and development phase of the DMP.

This system specification document should guide the next process – the development. This document describes the software's vision, benefits, constraints, functionalities, infrastructure models, security management techniques, and user interface designs. It is pretty obvious that the project has a large scope. So, it might take time to develop. The goal is to start early and to have a working product as soon as possible to meet the schedule needs.

2.0 Introduction

The DMP is a software that is made to help people organize their daily tasks more efficiently, spend less time on technology, recover from the recent effects of covid by enabling them to access important features quickly and easily, as well as make positive changes to their community. Users can sign up as a regular user or a poster. The only requirement is that users have to be above 18 years of age and after signing up, they have to wait three days for the background check to complete. After which if they are approved, they can have an account. If denied, they can't have an account. Regular users will be able to see various posts, sign up for them or follow their links. Posters will only be able to post contents. Regular users will also be able to post but under limited functionalities. The DMP will have sections for events, jobs, meetups, interest groups, life hacks, local news, local politics, and a planner. The news and politics sections are handled by the administrators. Neither regular users nor posters can post on those sections. But all users can interact with the posts by commenting, liking, sharing or replying. The planner section will be a private section catered to each user.

For more info – see System Proposal Part 1+2 Document Pages – 5 to 8.

2.1 Problem Statement / Project Vision

This software is made by taking into considerations a few different challenges faced by today's societies such as excessive technology use, busy schedules, covid's recent effects, and lack of prodigious political participators. It's no surprise people use a lot of technology these days and as a result, are missing out on the important human to human interactions. Technology use is so varied and widespread that there are so many things' people do and have to do online. Additionally, the recent epidemic, covid, has affected a few people's jobs and businesses. Lastly, not everyone is a fan of participating in political decision makings.

In order to organize people's online tasks, the DMP software wants to take some of the important features and put them together in one place. In relation to that, the DMP will have a planner that is set up to help people schedule and organize their various tasks. The hope is to reduce hopes online surfing times as well as enable people to quickly and efficiently execute their tasks. The DMP will also have sections for local events, meet ups, and interest groups that are meant to encourage people to go out, interact, and be active. The DMP wants to help people affected by covid, by giving them the opportunity to apply and get jobs. Additionally, the DMP wants to give people the opportunity to stay informed about their local politics as well as participate in the decision-making process. DMP makers believe that political participation is essential not only to make positive changes but also to make changes that the citizens want.

The main stake holders of the DMP are Ammanuel Beyene (Owner), users, event coordinators, job coordinators, various Companies with job/event/interest group availabilities, public sport enthusiasts, news stations, local politics and government, and the community as a whole. The DMP will provide the opportunity for job coordinators, event coordinators, and public meeting coordinators to post their jobs, events, or meetings and to get local people to view them as well as sign up for them. News stations and local - government will be able to get more people to review their posted information. Additionally local government will be able to get more decision-making participators. Therefore, the benefits are tremendous for everyone that is part of the participating community.

2.2 System Capabilities

The DMP software will provide the following functionalities and specified functional requirements catered to the two types of users.

2.2.1 Functional Requirements

The DMP will have a software system and design that is user-friendly and can be used seemingly easily with a good overall flow.

A detailed list of all the currently up to date functional requirements are stated below in their corresponding sections.

2.2.1.1 Account setup:

- Users will create their account by entering all the required information such as account type, personal info, contact info, and interests (chosen from a range of interests to help with matching users to the different posts and their requirements)
- Users must enter their interests when signing up to help match users to the different posts as well as their requirements

- The software will guide the user on how to use the application once the user has a valid account
- A message will be sent to the user from the application administrators on how to use the app and other welcome messages
- The software will provide a choice between a written description tutorial or a step-by-step guide tutorial for a user to use to learn about how to use the application
- Once an account is up and running, users can configure their settings, edit their profiles, edit their preferences, display settings, etc.

2.2.1.2 Security and Background check:

Main reference: Use Case ID #2: Pass a Background Check

- When new users sign up for an account, the system will let the users know it will conduct a background check, and if they don't pass that, they won't be able to create an account *(Reference – See - Use-Case ID #2 on System Proposal Document part 1 +2)*
- Once a user enters their personal information and submits a request to have an account, the background check validation process will take 3 days to check the user's personal info and respond with an approval or rejection message to create an account
- The system will also let users know that if they are reported for inappropriate behavior or suspicious activity and found to be guilty of it, their account will be removed *(Reference – See - Use-Case ID #7 on System Proposal Document part 1 +2)*
- Every time a user updates personal info, a background checker will reevaluate the users account to validate the information entered
- The system will alert the user that their information will be reevaluated every time they update their personal information or profile
- The software will accept and record user's reports.
- Security manager will assess the reports and take appropriate actions (*Reference* – see – *Use-Case ID #7 on System Proposal Document part 1 +2)*
- Up on assessment and investigation of the reports, if users are found to be guilty, irresponsible or a fraud, they will first be notified and depending on the severity of the situation, they will be warned or removed.

2.2.1.3 Common features:

- The home page will start with a display of the planner section.

- The buttons for links to the other sections will be listed on the left side of the home page as well as most of the pages
- Events, Jobs, Public Meetups, and Interest Groups pages will have a covid warning quote posted on top of the pages. *(Reference – See - Use-Case ID # 3 – on System Proposal Document part 1 +2)*
- The covid warning will indicate users to carry out covid precaution measures as they interact with people or carry out their day-to-day activities.
- The covid warning rendered on top of the pages will be catered to the user's respective community, city, and state's covid rules.
- The home page will have sections or buttons such as:
    - Post: for posting an event
    - Help: for accessing helping guides or tutorials about the software
    - Saved: for a collection of saved posts, events, jobs, ….
    - Message: a message section with all the messages that happened between the user and other accounts
    - Notifications: notifications section with all the notifications the user chose to be notified about and any other important notifications the user should be aware of
    - Manage posts: if the user ever posted anything – those posts would be here, and the user can go to them and manage them
    - Events: to check events posts
    - Jobs: Check job posts
    - Life hacks: Check life hacks posts posted by anyone
    - Local politics: Local politics related posts
    - Public meet ups: Posts about any public sports meetups happening in the area
    - Interest groups: new or old interest group posts
    - Planner: Private planner for each user
    - Contact support: To contact customer support
    - FAQ: check already asked and answered questions
    - Account: To check their profile account
    - Settings: To work with settings
    - DMP policy: To check DMP's policies

2.2.1.4 Process and Service:

- Users can click on one of the buttons listed above to check the contents of that link or page
- Posters will have the choice to view the pages as a poster or as a user, but when viewing them as a user, they can't sign up for events or request to join a group.
- When viewing as posters, posters will have the following sections with their functionalities.
    - Post:

- Review applications/requests/registered users:
- Messages:
- Planner:
- Manage posts:
- Account:
- Settings:
- DMP policy:
- Contact support:
- FAQ:
- Help:

- Users will also have access to post.
- Users can click on the post button to post on the user-post-allowed sections
- Users will click on the manage posts button to view, check, and manage the posts they have posted and other user's responses to them
- Users will click on the help button to view and access helping guides about the software, tutorials on how to use the software, and other information about the software
- Each post from any of the post sections will have the following buttons under the post header that users can click on and utilize:

    - Comment:
    - Reply:
    - Like:
    - Dislike:
    - Save post:
    - Message: This button's availability will depend on the poster's choice.

- The planner will have a section or bar for the year on top, followed by a bar for the month, and then days of the week.
- Users will add tasks by clicking on the 'add a task' button or clicking anywhere on any of the day's timeframe box
- Users can use built in functionality to edit a task on a planner
- To change a tasks date, users can edit the tasks settings or just simply drag their task schedule and change its time or date to another day or time
- Life hacks post will have a copy-paste button right next to the post text, so users can copy-paste the post text
- Users will click on a section from events, jobs, life hacks, interest groups, meetups, and local politics, to view the sections post. *(Reference – See - Use-Case ID # 6 on System Proposal Part 1+2)*

- Users will click on the comment, like, dislike, save post, message, or reply button found under each post in the posts section or on the specific posts page
- The message button will be optional and up to the administrator's choice
- Users will click on a specific event to view more information about the event and to go to the specific event's page
- Users will click on sign up for event or follow a link provided in the page to sign up using another website
- once signed up, users can click on 'add to planner' to add an event or any other topic to their planner
- Users will click on "sort by" to sort the posts using available and incorporated attributes
- Users will click on a specific post to view more information about the post, such as detailed descriptions, the date and time it was posted, the administrator's info, etc.
- Users will click on the contact support button to contact support and report issues or figure out how to fix an issue
- Users will click on the FAQ button to see if their questions have been asked and answered before
- Users will click on the account button to go to their accounts page or check their profile
- User will click on the settings button to configure settings for the software
- Users will click on the DMP policy button to read or learn about DMP's policies
- The system will check and remove users reported as being inappropriate after confirming reports
- Posters who post in the following sections (events, jobs, interest groups, and public meetups) will be notified to include their respective covid 19 requirements in their post
- Users who sign up for the following posts (events, jobs, interest groups, and public meetups) will be notified to follow current covid 19 precaution rules as required by their respective states and communities. *(Reference – See - Use-Case ID # 3 on System Proposal Part 1+2)*

2.3 Non-functional Requirements and Design Constraints

This software is currently facing a few constraints in the areas of resource feasibility and legal and contractual feasibility. As of right now, I do not have all the resources, skills, and support I need to fully develop the software with all of its intended functionalities. That also affects the schedule feasibility of the DMP software development. Additionally, I have not yet worked out the exact steps needed for the legal and contractual agreements. I need more credible information in that area, which I am working on. Otherwise, the DMP is able to be developed and work as intended. At most, the MVP is feasible and considering the scope of the project, the MVP is pretty good.

Below are some of the main constraints and non-functional requirements:

2.3.1 Main Constraints

- Setting up the legal and contractual agreements for implementing the local politics participation section
- Acquiring all the information and skills necessary to implement a strong security for the system
- Acquiring all the money needed to implement the software
- It may be difficult to attract users who are willing to wait three days for the background check to be completed before they can have an account

2.3.2 Main nonfunctional requirements:

- The project will be done with minimum cost as much as possible and by using as much off the shelf resources
- The software will be implemented as a website, Desktop, and Phone application
- System support feature will require the hiring of customer service employees, so that feature will be implemented after the software becomes popular and successful
- Once users arrive at the DMP, the system should load and display the login and sign-up page within half a second.
- The system will be built to operate fast (1 sec max) except for operations such as sending messages with large files or submitting applications.
- Users can choose to create an account as a poster or regular user
- The background check will take about 3 days, so users have to wait 3 days until they can get a valid DMP account.
- The system's administrators can inspect users' and poster's actions and if necessary, they can remove or communicate with posters and users.
- Posters can be company hirers, independent posters, or self-employed posters
- Posters will be able to post on most of the sections except for the politics and news pages, which acquire their post from other websites.
- The post category input, which posters have to fill out when submitting a post request will be used to check if the post is valid. (References – see UML page/link – 32, and UI page/link – 69)
- Users can also post but only as an independent or self-employed posters with limited sections and functionalities
- Users or posters can report any inappropriate activity to the DMP administrators. (See appendices section – page - 73 - for sample inappropriate activity report template)
- The system will provide a help menu, that users can use as helping guides on how to use and navigate through the application

- While all the sections have almost similar display properties and features, the planner and the local politics section will have somewhat different display properties and features
- Since the first few months of operation, the software will be operating with a few customer-support, it will provide users with a help menu and FAQs page. We will also give them a chance to write their complaint, tow which they can expect a response within two weeks. (See appendices – page 73 – for report template and page/link - 68 for FAQs page UI)

2.3.3 Main Feasibility Assessment concerns

- The software is currently trying to find solutions for resource, security, and legal and contractual feasibility risks, which are the main constraints
- The security feasibility risk is brought about by lack of resource and skills on developing a strong security system
- Lack of resource and skills also stem from lack of money and related experience.
- Currently the developers have not yet fully worked out the legal and contractual requirements. Therefore, there is some risk there, which is associated with implementing the news and local politics section as it was fully intended, which is one of the main goals for developing the DMP.
- Depending on how the risks are mitigated, the development time could 3 to 6 months, affecting the schedule feasibility

2.4 System Evolution

This software has the potential to be modified and include a lot of additional functionalities in the future. But first it needs to get a good number of users and its capabilities needs to be assessed using the usage data. The following modifications should be added to version 2 and 3.

2.4.1 Version 2 changes:

- In addition to the current messaging communication method, add features to make it possible to call people using audio or video for various purposes such as meeting and chatting.
- When commenting on post, there will be a section that users have to fill out, which asks for the expertise level of the user in that area. So, when other users see a user's post or comment, they can click expertise level, which will show them how experienced the user is in that particular field. The system's security can validate the user's expertise level, although it will be difficult to check and validate all types of expertise.
- Updated covid 19 precaution requirements and measures (suggested before user commits to a post) displayed on top of each page. The posted message can be updated per the respective community's covid rules and changes.
- If any similar informatic displays are needed or a new epidemic arises, the home page should display the warning message along with the guidelines needed for combative measures

2.4.2 Version 3 changes:

- Step up the level to include not only local events and posts but also city level events as well as posts
- Add social media sections where users can have all their social media accounts under one social media section with subsections for the different social media accounts. Most of the sections and the planner section should be linked with this. For example, the planner can be updated based on events, meetings, and schedules from the social media as well

2.5 Document Outline

In the rest of this document, there will be organized descriptions of the following:

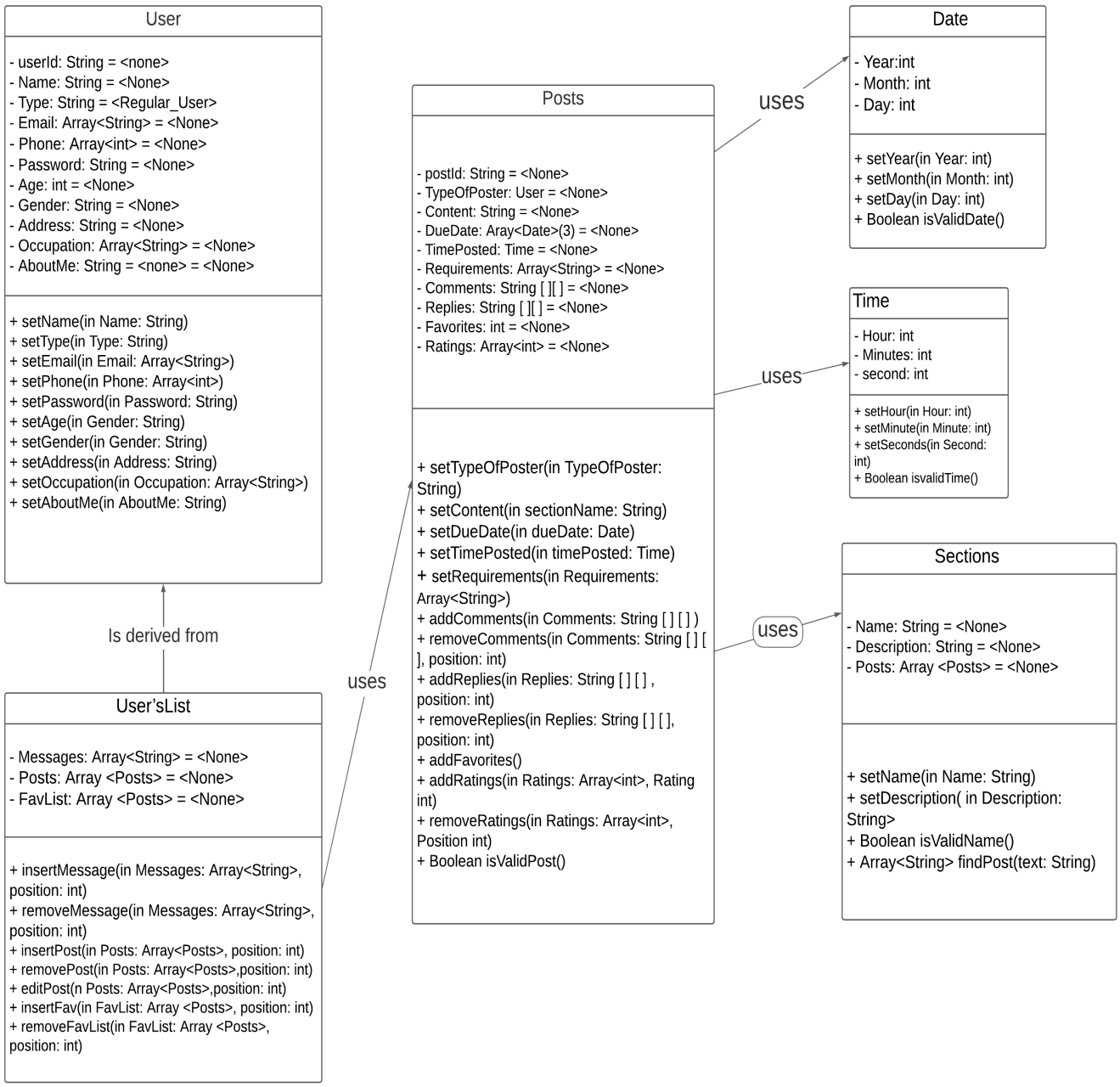
1. Structural Model: This section will show the class diagrams as well as the metadata to describe each classes attributes and operations
2. Architecture design: This section will show the architectural design and infrastructural model as well as provide the components requirements and security system plans needed for proper implementation of the software architecture.
3. User Interface: This section will show the main user interface diagrams and prototypes as well as give descriptions about each UI
4. Appendices: This section will have the glossary, bibliography, and supporting documents that can be used as references or for further research about this software development

3.0 Structural model

3.1 Introduction

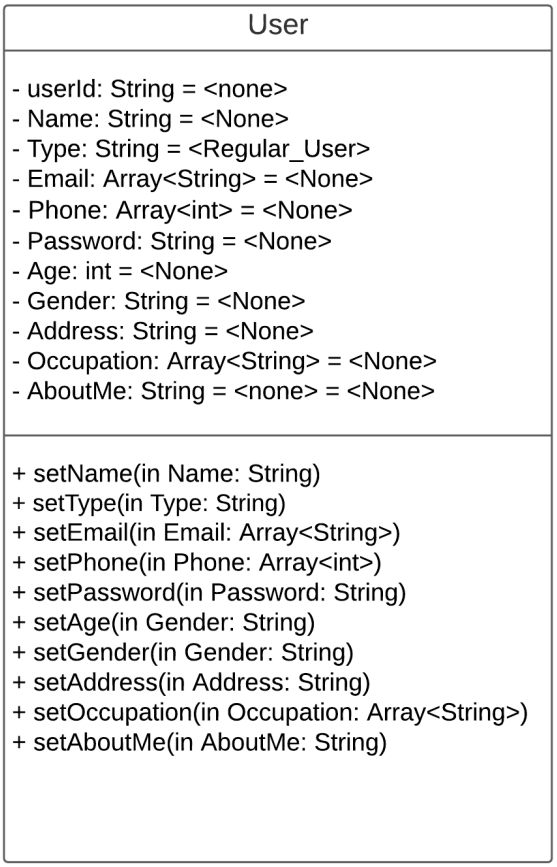
This section gives the class diagram model for the DMP software design and development. The software will have a class for User, Sections, User's Lists, Posts, Date, and Time. Following the UML diagrams, the attributes and operations are described using tables.

3.2 Class Diagrams



3.3 Metadata

Table 1 – User class



1. **Description**: Represents each user of the software
2. **Visibility**: Public
3. **IsAbstract**: No

Attributes:

Attribute Name	Description	Data Type	Is Derived	Is Read Only	Visibility	Multiplicity	Default Value
Name	Last, First Name	String	No	No	Private	1	None
Type	Regular user or Poster	String	No	No	Private	1	None

Email	Users email address, also used for logging in	String	No	No	Private	1…3	None
Password	User's login password	String	No	No	Private	1	None
Age	User's age	int	No	No	Private	1	None
Gender	User's gender	String	No	No	Private	1	None
Address	Users address	String	No	No	Private	1	None
Occupation	Users occupation	String	No	No	Private	1..5	None
AboutMe	A little description about the user	String	No	No	Private	1	None

Operations:

Name	setName
Description	Used to set the name of the user when registering or editing their name
Return Types	None
Parameters	Name Direction: In String Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline

Customer registers with a name (first and last) and can also later change their name in their profile settings.

Enter name

User enters name

Name set to input

Operations:

Name	setType
Description	Used to set the type of user the registerer wants to be. They can be regular user or a poster. By default, the type of user is set to be a regular user.
Return Types	None
Parameters	Type Direction: In String Default: Regular_User
Visibility	Public
Scope	Instance
Is Query	No

Processing Outline:

Customer chooses the type of user they want to be when they register. They can either be a regular user or a poster.

Steps:

Customer chooses type

Customer type is decided

Operations:

Name	setEmail
Description	Customer inputs their email when they register. If they have more than one email, they select one as the main email.
Return Types	None
Parameters	Email Direction: In Array<String> Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

- ➔ User enters their email
- ➔ If user has more than one email and they want to enter them, they enter those as well
- ➔ User chooses one email as the main email for contact but can use another for signing in.

Operations:

Name	setPhone
Description	Customer inputs their phone number when they register.
Return Types	None
Parameters	Phone

	Direction: In Array<int> Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

➔ User enters their phone number

➔ Their phone number is set to that

Operations:

Name	setPassword
Description	Customer sets their password when they register, they can also later change their password in their profile settings.
Return Types	None
Parameters	Password Direction: In String Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

User sets their password when they sign up. They can also later change their password using their profile settings

➔ Request for changing password

➔ Enter old password

If (password matches)

Get new password

If (password doesn't fulfill the requirements or password is weak)

Ask to reenter password

Else

Set new password

Else

Ask to renter password three times before locking of asking for security questions

Operations:

Name	setAge
Description	During registration, user enters their age information, and if they are 18 years or older, their age is set. Their age is also updated as the years go by.
Return Types	None
Parameters	Age Direction: In Int Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

Registration:

User enters their age

If (age is older than 18)

Set user age

Else

Notify user only 18 years and above can register

Operations:

Name	setGender
Description	User fills their gender information, and their gender is set.
Return Types	None
Parameters	Gender Direction: In String Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

Registration:

User selects their gender

User's age is set

Operations:

Name	setAddress
Description	User enters their current address information and if the address is valid, their address is set that. User can also later change their address, if user moved.
Return Types	None
Parameters	Address Direction: In

	String Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

Registration/ changing address:

User enters their address information

If (address is valid)

Accept address

Else

Ask to reenter a valid address

Operations:

Name	setOccupation
Description	User enters their current occupation information. If they have more one current occupation, they can enter up to five occupations.
Return Types	None
Parameters	Occupation Direction: In Array<String> Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

Enter current number of occupations

While (number of occupations > 0)

Enter occupation

Number of occupation - -

Operations:

Name	setAboutMe
Description	User enters information about them to describe themselves, their interests, hobbies, plans, goals, or whatever they want to enter.
Return Types	None
Parameters	AboutMe Direction: In String Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

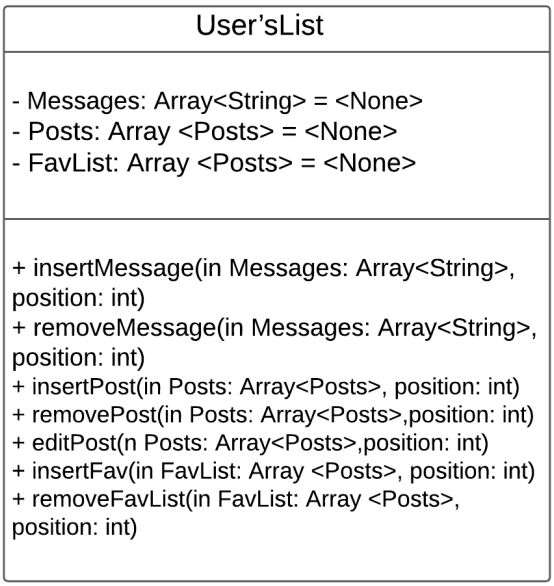
First time/ changing older version:

Enter text in about me section

If (text is within allotted number of characters)

User's about me section is set to that

Table 2: User's List



1. **Description:** Represents user's messages, posts, and favorited posts. These are shown in the messages, manage posts, and favorites sections as well as used in various other methods
2. **Visibility:** Public
3. **IsAbstract:** No

Attributes:

Name	Description	Data Type	Is Derived	Is Read Only	Visibility	Multiplicity	Default Value
Messages	A list of user's messages	Array<String>	No	No	Private	0…*	None
Posts	A list of User's posts	Array<String>	Yes	No	Private	0..*	None

FavList	A list of user's favorited posts	Array<String>	Yes	No	Private	0..*	None

Operations:

Name	insertMessage
Description	Used to add a message to the list of messages when user either receives a message or sends a message.
Return Types	None
Parameters	Messages Direction: In Array<String> Default: None Position Int Default: None
Visibility	Public
Scope	Classifier
Is Query	No

Processing outline:

User receives or sends a message.

The message is added to the user's messages list

Operations:

Name	removeMessage
Description	When user deletes a message, the message is removed.

Return Types	None
Parameters	Messages Direction: In Array<String> Default: None Position Int Default: None
Visibility	Public
Scope	Classifier
Is Query	No

Processing outline:

User wants to remove a message

User clicks delete message and the message is deleted

Operations:

Name	insertsPost
Description	User posts a post and the post is added to the list of user's posts
Return Types	None
Parameters	Post Direction: In Array<Posts> Default: None Position Int Default: None
Visibility	Public

Scope	Classifier
Is Query	No

Processing outline:

User posts a post

The post is added to the user's posts list

Operations:

Name	removePost
Description	User deletes one of their posts, then the post is removed from the user's posts list
Return Types	None
Parameters	Post Direction: In Array<Posts> Default: None Position Int Default: None
Visibility	Public
Scope	Classifier
Is Query	No

Processing outline:

User deletes one of their posts

The post is removed from the user's posts list

Operations:

Name	editPost

Description	User wants to edit one of their posts, then the user goes ahead to that post and edits some or all of its contents, requirements and/or due date.
Return Types	None
Parameters	Post Direction: In Array<Posts> Default: None Position Int Default: None
Visibility	Public
Scope	Classifier
Is Query	No

Processing outline:

User wants to edit one of their posts:

User navigates to the specific post

User edits some or the posts contents

User confirms edit

Operations:

Name	insertFav
Description	User favorites a post and that post is added to the user's favorites list
Return Types	None
Parameters	FavList Direction: In

	Array<Posts> Default: None Position Int Default: None
Visibility	Public
Scope	Classifier
Is Query	No

Processing outline:

User favorites a post

That post is added to user's favorites list

Operations:

Name	removeFavList
Description	User deletes a post from their favorite list and the post is removed from the user' s favorites list
Return Types	None
Parameters	FavList Direction: In Array<Post> Default: None Position Int Default: None
Visibility	Public
Scope	Classifier
Is Query	No

Processing outline:

User wants to delete a post from their favorite list

User clicks delete button next to post in favorite list

The post will no longer be in their favorite list unless user's favorites that post again

Table 3 – Posts class

Posts
- postId: String = <None> - TypeOfPoster: User = <None> - Content: String = <None> - DueDate: Aray<Date>(3) = <None> - TimePosted: Time = <None> - Requirements: Array<String> = <None> - Comments: String [ ][ ] = <None> - Replies: String [ ][ ] = <None> - Favorites: int = <None> - Ratings: Array<int> = <None>
+ setTypeOfPoster(in TypeOfPoster: String) + setContent(in sectionName: String) + setDueDate(in dueDate: Date) + setTimePosted(in timePosted: Time) + setRequirements(in Requirements: Array<String>) + addComments(in Comments: String [ ] [ ]) + removeComments(in Comments: String [ ] [ ], position: int) + addReplies(in Replies: String [ ] [ ] , position: int) + removeReplies(in Replies: String [ ] [ ], position: int) + addFavorites() + addRatings(in Ratings: Array<int>, Rating int) + removeRatings(in Ratings: Array<int>, Position int) + Boolean isValidPost()

1. **Description:** Used for representing each post
2. **Visibility:** Public
3. **IsAbstract:** No

Attributes:

Name	Description	Data Type	Is Derived	Is Read Only	Visibility	Multiplicity	Default Value
postId	The Id of the post	String	No	Yes	Private	1	None
TypeOfPoster	The type of poster or user	User	Yes	Yes	Private	1	None
Content	The post itself or content of the post	String	No	No	Private	1	None
DueDate	The posts due date, if it has any, after which it will be removed from the section's posts list	Date	Yes	No	Private	1..10	None
TimePosted	The date and time the post was posted on	Time	Yes	Yes	Private	1	None

Requirements	A list of the requirements for engaging with this post or applying to it, if it has any	Array<String>	No	No	Private	1…50	None
Comments	The posts comments	Array<String>	No	No	Private	0..*	None
Likes	The posts likes	int	No	No	Private	0..*	None
Favorited	The number of favorites the post gets	int	No	No	Private	0..*	None
Ratings	The posts ratings	Array<int>	No	No	Private	0..*	None

Operations:

Name	setTypeOfPoster
Description	Each post requires the type of poster to be identified so that limitations will be applied accordingly
Return Types	None
Parameters	TypeOfPoster Direction: In User Default: None
Visibility	Public
Scope	Instance

Is Query	No

Processing outline:

User enters their account type when trying to post

If (type is valid for the post section)

Submit post

Else

Request to edit post request inputs

Operations:

Name	setContent
Description	User enters the content of the post they are trying to post, and the content is set to that
Return Types	None
Parameters	Content Direction: In String Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

User enters the content of the post

The content of the post is set to that

Operations:

Name	setDueDate
Description	User enters the due date for the post, if it has any, and the due date is set to that
Return Types	None
Parameters	DueDate Direction: In Date Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

User enters the dude date for the post or the due dates

The due date/s for the post is/are set accordingly

Operations:

Name	setTimePosted
Description	The time posted is calculated by the system using the current time. As soon as the user submits the post form, the time posted is set to that time.
Return Types	None
Parameters	timePosted Direction: In

	Time Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

User submits a post form

The system records the time of the post

The time the post is posted is set to that time

Operations:

Name	setRequirements
Description	User enters requirements and the requirements are set to that
Return Types	None
Parameters	Requirements Direction: In Array<String> Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

User enters the requirements for a post

The posts requirements are set to that

Operations:

Name	addComments
Description	Users comment on a post and the comments are added to the end of the comments list, which is a 2d array. The first array parameter is used for comments and the second one is used for replies.
Return Types	None
Parameters	Comments Direction: In String [ ] [ ] Default: None
Visibility	Public
Scope	Classifier
Is Query	No

Processing outline:

comment on a post, the comment is added to comments list.

The comment's two-dimensional string array's first parameter saves the list of

comments and the second saves the list of replies to that comment

Only two threads of comments and replies are allowed.

Users cannot reply to a reply of a comment. But they can comment to a reply of a comment

Operations:

Name	removeComments
Description	Users deletes their comment. This method finds the user's comments from the list and deletes their comment. The replies to the comment are not deleted. Only this comment will be marked deleted.
Return Types	None
Parameters	Comments Direction: In String [ ] [ ] Default: None Position Int Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

User deletes their comment

The comment will be marked [Deleted]

All other connected treads are not affected

Operations:

Name	addReplies
Description	Users reply to a comment. The replies are added to replies list and then to the second parameter of the corresponding comments list. The first parameter in the 2d array is for replies and the second is for replies of

	replies. Users cannot reply to a reply of a reply. The trend stops after two stages.
Return Types	None
Parameters	Replies Direction: In String [ ] [ ] Default: None Position Int Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

User replies to a comment

The reply is added to that comment's reply list.

Operations:

Name	removeReplies
Description	Users deletes their reply. This method finds the requested reply from the reply list and deletes the reply. Connected threads are not affected.
Return Types	None
Parameters	Replies Direction: In String [ ] [ ] Default: None

	Position Int Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

User deletes their reply

The reply will be marked [Deleted]

All other connected treads are not affected

Operations:

Name	addFavorites
Description	Keeps a count of the number of favoritism a post receives.
Return Types	None
Parameters	None
Visibility	Public
Scope	Classifier
Is Query	No

Processing outline:

Users favorite a post.

The number of favorites for that post is added and updated.

Operations:

Name	addRatings
Description	Gets a user's rating value and adds it to the list of ratings for that post.
Return Types	None
Parameters	Ratings Direction: In Array<Int> Default: None Rating Int Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

User rates a post

The rating value is added to the list of ratings.

Operations:

Name	removeRatings
Description	Used to delete or change a rating. When a user deletes a rating or wants to change their ratings, their original ratings is removed from the list. If they are changing a rating, their new rating is readded.

Return Types	None
Parameters	Ratings Direction: In Array<Int> Default: None Position Int Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

User deletes their rating

The rating value is deleted from the ratings list.

User rates the same post again

Their original ratings are deleted from the ratings list.

The new rating value is added to the ratings list

Operations:

Name	isValidPost
Description	Checks to see if a post is valid or not
Return Types	Boolean
Parameters	None
Visibility	Public
Scope	Classifier
Is Query	Yes

Processing outline:

User submits a post request

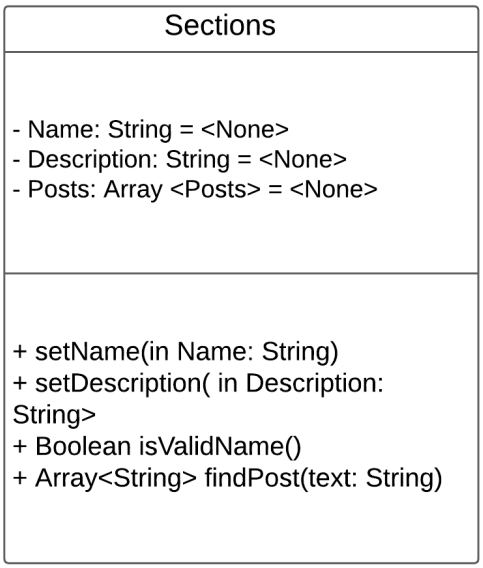
If (post is valid)

Post

Else

Highlight invalid inputs and request for edit before posting

Table 4 – Sections class



1**. Description:** Used for representing each section with a name and a description for each

2**. Visibility:** Private (for internal use only)

3**. IsAbstract:** No

Attributes:

Name	Description	Data Type	Is Derived	Is Read Only	Visibility	Multiplicity	Default Value
Name	Name of the section	String	No	Yes	Private	1	None

Description	The description for the section	String	No	Yes	Private	1	None
Posts	A list of posts under this section	Post	Yes	No	Private	0..*	None

Operations:

Name	setName
Description	Used to set the name of the section. The name is used to navigate to a specific section or identify certain posts for various functions.
Return Types	None
Parameters	Name Direction: In String Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

User wants to post a post:

The section name is identified from the category input

The post is directed to that section page

Operations:

Name	setDescription
Description	Used to set the description of a section. The description will be used as informative

	text in places where a description about a section is required
Return Types	None
Parameters	Description Direction: In String Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

For each section a description is set to describe what the section is about

This will mostly be done by the developers

The description will be used in some places to describe what a specific section is about

Operations:

Name	IsValidName
Description	Used to check the name of the section and only allow valid names to be used as a section.
Return Types	Boolean
Parameters	None
Visibility	Public
Scope	Classifier
Is Query	Yes

Processing outline:

When a name is set as a section, the name is tested to check if it is a valid or accepted name

Operations:

Name	findPost
Description	Used to find matching posts ids for user search text using a string input. It will return an array of post ids saved as string. Then those posts can be shown to the user for the user to choose which one they want.
Return Types	Array<String>
Parameters	Text String Default: None
Visibility	Public
Scope	Classifier
Is Query	Yes

Processing outline:

User searches for a post

While (posts in section)

If (user text somewhat matches post content)

Add the posts id to string array list

Table 5 – Date Class

1**. Description:** Used for representing date, for posted date, due date, todays date, etc

2**. Visibility:** Public

3**. IsAbstract:** No

Attributes:

Name	Description	Data Type	Is Derived	Is Read Only	Visibility	Multiplicity	Default Value
Year	The year	int	No	Yes	Private	1	None
Month	The month	int	No	Yes	Private	1	None
Day	The day	int	No	Yes	Private	1	None

Operations:

Name	setYear
Description	Used to set the year number of the date
Return Types	None
Parameters	Year Direction: In Int Default: Current year
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

Administrator or User chooses year or enters year number

The year is set to that year

Operations:

Name	setMonth
Description	Used to set the month number of the date
Return Types	None
Parameters	Month Direction: In Int Default: Current Month
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

User or administrators chooses month or enters the month number

The month is set to that month

Operations:

Name	setDay
Description	Used to set the day of the date
Return Types	None
Parameters	Day Direction: In Int Default: Current Day
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

User or administrators chooses the day or enters the day number

The day is set to that day

Operations:

Name	isValidDate
Description	Used to check the validity of the date and make sure it is not invalid by checking for things such as leap years
Return Types	Boolean
Parameters	None
Visibility	Public
Scope	Classifier
Is Query	Yes

Processing outline:

User or administrator enters a date

If (date is within valid year)

If (date is within valid month)

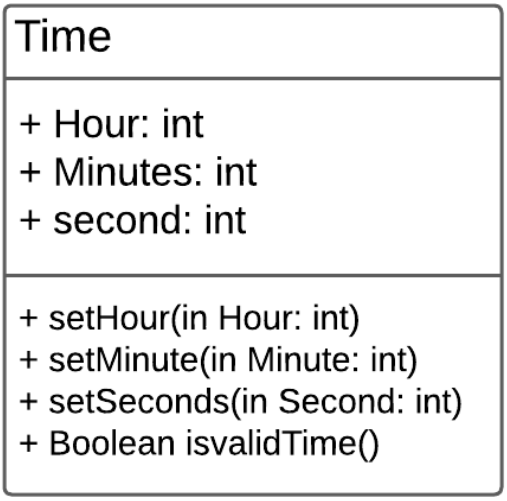
If (date is within valid day)

Return true

Else

Return false

Table 6 – Time class



1**. Description:** Used for representing each post

2**. Visibility:** Public

3**. IsAbstract:** No

Attributes:

Name	Description	Data Type	Is Derived	Is Read Only	Visibility	Multiplicity	Default Value
Hour	The hour	Sections	Yes	No	Private	1	None
Minute	The minute	user	Yes	Yes	Private	1	None
Seconds	The seconds	String	No	No	Private	1	None

Operations:

Name	setHour
Description	Used to set the hour of the time input
Return Types	None
Parameters	Hour Direction: In Int Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

User or administrator chooses or enters the hour input

The hour is set to that

Operations:

Name	setMinute

Description	Used to set the minute of the time input
Return Types	None
Parameters	Minute Direction: In Int Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

User or administrator chooses or enters the minute input

The minute is set to that

Operations:

Name	setSeconds
Description	Used to set the second of the time input
Return Types	None
Parameters	second Direction: In Int Default: None
Visibility	Public
Scope	Instance
Is Query	No

Processing outline:

User or administrator chooses or enters the second input

The second is set to that

Operations:

Name	isValidTime
Description	Used to check the validity of the time input and make sure it is not invalid
Return Types	Boolean
Parameters	None
Visibility	Public
Scope	Classifier
Is Query	Yes

Processing outline:

User or administrator enters a date

If (time is within valid hour)

If (time is within valid minute)

If (time is within valid second)

Return true

Else

Return false

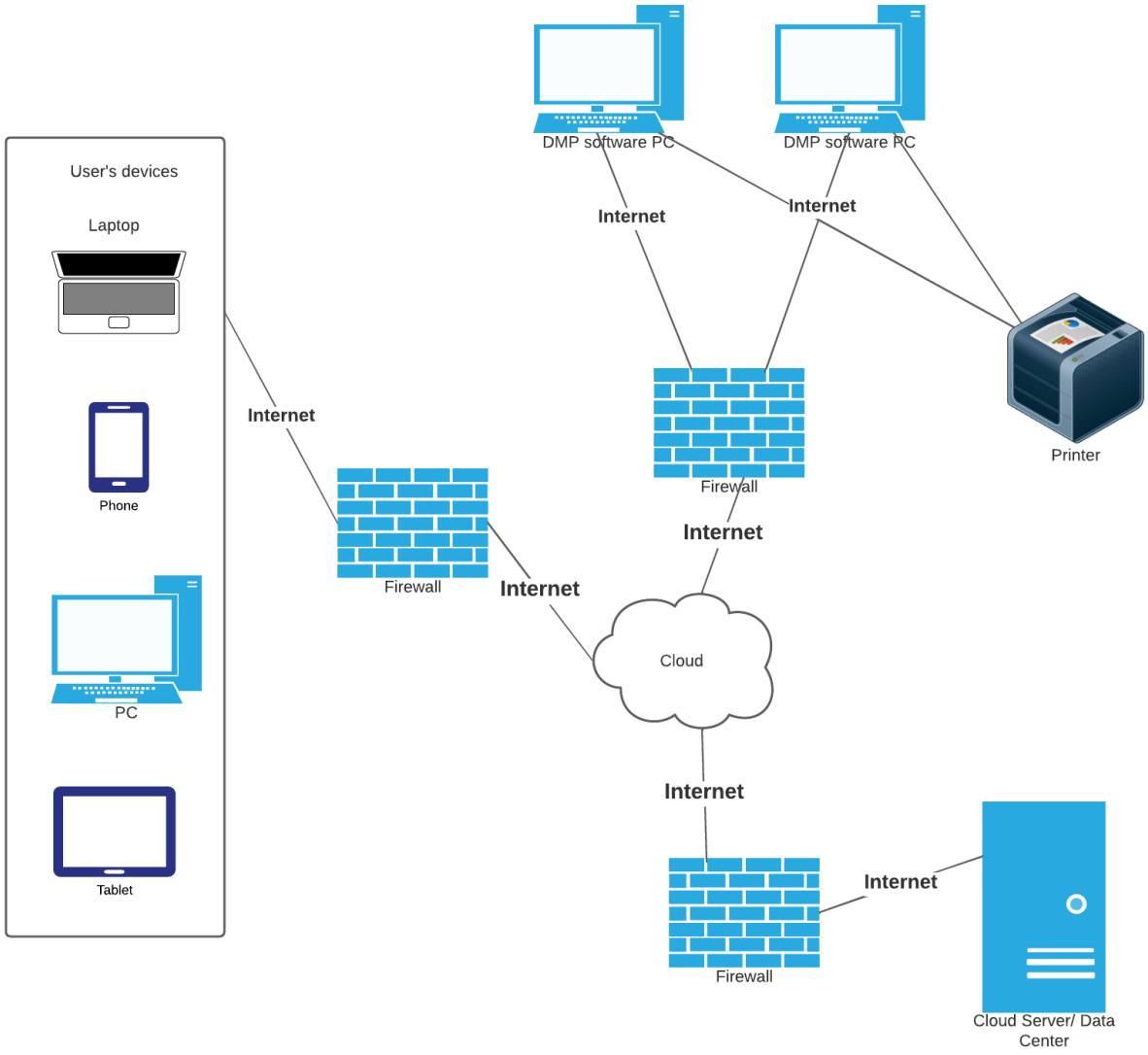
4.0 Architecture Design

4.1 Introduction

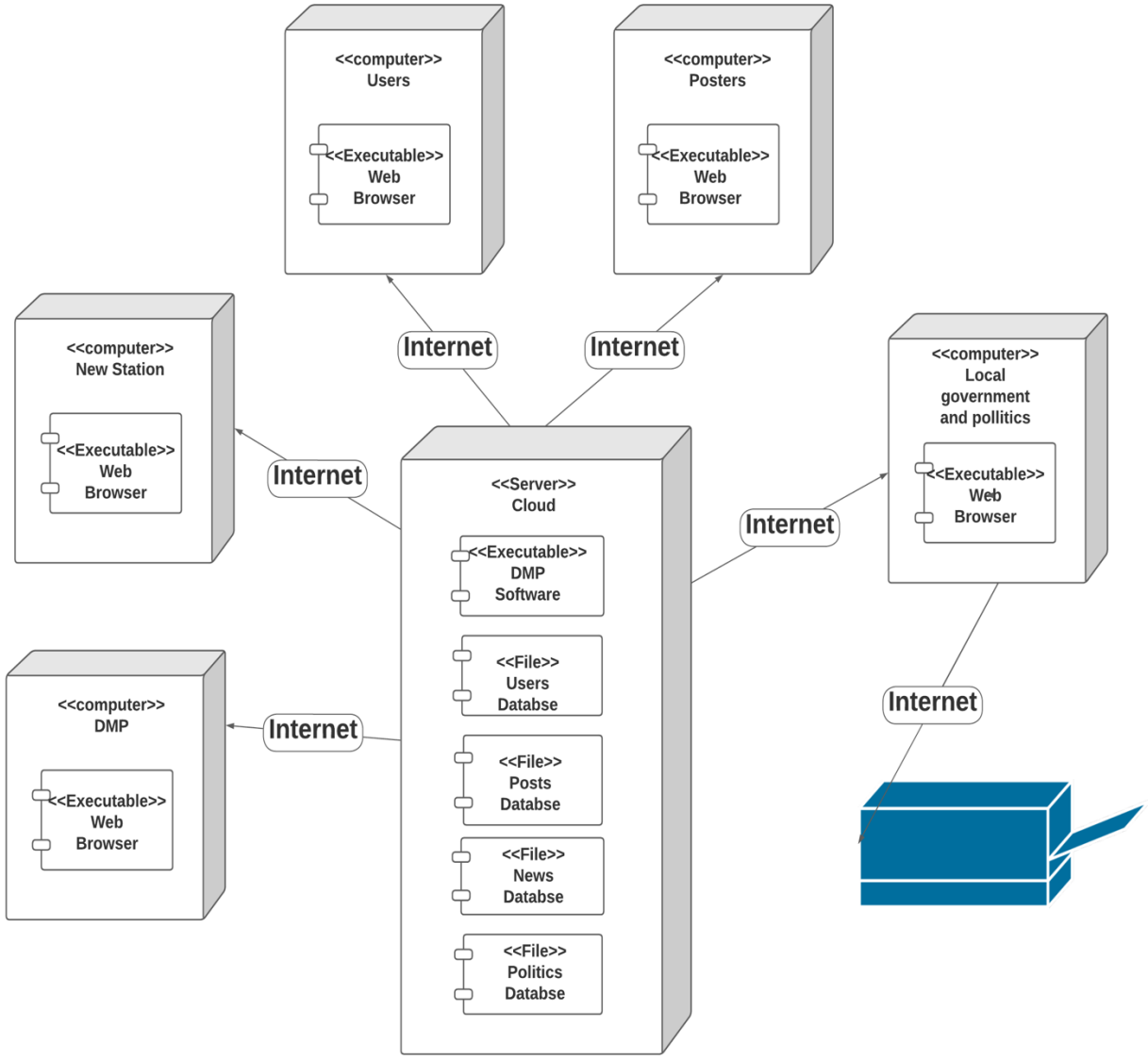
The DMP will be designed to be hosted from a cloud server. Data will then be distributed to different sectors such as users and administrators, using the internet. Since the application will be applicable to Desktops, Websites, tablets, and Phones, users will be able to access the software from all those platforms. The administrators will be using their own printers to print forms and work with some data.

4.2 Infrastructure Model

4.2.1 Deployment Diagram 1: Architectural View



4.2.2 Deployment Diagram 2: Nodes and Artifacts



4.3 Hardware and Software Requirements

The DMP will be using hardware provided through the cloud services. Throughout the applications use, the administrators will be managing the application as needed. But for launching the application, using a cloud serviced hardware would be ideal to reduce startup cost and use a reliable medium.

4.3.1 Required Hardware Components:

- Cloud hardware: For server hosting and database storage, a cloud server is needed, which is probably the best way to start running the DMP. Until the DMP is successful and can run its own server, it will need to rely on paid cloud service for server and database purposes.
- Computers: DMP administrators and staff members will need a working computer preferably a laptop or pc. But tablets or mobile phones should work as well. For on the go, mobile phones should work.
- Computer: News stations and Local government sources will need a computer as well for the DMP to work properly because it will be exchanging data with them through web API via the internet.
- A printer: DMP administrators will need a printer for printing important documents, reports, or working and processing data.
- Users: All types of users will need either a PC, Laptop, Tablet, or Smart phones to be able to access the DMP application or website and use it.
- Backup storage: Administrators will need a reliable data storage backup device. This will help lower storage cost and will be helpful in case of emergency.

4.3.2 Required Software Components:

- Cloud service must be compatible with administrators' browsers to work with and manage storage and hosting settings.
- Sonatype, Nexus Lifecycle or Checkmarx software security testing tools: These tools are important to test the security of the software before deployment. DMP deals with a lot of sensitive data and having a strong security system is essential for the well-functioning of the software. Therefore, testing the software's security system before deployment is one of the first steps in ensuring a strong security for the DMP.
- Web Browser: DMP must be implemented to support a most of the popular web browsers such as Chrome, Safari, Firefox, and Microsoft Edge
- Antivirus: Even if the application is hosted through cloud servers, the administrators should protect their devices to protect any data that is located on their devices and to be able to manage the application properly from their device without causing unexpected or unwanted interruptions to the application users
- DMP must be implemented to support all popular operating systems including Windows 8 or later, Mac Catalina or later, Android 8 or later, and iOS 13 or later

4.4 Security Plan

4.4.1 Security Overview

The DMP might face a lot of security threats. Since the software wants to help people interact and people don't want to be interacting with other people they can't trust, a strong security is needed. Threats could come from users of the application themselves or from outside sources such as hackers. The most significance importance is as stated above, the user's trust for the app. If users don't trust the application for whatever reason, then they are not going to be using it as much or as hoped, which will definitely affect the success and the longevity of the application. The most significant concern is hackers getting user's sensitive information. That would be betraying our users and their trust for us. So, a strong security system is crucial.

Currently the best security measures I am going to employ while building the application and after are the following:

While building the application:

- Make sure the coding is tight and secure with a proper flow and well-organized classes
- Employ secure passes from one iteration to the next and not allow leaks
- Have the least number of bugs as much as possible
- Use code obfuscation
- Run the code through security testing tools such as SAST, DAST, IAST, and RASP. – (See appendices - glossary - page – 72)
- Use best application security software such as Sonatype, Nexus Lifecycle, and Checkmarx

After the software is deployed:

- Stay up to date on learning how to deal with threats and intruders
- Deploy multi-factor authentications for all users
- Identify possible weak areas throughout the various end points such as hardware, users, resource, and operations. Then make sure strong securities are applied to those
- Identify failed threats and prepare for redemptive strikes
- Monitor users' activities and look for any suspicious activities

- Notify user of new device and new location logins and get confirmation from them to identify unauthorized accesses
- Store posts and frequently updated files on personal storage devices and keep devices in secure locations. Delete those files from application after three months. This will help free up space and keep data secure.
- User finger - print equipped devices to protect administrators' devices as well as encourage the same for users by including similar type of message where applicable.
- Train users to avoid clicking unknown links
- Use firewalls
- Use encryption methods for passwords and sensitive data

4.4.2 Security Plan

Threats	**Disruption, Destruction, Disaster**					**Unauthorized Access**		
Components	**Fire**	**Flood**	**Power Loss**	**Circuit Failure**	**Virus**	**External Intruder**	**Internal Intruder**	**Eavesdrop**
Servers	1,2	1,3	4	1,5,6	7,8	9,10,11,12	9,10	4, 8, 15
Client Computers	1, 8	1, 8	8	8	8	5, 6, 9, 10, 11, 12, 13, 14, 15	4, 5, 6, 9, 10, 11, 12, 13	4, 8, 15
Communicate Circuits	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Network Devices	1, 2, 7, 8, 14	1, 3, 7, 8, 14	3, 7, 8, 14	1, 3, 7, 8, 14	7, 8, 10, 11, 12,13, 15	4, 7, 8, 9, 10, 11, 12, 13, 15	4, 5, 7, 8, 9, 10, 11, 12, 13, 15	4,5,6,15
Network Software	1, 2, 7, 8, 14	1, 3, 7, 8, 14	3, 7, 8, 14	1, 3, 7, 8, 14	7, 8, 10, 11, 12, 13, 15	4, 7, 8, 9, 10, 11, 12, 13, 15	4, 5, 7, 8, 9, 10, 11, 12, 13, 15	4,5,6,15
People	1, 2, 8, 14	1, 3, 8, 14	3, 8, 14	1, 3, 8, 14	8,10,11,12,13,14	8,10,11,12,13	4,5,6,8,10,11, 12,15	4,5, 6,15

Controls

1. Disaster Recovery plan
2. Fire alarms in every room, sprinklers in rooms without computers or servers
3. Uninterrupted Power Supply (UPS) on servers and computers
4. Key card access required to enter equipment rooms
5. Lock equipment in place
6. Security cameras and monitors

7. Purchase insurance for equipment
8. Back up data on a server located elsewhere
9. Data encryption
10. Strong password software
11. Virus detection software
12. Anti-spyware software
13. Firewalls around application and database servers
14. Employee training at Homestyle Solutions
15. Access control list (i.e. only authorized employees can access database server)

5.0 User-Interface

5.1 User-Interface Requirements and Constraints

This section has the system navigation diagram and most of the main user interface diagrams. Detailed navigation and user interface information have not been provided because of the scope of the project and space consuming issue of adding more user interfaces. But the basic navigation system is pretty simple. The basic layout goes from home page (planner section) to any other specific section category. Each category has a list of posts that user can interact with. Some of the user's interactions are saved in the favorites, messages, and manage posts list, which are found on top right corner of each page. Users can also access the full list of those from their account page. Posters don't have access to news and politics because they can't post on those sections. Besides that, the navigations are pretty common.

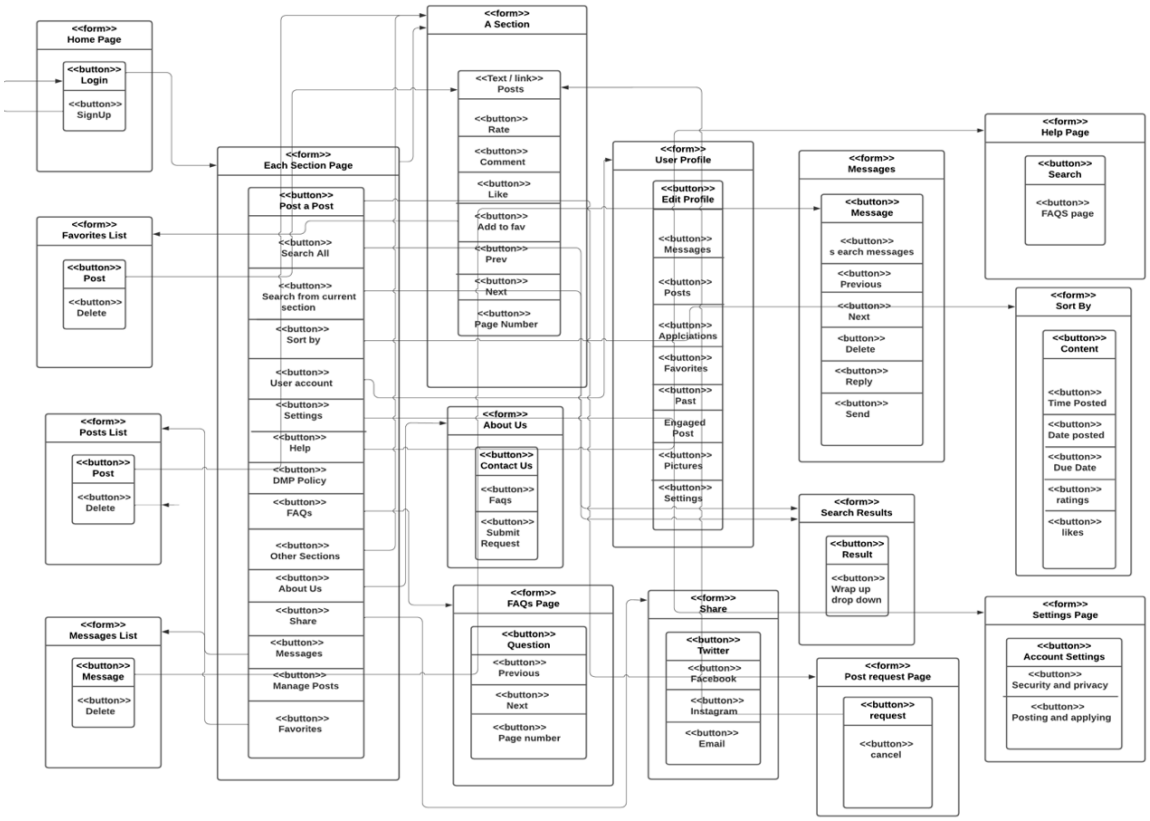
The wireframes portray the basic outlook and feel of the software. Although much is included in each wireframe protype, not everything is covered. Some common and DMP related details would have to be added to the real software. The formal output report section has been omitted because there are no forms to print associated with the application. The application was originally meant to be a paid application but after reconsideration, it was decided to make it a free application. That was done by considering marketing platforms, competing systems, and user attraction for better profit margins. But user may print forms such as job application, documents, directions to meetups, etc. Those will be provided by the posters and users will have to follow the proper steps to print those out.

The navigations system was designed using lucidChart and the wireframes were designed using Pencil application.

Navigation Diagram Link:

https://lucid.app/lucidchart/af3c3b09-db9f-47f3-b90d-e826caf26470/edit?beaconFlowId=EF608CE55E2D1F93&page=0_0&invitationId=inv_a30eabca-1bb7-4b03-9dfb-2642ec413bda#

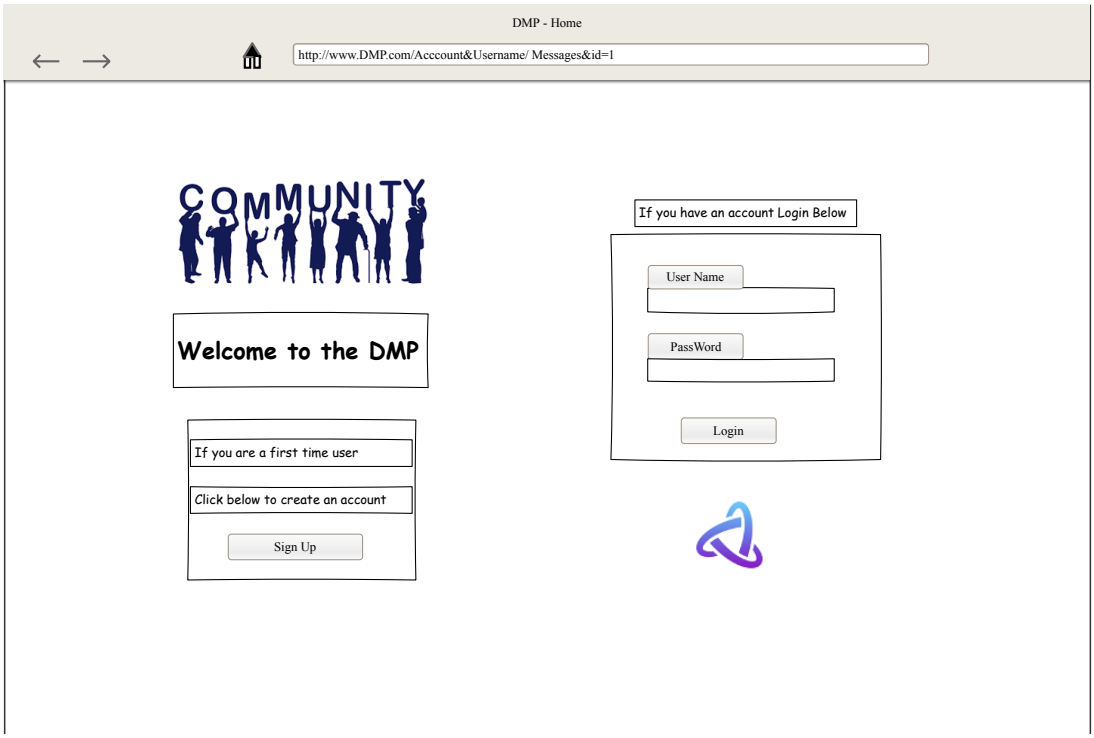
5.2 Window/Screen Navigation Diagram



5.3 UI Wireframes

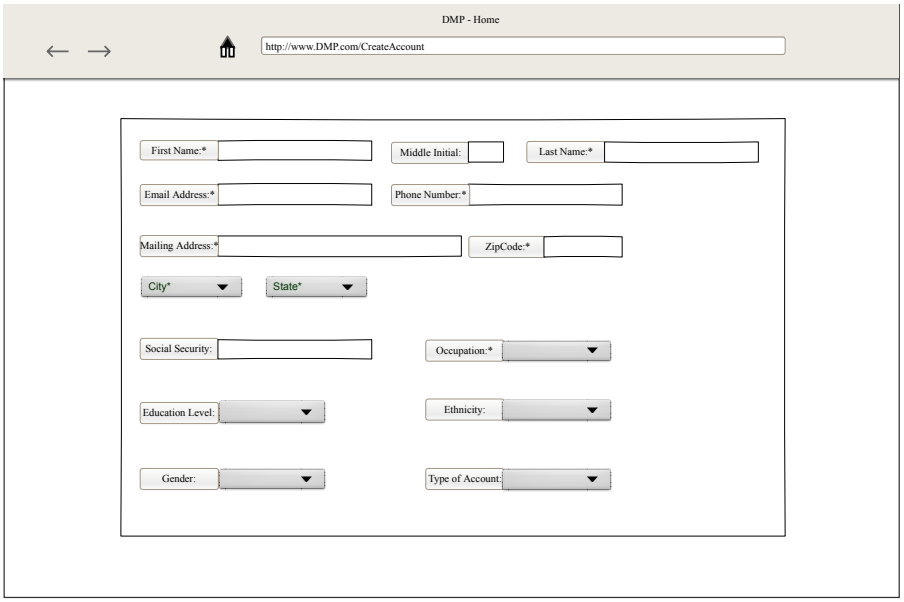
DMP – Home Page UI – Also login page
The following is the welcome page UI, with a login and create account options

Login



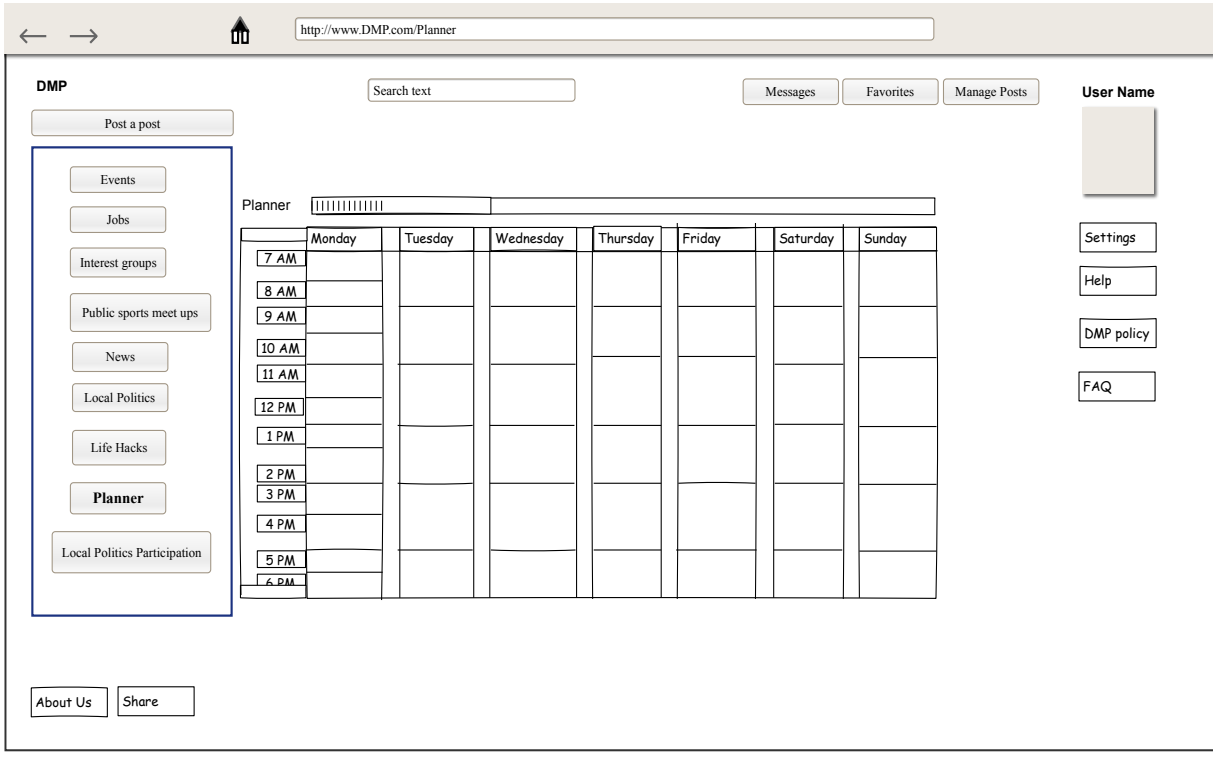
DMP – Create Accounts UI– sign up form

The following UI is the create account page, where users can sign up as a regular user or poster, which they have to specify where it asks for 'Type of account'. After signing up, users have to wait for three days before getting their approval or denial.



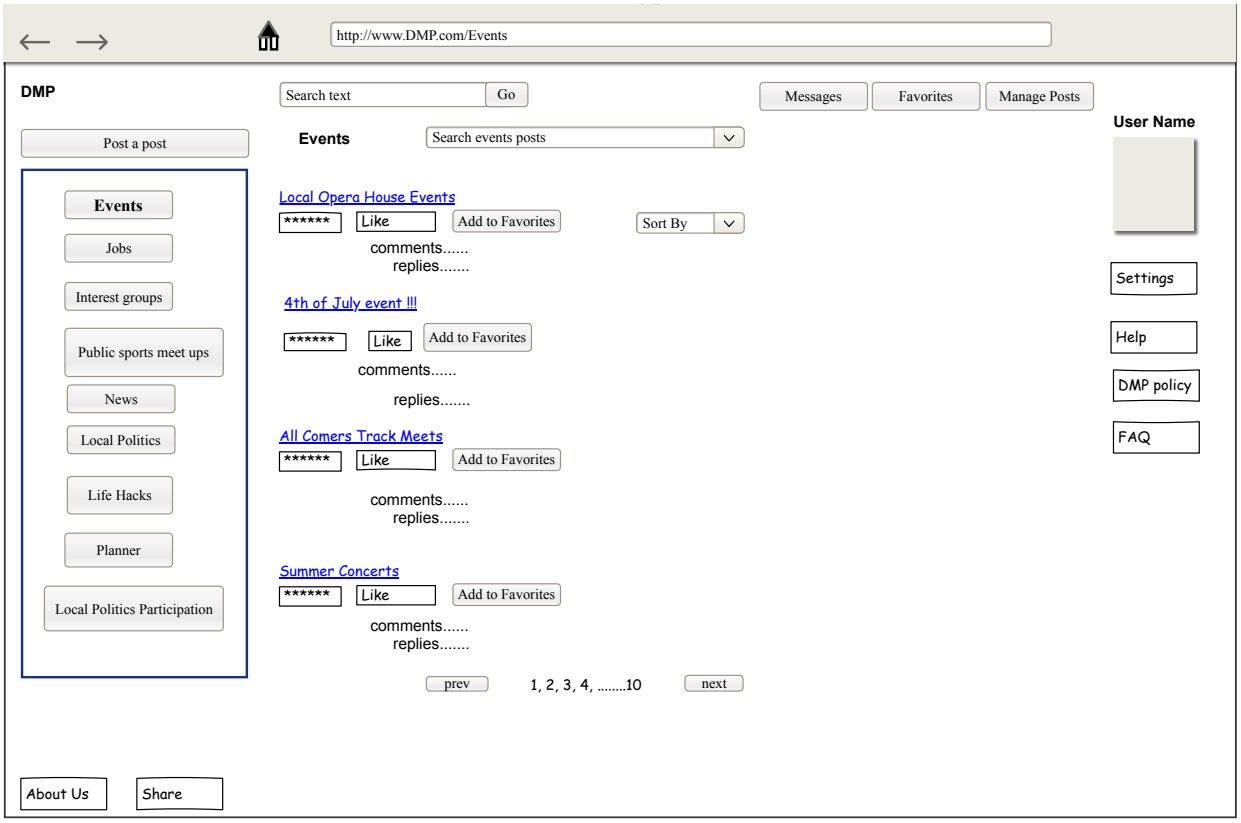
DMP – Home Page – Planner UI

This is the DMP main page or the first page that shows up. It is different for a poster and a regular user. The one below is for a regular user. A main page for a poster is included a few pages down.



DMP – Events Page UI

This is the DMP's events page UI for a regular user.

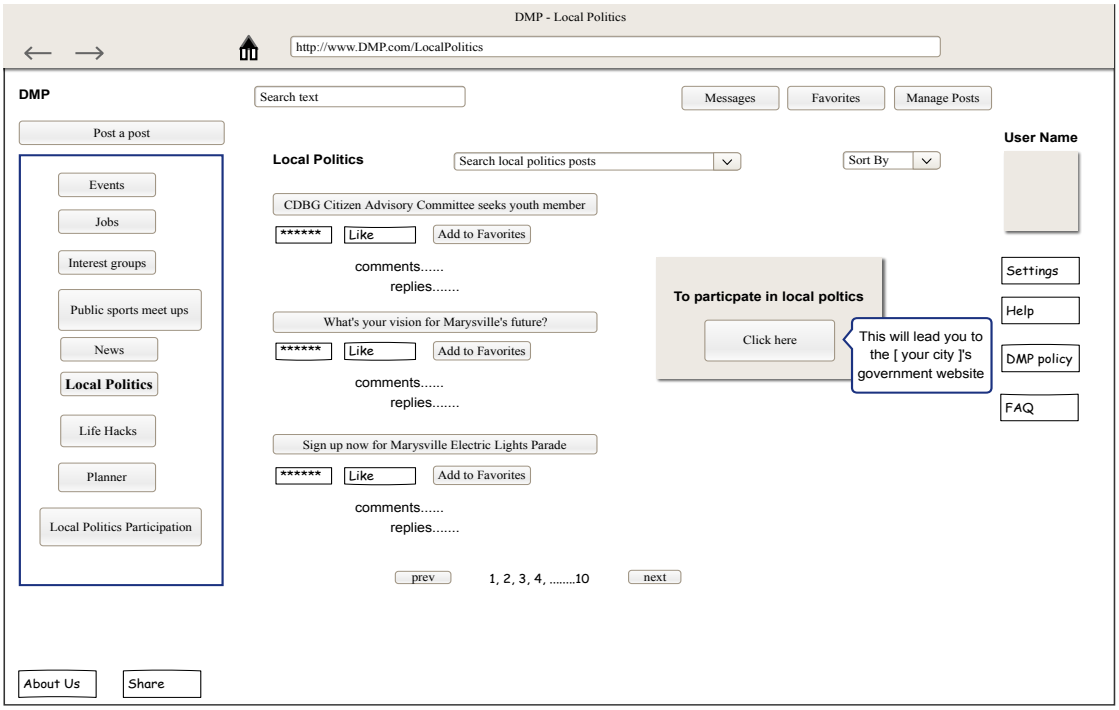


From the UI's above, you can notice that there is a trend for each section. Thus, each section will follow the same format. The news and politics page look a little different but only in terms of their posts and other small details. I have included the politics page UI below because it has the unique connection to the local government website.

Keep in mind, posters don't have this website. Because the administrators handle the politics page. No user post is allowed on the politics page. This is checked when users enter the section for their post. As the UML diagram indicates, the isValidPost method should check for the validity of each post before they are posted.

The following is the DMP's local politics page UI

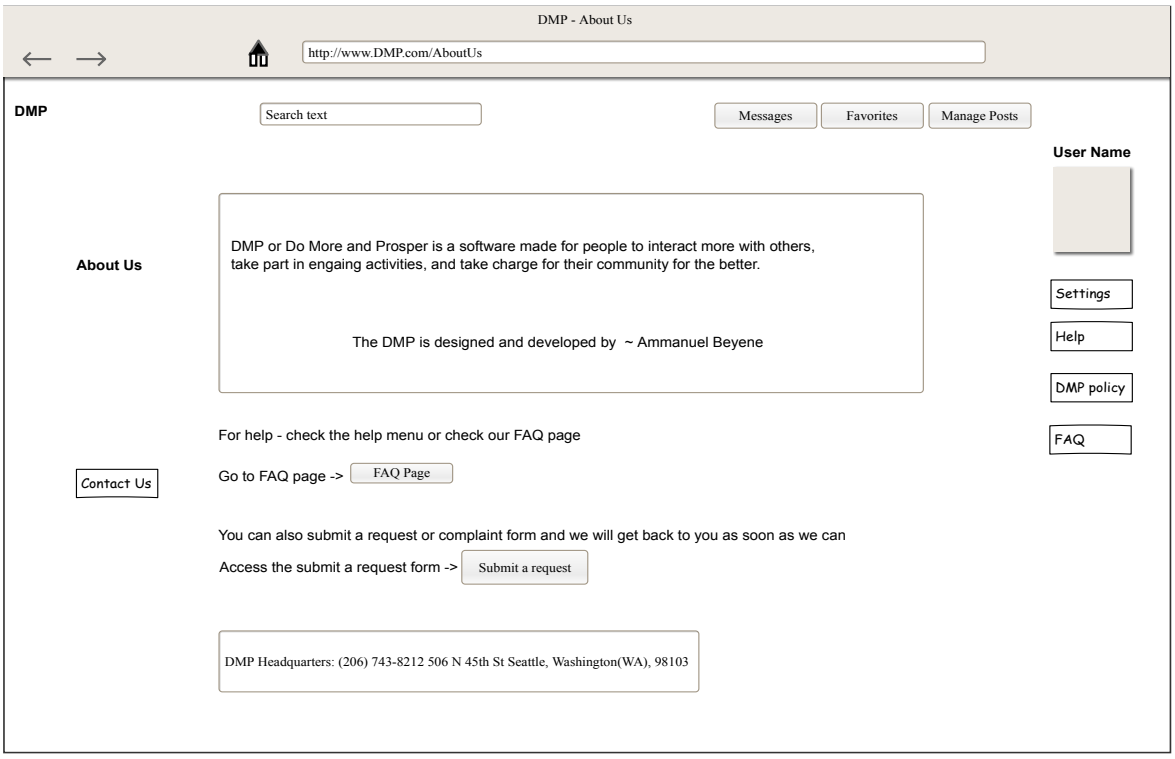
Local Politics



DMP – About Us UI

The following is the About us Page UI of the DMP. It is the same for regular users and posters.

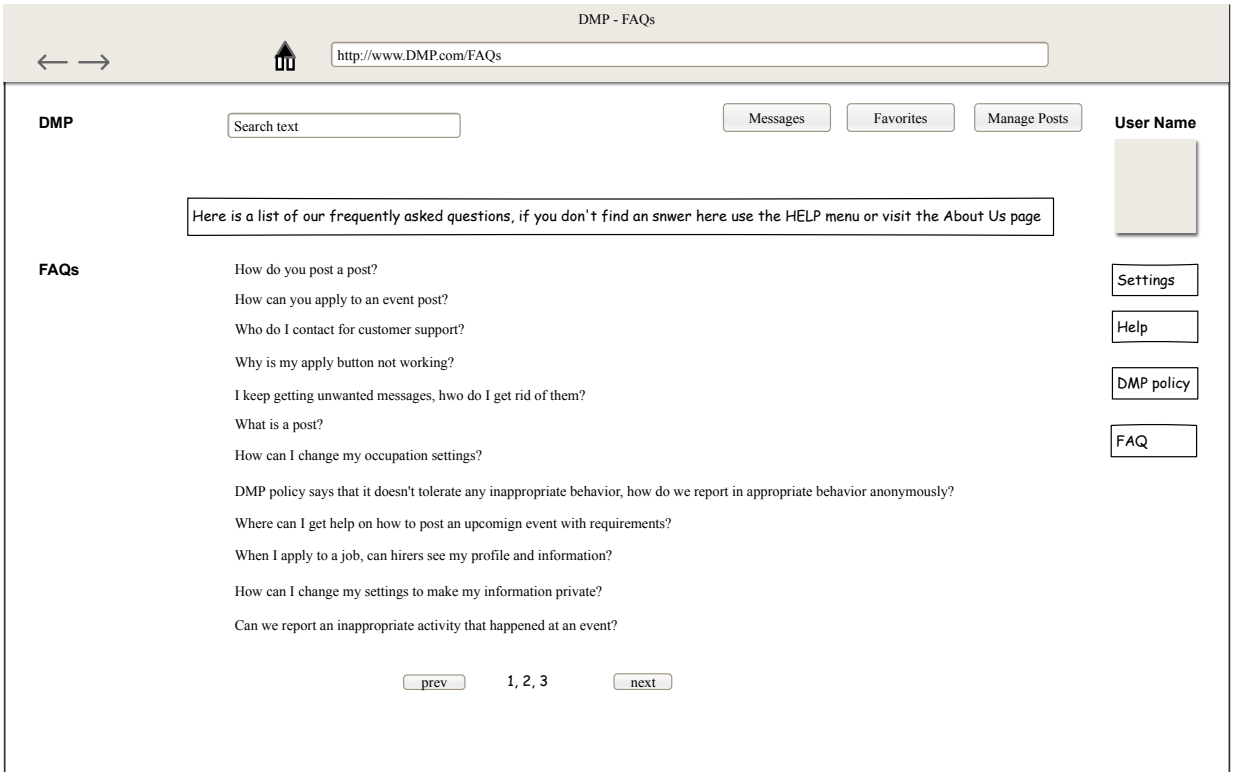
About US



DMP – FAQs page UI

The following is the DMP's FAQ's page UI. Users can look at already answered questions using the FAQ page. It should be the same for regular users and posters.

FAQ Page

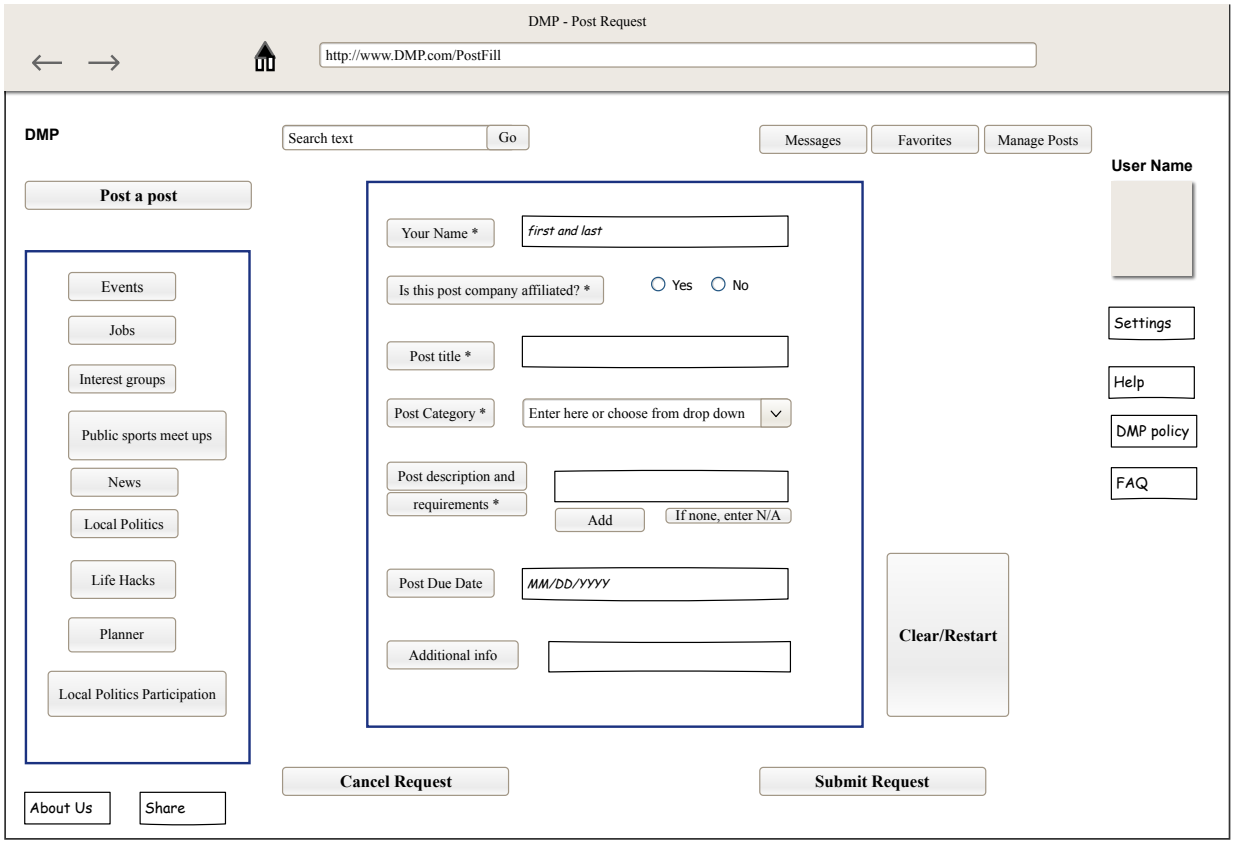


DMP – Post Request UI

The following is the UI for DMP's post request page. It is the same for regular users and posters. The difference is what they put into the boxes. Based on their name, which is linked to their userId (see UML page - 17) and the type of content they want to post or the section they want to post to, the system will determine if the post is valid. The UML diagram shows these methods, userId, postIsValid, etc.

References - See UML diagrams – page 16

Posting

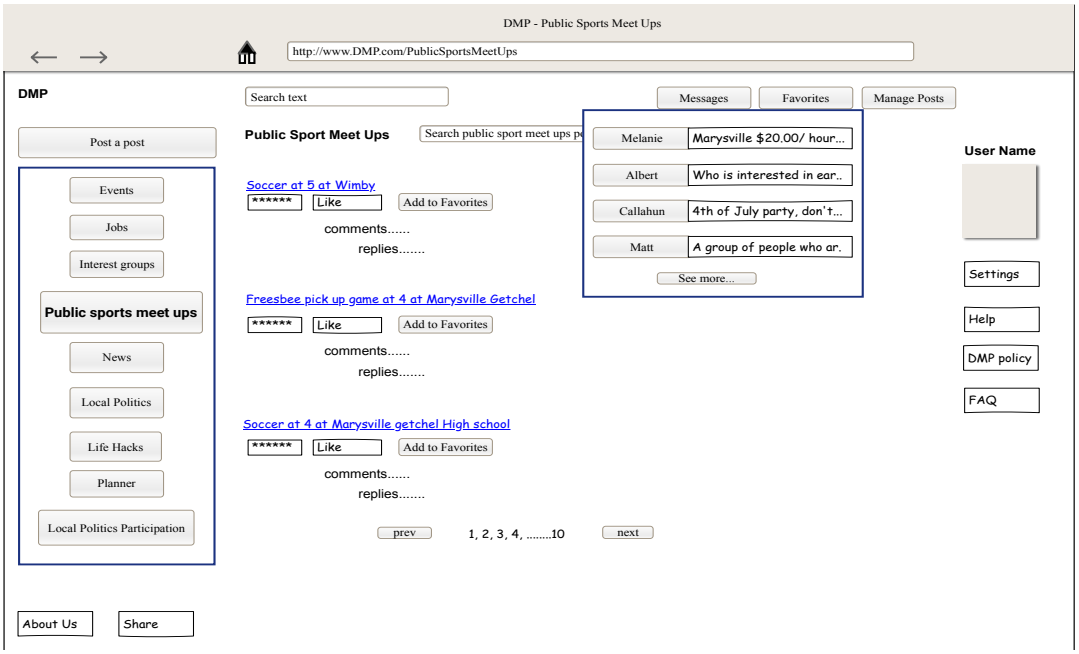


DMP small Messages list box - UI

The following is the UI for the messages list that is shown when the messages button is clicked.

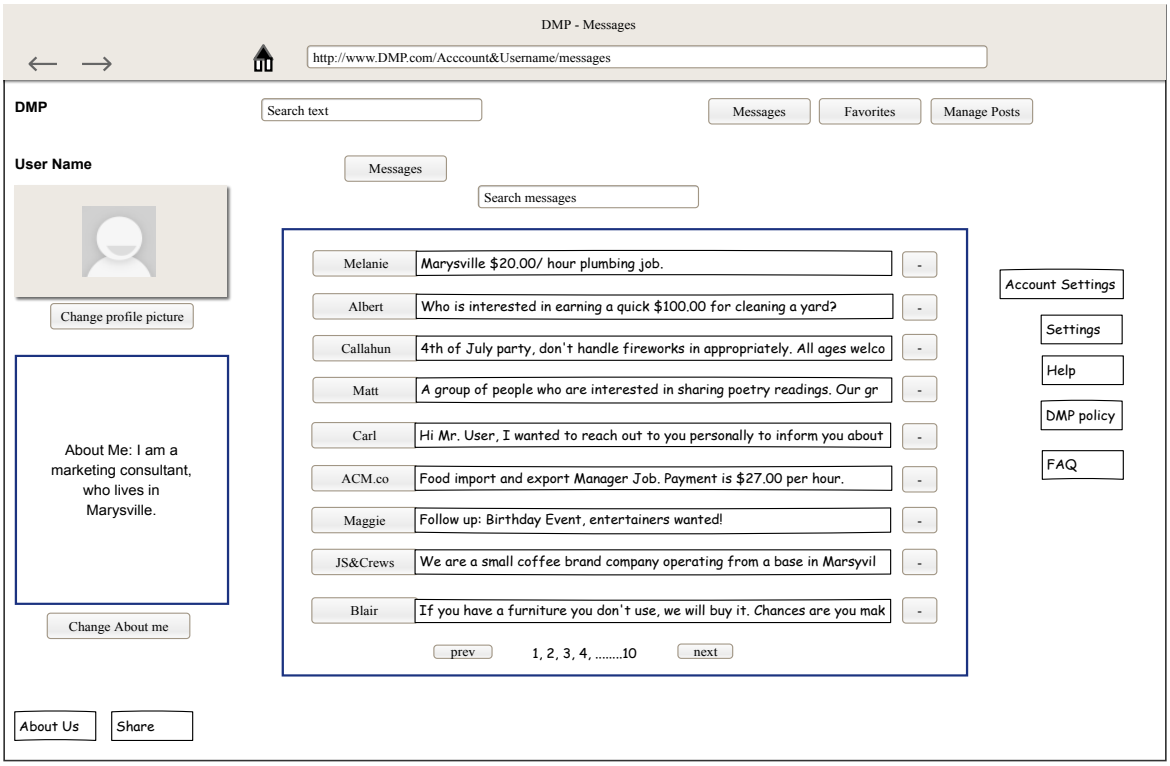
The button is located on top right corner of most of the DMP's pages. The favorites and manage posts buttons follow the same format. So, they are not included in this section.

Messages



The following is the UI for all messages page. Users can click see more from the messages drop down list or go to messages section from their profile. It will be the same for both a poster and a regular user.

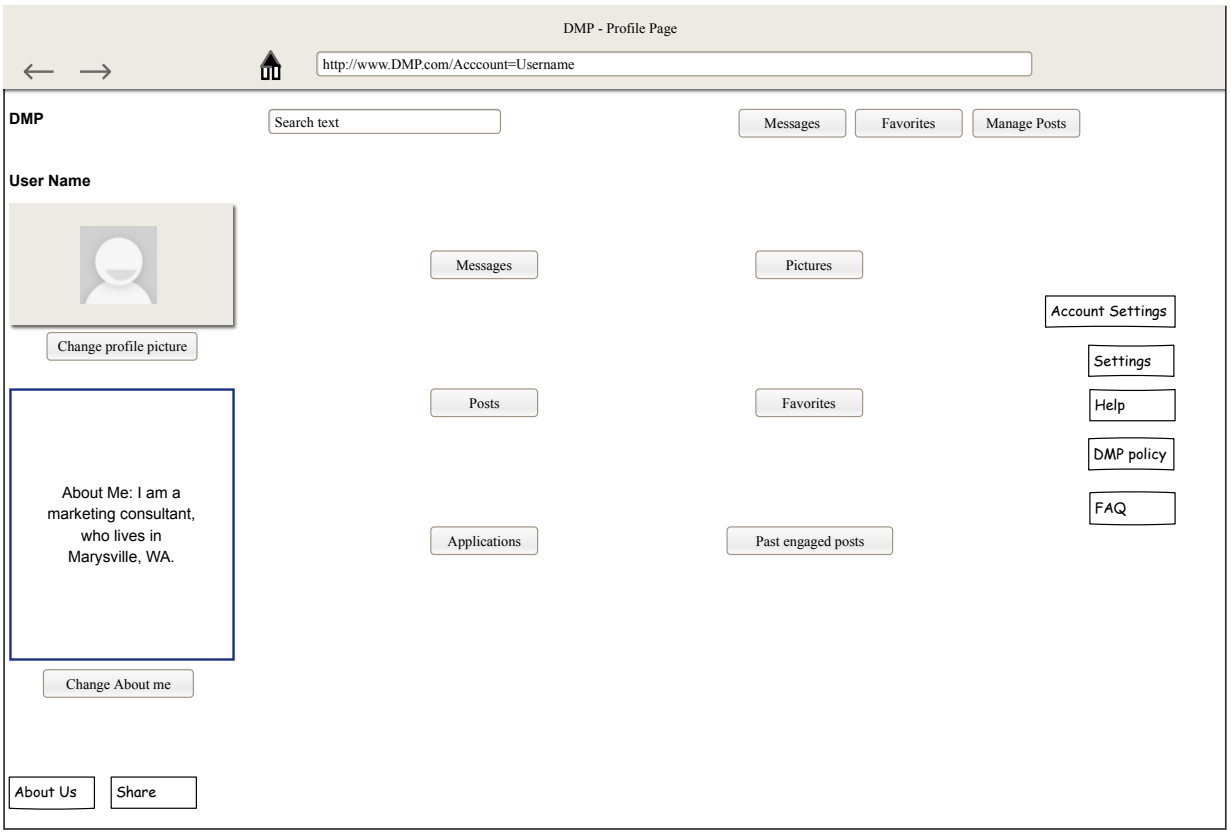
Messages (from see more or account page)



DMP's user profile UI

The following is a regular user's profile page UI. A posters profile would also look the same.

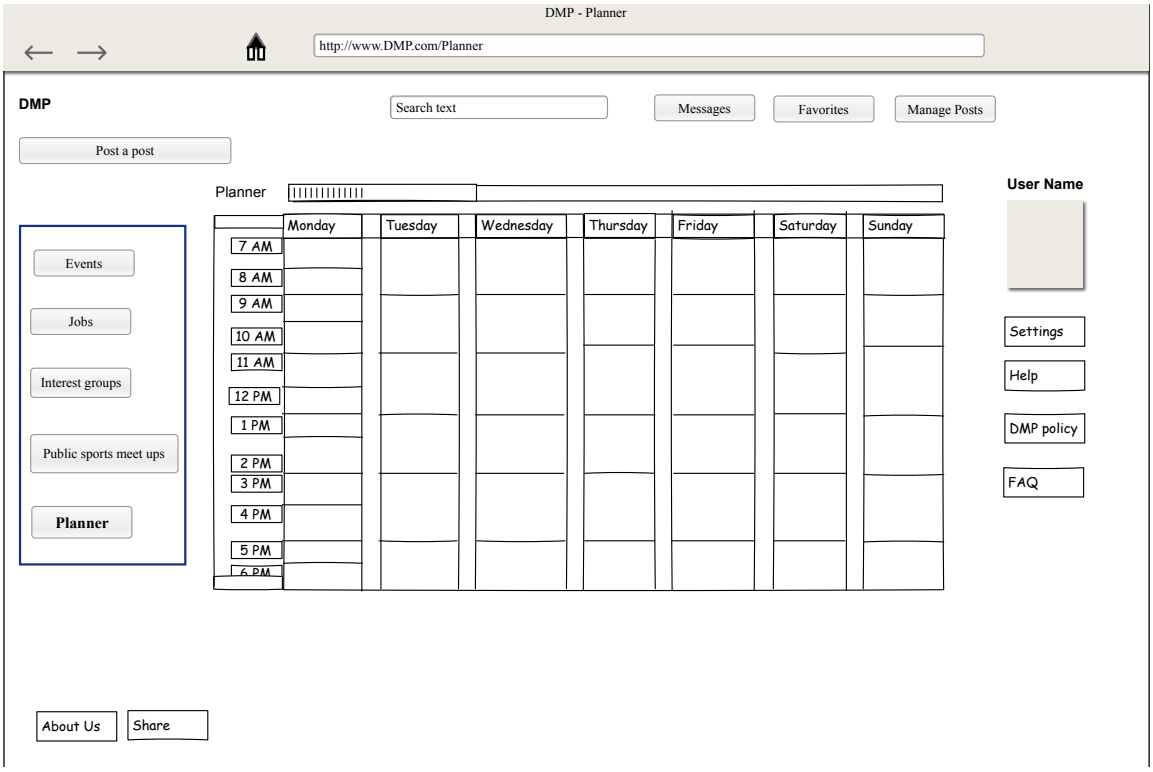
Profile



DMP posters' main page UI

It is a little different from a user's page because posters don't need to post on pages like news, politics, or life-hacks. A user with a poster account is expected to be a self-employed or organization linked event coordinator or talent seeker. Thus, for posting on other sections, a regular user account is required.

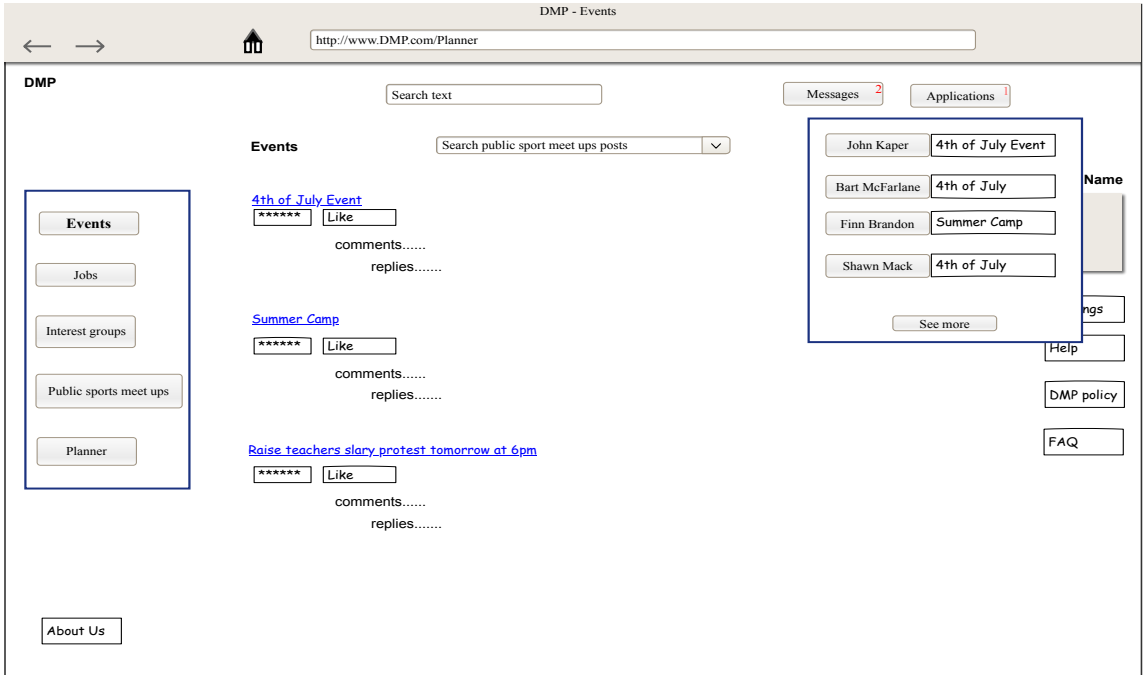
Posters Home Page



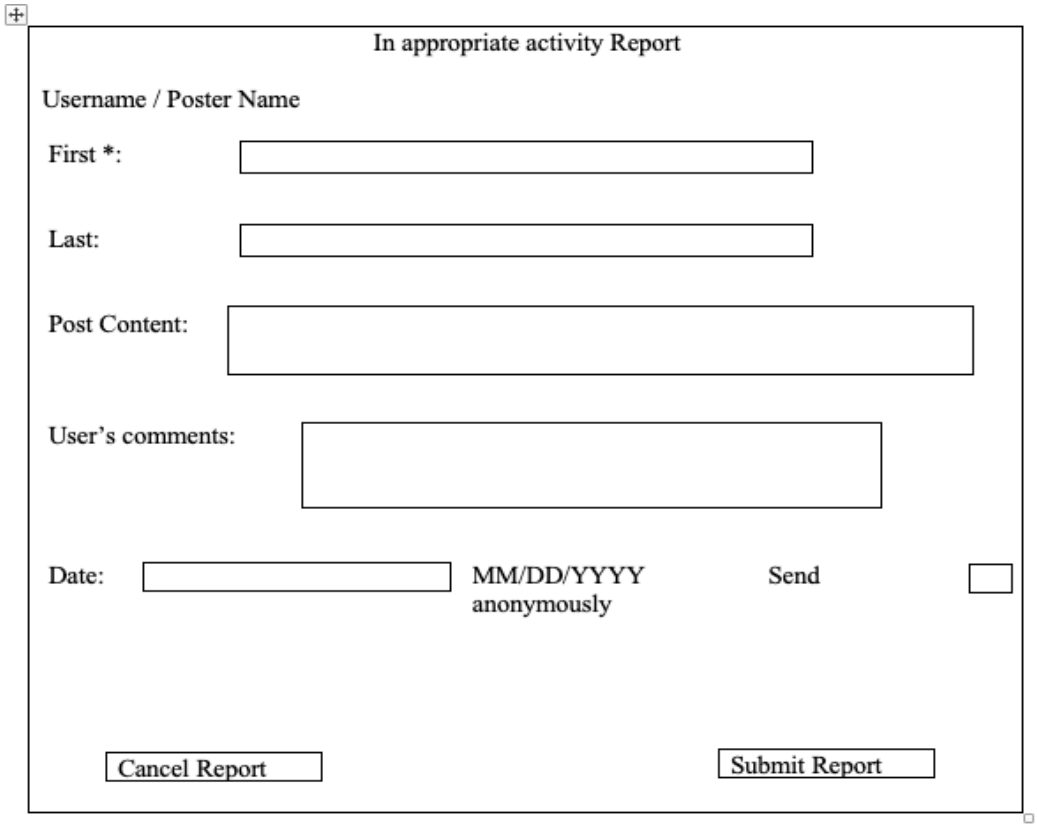
DMP – Application list – drop down UI for posters

The following is the DMP posters page application list UI. It follows the same format as a regular user's messages list or favorites list.

Posters Random page - Applications list Prototype



6.0 Appendices



6.1 Glossary

DMP – Do more and proper
FAQ – Frequently asked questions
UI – User interface n
UML – Unified modeling language
SAST – Static application security testing
DAST – Dynamic application security testing
IAST – Interactive application security testing
RASP – Runtime application protection

6.2 References/ Bibliography

Brogaard, Berit. “Why Are We Busier than Ever despite Having Nowhere to Go?” *Psychology Today*, Sussex Publishers, https://www.psychologytoday.com/us/blog/the-mysteries-love/202004/why-are-we-busier-ever-despite-having-nowhere-go

Kates, Jennifer Follow @jenkatesdc on Twitter, et al. “Economic Impact of Covid-19 on PEPFAR Countries.” *KFF*, 4 Feb. 2022, https://www.kff.org/global-health-policy/issue-brief/economic-impact-of-covid-19-on-pepfar-countries/#:~:text=The%20toll%20the%20COVID%2D19,downturn%20since%20the%20Great%20Depression

Larman, Craig. *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*, 3rd ed., Prentice-Hall, Upper Saddle River (New Jersey), 2005, pp. 89–91.

Richter, Felix. "Majority of Teens Admit to Excessive Cellphone Usage." *Statista*, Statista Inc., 23 Aug 2018, https://www-statista-com.ezproxy.spu.edu/chart/15197/teenagers-views-on-technology-use/