

Gordon Petry

VM Name: csc415-server13.hpc.tcnj.edu

Directory: ~/vm-csc415/assignments/BeaconResources

Assignment 3

Analysis and Design Document

Use Case Descriptions

Use case: Select a filter to view relevant resources

Primary Actor: User

Goal in context: Allow for users to select the filter that applies to the type of research they are specifically looking for.

Preconditions: Data has been pulled from the database and organized into their assorted categories.

Trigger: User selects the category of their choice from the filters.

Scenario:

1. User navigates to the website
2. User hits "Runaway Youth Shelter" from filters
3. Page renders options under this category under each county

Exceptions:

1. None

Priority: Primary essential, must be implemented

When available: First increment

Frequency of use: Many times a day

Channel to actor: Via constructed web site

Secondary actors: None

Channels to secondary actors: None

Open issues:

1. In what order should resources be displayed?

Use case: Admin sign in

Primary Actor: Admin

Goal in context: Allow admins to sign in

Preconditions: Admin account is created

Trigger: Admin selects "Sign In"

Scenario:

1. Admin navigates to the website

2. Admin hits "Sign In" on the page
3. Admin enters username and password
4. Admin hits "Sign In"

Exceptions:

1. Database has not been created
2. Password Incorrect: Re Enter password

Priority: Secondary essential, must be implemented

When available: First increment

Frequency of use: Single time per navigation to site

Channel to actor: Via constructed web site

Secondary actors: None

Channels to secondary actors: None

Open issues:

1. How are accounts created? I'm thinking the site owner must automatically add them.

Use case: Add a resource

Primary Actor: Admin

Goal in context: Allow admins to add a resource

Preconditions: Database is setup and allows for admin input

Trigger: Admin selects "Add Resource"

Scenario:

1. Admin navigates to the website
2. Admin hits "Add Resource" on the page
3. Page renders to allow for admin to input data about the resource
4. Admin hits "Submit"

Exceptions:

1. Database has not been created
2. Admin is not signed in

Priority: Secondary essential, must be implemented

When available: After admin is signed in

Frequency of use: As many times as needed

Channel to actor: Via constructed web site

Secondary actors: None

Channels to secondary actors: None

Open issues:

1. How should this be accessed, through a sidebar similar to adding a resource or through the resource itself?

Use case: Remove a resource

Primary Actor: Admin

Goal in context: Allow admins to remove a resource

Preconditions: Database is setup and allows for admin input

Trigger: Admin selects "Remove Resource"

Scenario:

1. Admin navigates to the website
2. Admin hits "Remove Resource" on the page
3. Page renders to allow for admin to input data about the resource
4. Admin selects "Remove" on the resource they want to remove

Exceptions:

1. Database has not been created
2. Admin is not signed in

Priority: Secondary essential, must be implemented

When available: After admin is signed in

Frequency of use: As many times as needed

Channel to actor: Via constructed web site

Secondary actors: None

Channels to secondary actors: None

Open issues:

1. How should this be accessed, through a sidebar similar to adding a resource or through the resource itself?

Use case: Add a suggestion of a resource to site

Primary Actor: User

Goal in context: Allow for users to create a suggestion for administrators to add additional resources to the site.

Preconditions: Database has been setup to allow user suggestions

Trigger: User selects "Suggest Resource" on the web page

Scenario:

1. User navigates to the website
2. User hits "Suggest Resource" on the page
3. Page renders to allow for user to input the name and location of a resource for administrators to look into further

Exceptions:

1. Database has not been created

Priority: Tertiary essential, must be completed

When available: First increment

Frequency of use: As many times as needed

Channel to actor: Via constructed web site

Secondary actors: Administrators

Channels to secondary actors: Through database suggestions

Open issues:

1. How should this be accessed, through a sidebar?

Use case: Suggest removal of a resource to site

Primary Actor: User

Goal in context: Allow for users to add a suggestion for administrators to remove a resource from the site.

Preconditions: Database has been setup to allow user suggestions

Trigger: User selects "Suggest Resource" on the web page

Scenario:

1. User navigates to the website
2. User hits "Suggest Resource" on the page
3. Page renders to allow for user to input the name of the resource as specified on the site and a description of why it should be removed

Exceptions:

1. Database has not been created

Priority: Tertiary essential, must be completed

When available: First increment

Frequency of use: As many times as needed

Channel to actor: Via constructed web site

Secondary actors: Administrators

Channels to secondary actors: Through database suggestions

Open issues:

1. How should this be accessed, through a sidebar similar to adding a resource or through the resource itself?

Use case: View Suggestions

Primary Actor: Admin

Goal in context: Allow admins to view add or removal suggestions

Preconditions: Database has been setup to allow user suggestions, admin is signed in

Trigger: Admin selects "View Suggestions" on the web page

Scenario:

1. Admin navigates to the website
2. Admin hits "View Suggestions" on the page

Exceptions:

1. Database has not been created

Priority: Tertiary essential, must be completed

When available: When admin is signed in

Frequency of use: As many times as needed

Channel to actor: Via constructed web site

Secondary actors: None

Channels to secondary actors: None

Open issues:

1. None

Use case: Enter location and view closest options nearby

Primary Actor: User

Goal in context: Allow users to view resources nearby them

Preconditions: User has entered a valid location

Trigger: User selects "Enter location" on the web page

Scenario:

1. User navigates to the website
2. User hits "Enter location" on the page
3. User enters a location
4. Page renders and displays options closest

Exceptions:

1. Location is invalid
2. Algorithm is not implemented

Priority: Primary extra, will be completed if there is time

When available: First increment

Frequency of use: As many times as needed

Channel to actor: Via constructed web site

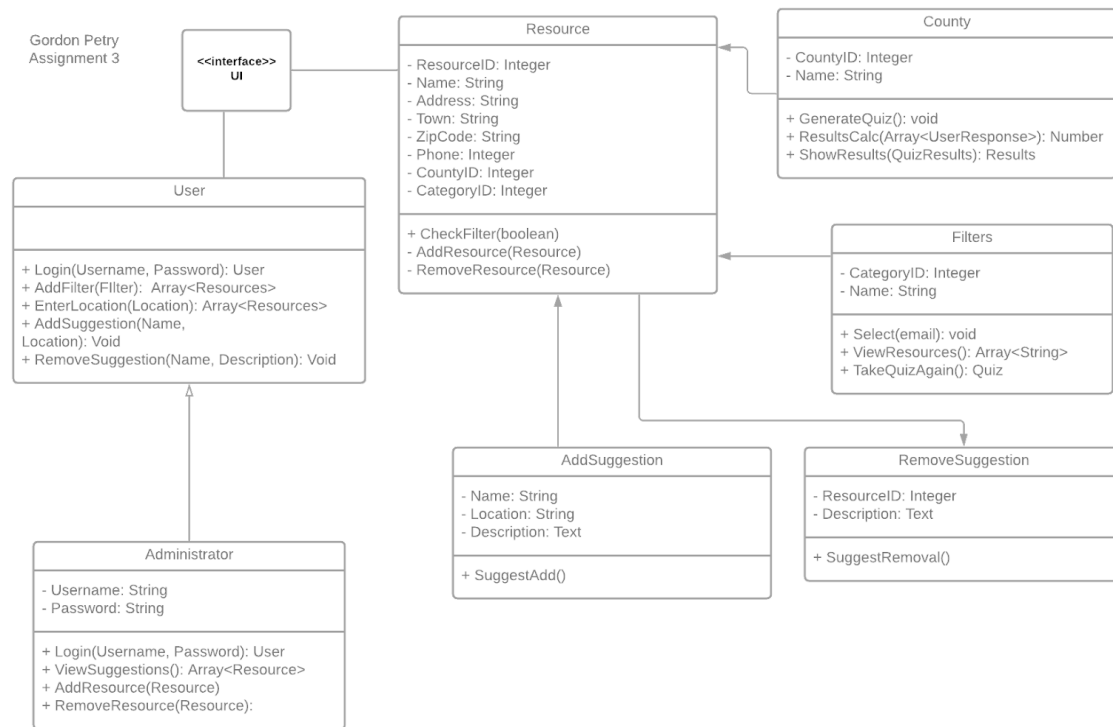
Secondary actors: None

Channels to secondary actors: None

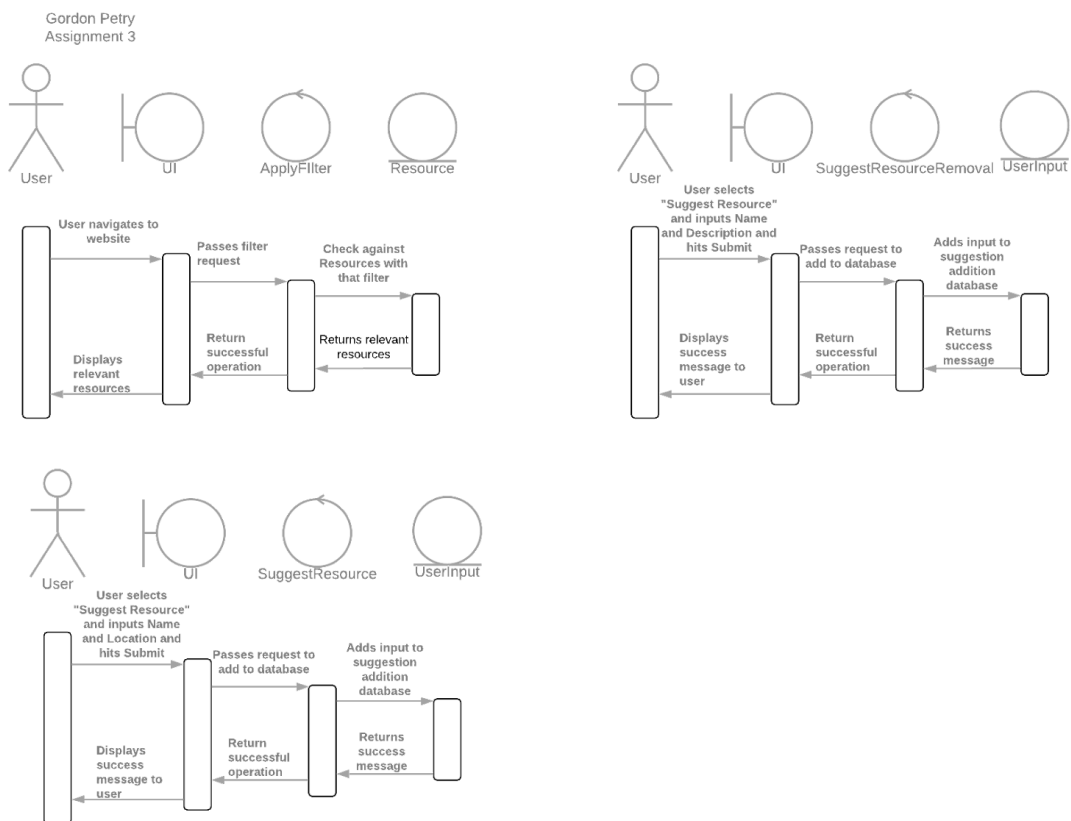
Open issues:

1. How could this algorithm be implemented? Should it be based on zip code to simplify searching?

Detailed Design Class Diagram

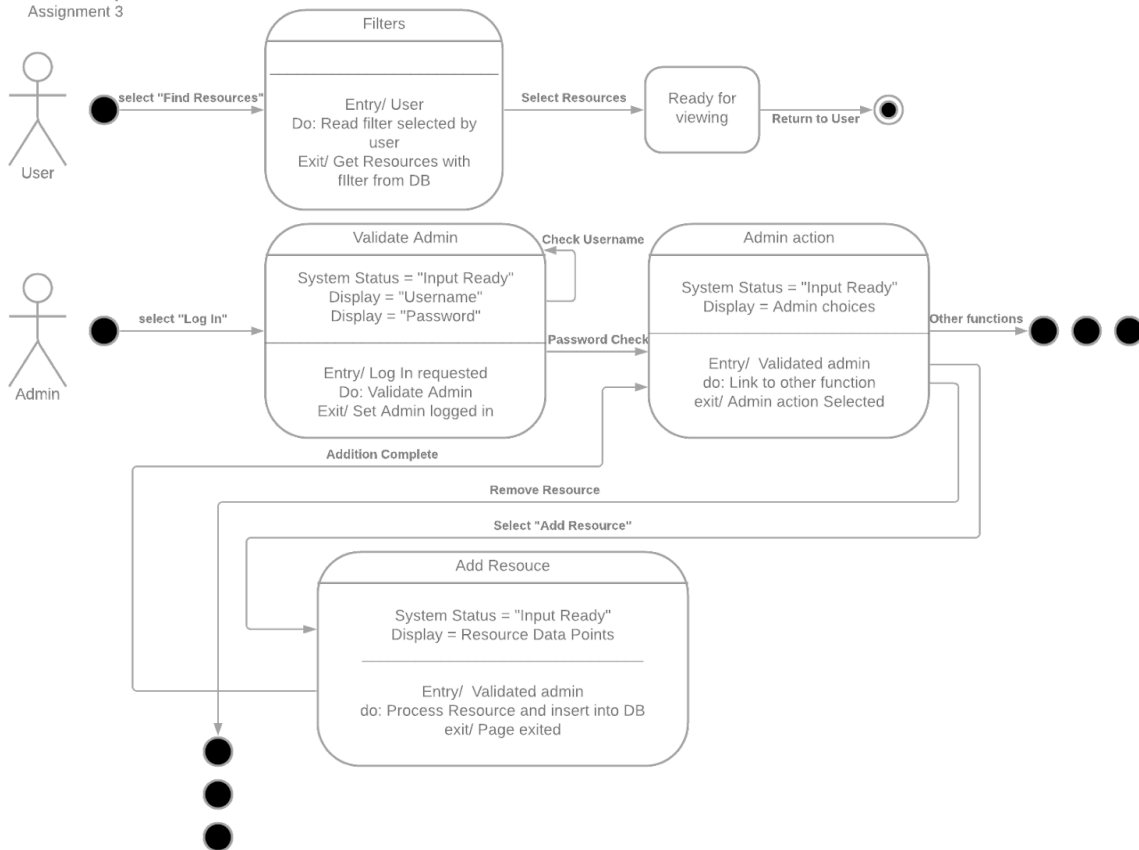


System Sequence Diagram



Statechart

Gordon Petry
Assignment 3



UI Mockups

Gordon Petry
Assignment 3

The image displays four UI mockups for the BeaconResources application, arranged in a 2x2 grid. Each mockup represents a different user flow or state of the application.

Top Left Mockup: Shows the main interface for finding resources. It includes a top bar with navigation icons and buttons for "Suggest Resource" and "Sign In". Below the top bar, there's a section for "Mercer County" and "Ocean County", each containing a table of resources with columns for Name, Address, Town, State, Zip, and Phone. A "Find Resources" button is visible.

Top Right Mockup: Shows the "Suggested Additions" and "Suggested Removals" section. It includes a "View Suggestions" button and a "Hello Admin!" greeting. The "Suggested Additions" section has a table with "Name" and "Location" columns, and an "Add Resource" button. The "Suggested Removals" section has a similar table and a "Remove Resource" button.

Bottom Left Mockup: Shows the "Admin Sign In" section. It includes a "Sign In" button and a message: "This page is for site admins to add and remove relevant resources! You can use the site normally as a regular user without signing in." The sign-in form has fields for "Username" and "Password".

Bottom Right Mockup: Shows the "Create Resource Suggestion" section. It includes a "Suggest Resource" button and a "Sign In" button. The form has two columns for "Name of Resource" and "Location of Resource", each with a "Name" field and a "Reason for Removal" field. There are "Create Addition Suggestion" buttons at the bottom of each column.

This application will provide visibility to the user by showing the user exactly what's happening by selecting specific options on the page. By selecting an option on the dropdown and hitting "Find Resources", the user will be led to the next page displaying this information. This also goes for affordance, as the user should be able to look at the buttons and options available to them on the page and intuitively be able to work with them off the bat. The mockups I have created demonstrate the 8 golden rules for designing interactive interfaces in the following ways:

1. The UI is consistent in that all of the information is in the same location each time the page is loaded. The top always contains the same top bar, the main page always contains either information input or some database tables for viewing.
2. There isn't really a need for shortcuts within the application seeing that there are few pages to navigate through, but all the links accessible are always contained within the current page based on the user, so this should be fine.

3. The current state of the site is relatively self explanatory, but there is still a help icon for those with questions. On sites irrelevant to specific users there are text boxes to describe what is required of the user.
4. Upon signing in or adding a suggestion, there will be a popup to indicate that the action went through and that information was received.
5. When adding or removing a resource or suggestion, the user will be prompted with a box to check that they are finished inputting information to prevent any misclicks or errors.
6. Simply unselecting the filter for the user will undo any selections they have made since only one can be selected from the dropdown list at a time.
7. The site, from my point of view, is relatively easy to understand and intuitive. Everything the user needs to access is just a click away and if they need more information, they can just click the help icon.
8. Since everything the user needs to do can be done at a glance, there is really no load for the short term memory for the user to handle, as the options available to them will be right at the drop down.

Modularity, efficiency and database appropriateness of the application.

- Modularity plays a big role in the simplicity of the project! By making the application modular, it will make it much easier to read and understand for those who take on the project in the future. With a modular approach, we can also ensure that different data points are changed when we want them to be. By using encapsulation through ruby's accessor methods, we can isolate the data so that only certain parts of classes can be changed. By making things private in this way, we make it so that the code cannot be manipulated by those who want to do harm to the application. Through this application, I aim to do just that. By allowing only admins to change anything about the resources themselves, we can make it so that regular users cannot access different parts of the program without a login.
- Through this application, my biggest focus was simplicity for the users. By working towards this, we can achieve elegance and efficiency in our algorithms in that we simply only need them to be able to view and suggest information. This is done as simply as possible by allowing them just to input some things such as a name of a location they want to suggest or a name and the description of why it should be removed. This is then given to the admins to look into further so we don't have people putting in incorrect suggestions because of a bad experience there and then removing the resource themselves. By allowing users to only work with a few things on their end we are keeping it straight forward and simple, and thus there is the elegance. On the end of the administrators, I hope to allow them to access exactly what they want and then leave without any issues. As shown in the state diagram, we can see that when an admin signs in, they have their choice of which action they want to perform and then allow them to

leave the site afterwards. As I mentioned, simplicity in the algorithms will keep it both elegant and efficient.

- By using databases for each of the different points, addition suggestions, removal suggestions, and resources themselves, I will be able to appropriately restrict access to specific users. For example, only Admins are allowed to change anything about the resource database, but both suggestion databases can be altered in any way that is necessary. By using the same form of database throughout, the application maintains its simplicity by allowing us to restrict information and access it all at a glance very easily, as is the intention with the project.