AGILE HOOF

Elaboration Spec

Caitlin Sullivan, James Jordan, Emily Green,
Caleb DeSpain, Logan Robinson, Turner Barnett

Table of Contents

System Requirements

System Requirements

We have compiled a list of system requirements, a statement that defines what a specific system component must accomplish. We have listed functional requirements, which define a process that a system must perform, and nonfunctional requirements, which define characteristics that a system must have.

Functional Requirements:

Payment Systems

- The payment system will be able to accept cash
- The payment system will be able to accept checks
- The payment system will be able to accept online payments
- They payment system will allow members and non-members the ability to donate online

Database

- The system will collect camper information
- The system will collect volunteer information
- The system will collect donor information
- The system will collect information about items for the silent auction
- The system will collect information about horseback riding gear on hand
- The system will collect information about supplies on hand
- The system will allow users to update, delete, and modify records in the database

Website

- The content management system will have an events page
- The content management system will have a fundraising page
- The content management system will have an about us page
- The content management system will enable users to sign up for a newsletter
- The content management system will allow users to sign up for volunteering events
- The content management system will allow users to apply for board member positions
- The content management system will allow users to donate items
- The content management system will allow users to send out newsletters
- The content management system will allow users to update, delete and modify various web pages and features on the web pages

Non-functional:

Security

- The payment system will offer fraud protection
- The payment system will offer encryption

- The payment system will meet payment card industry standards council standards
- The camper database will comply with confidentiality rules and have security to protect sensitive information
- The donor database will provide security to protect donor data

Capacity

- The database will be able to store at least 1,000 donor records
- The database will be able to store at least 1,000 camper records
- The database will be able to store at least 1,000 volunteer records
- The database will be able to store at least 1,000 silent auction item records
- The payment system can accept donation up to $10,000

Stress requirements

- Up to 50 users will be able to access the website concurrently
- All board members will be able to access the database concurrently

Throughput Requirements

- “Modify board member data” may apply up to 2,000 updates per work day
- “Modify volunteer data” may apply up to 2,000 updates per work day
- “Modify camper data” may apply up to 2,000 updates per work day
- “Modify auction data” may apply up to 2,000 updates per work day

Reliability:

- The website will be available to users 97% of the time
- the database will be available to board members 99% of the time

Back-up Requirements:

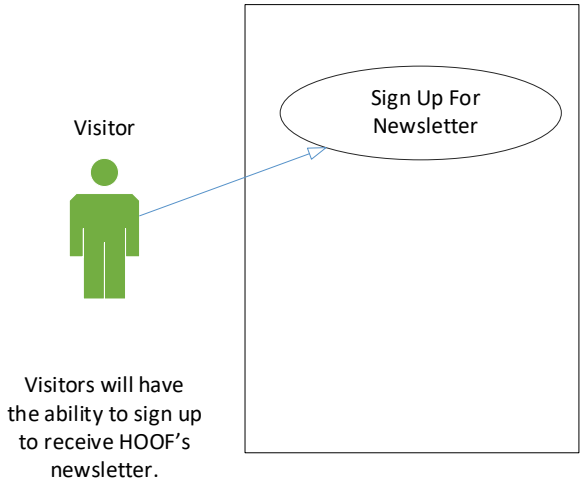
- Non-critical data will be backed-up weekly
- Critical data will be backed-up daily

Use Case Diagrams

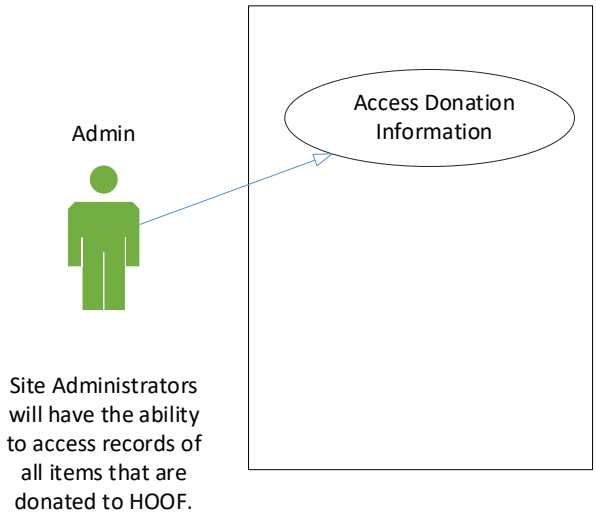
Use Case Diagrams

A use case diagram displays how the users interact with the system to complete a task. The actors are connected using a line to the use cases they instantiate.

UC1: SignUpForNewsletter

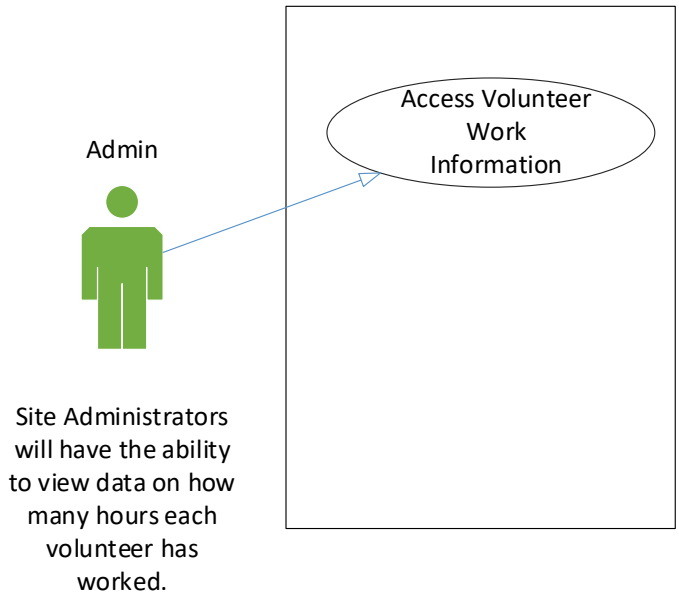


UC2: AccessDonationInformation

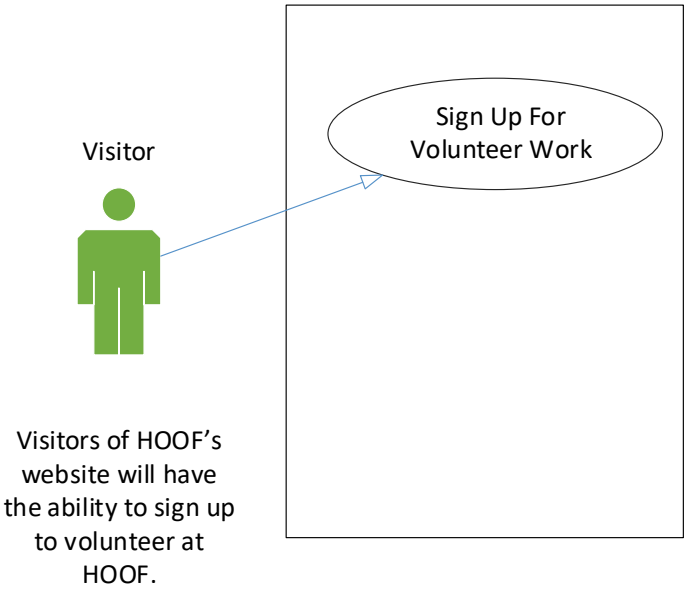


UC3: SignUpToDonateGoods

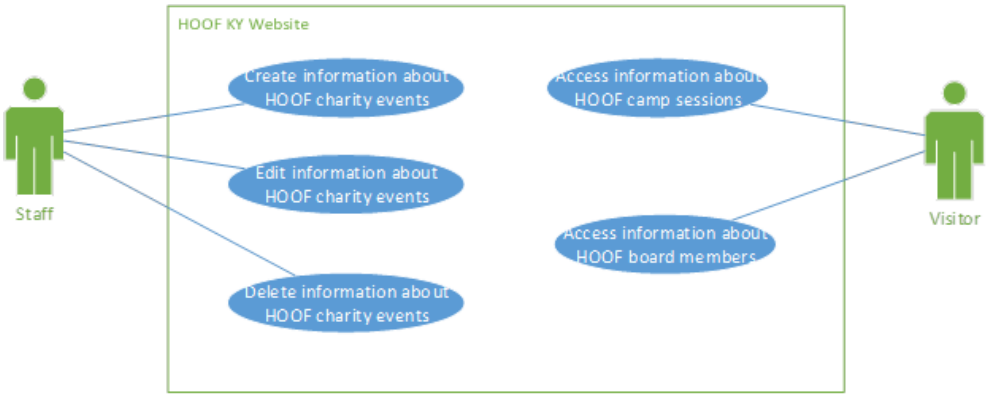
UC4: AccessVolunteerWorkInformation



UC5: SignUpForVolunteerWork

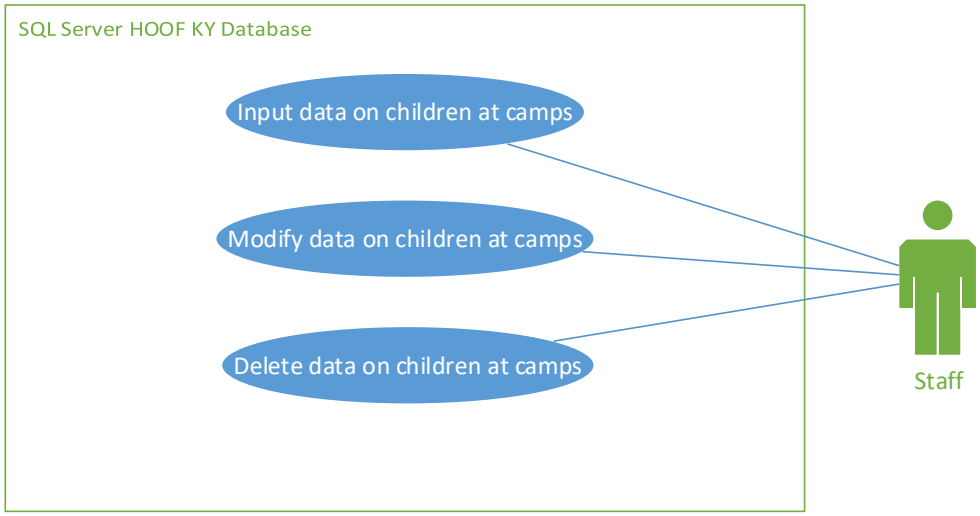


UC 6-7 , 9-11 :



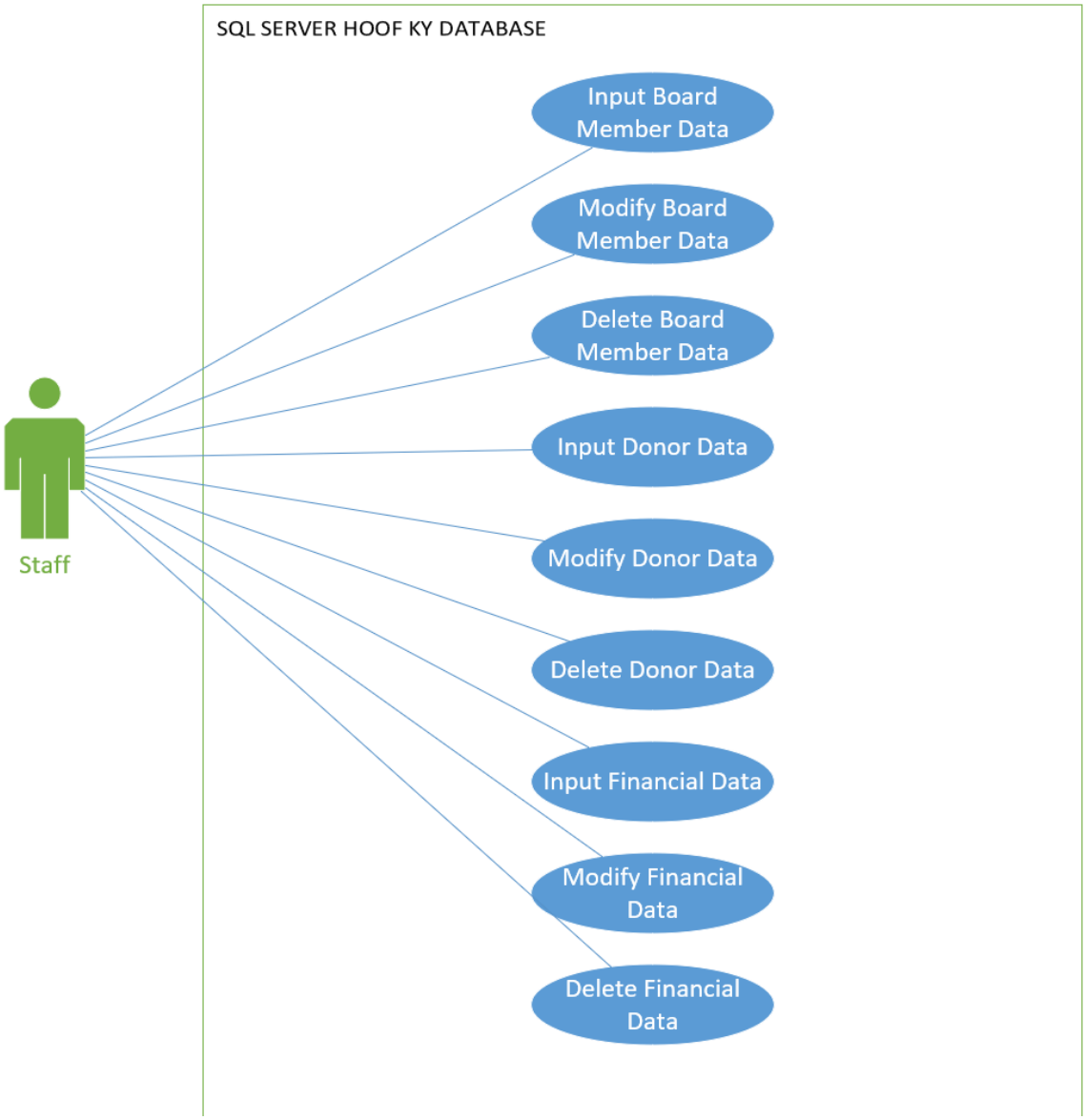
The above Use Case Diagram includes two actors, staff and visitor, which both interact with the HOOF KY Website system to complete the above use cases. The visitor within this diagram interacts with the HOOF KY website to access information about HOOF camp sessions and HOOF board members. The staff interacts with the HOOF KY website and WordPress to create, edit, and delete information about charity events under the charity events tab on the HOOF KY website.

UC 12-14



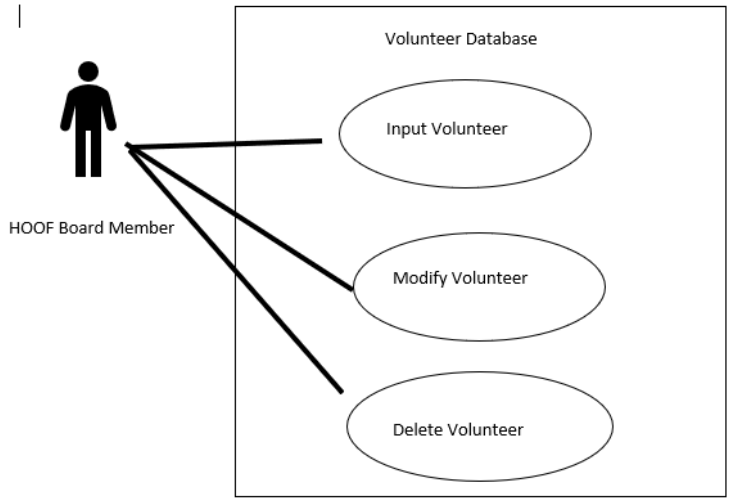
The Use Case Diagram above includes one actor, which is staff of HOOF KY. The subsystem, or the rectangular border around the use cases, is the SQL Server HOOF KY Database that stores all the information collected regarding business. The staff members interact with SQL Server to connect to the HOOF KY Database to input, modify, and delete data that has been collected for the children that attend HOOF camps.

UC 15-23



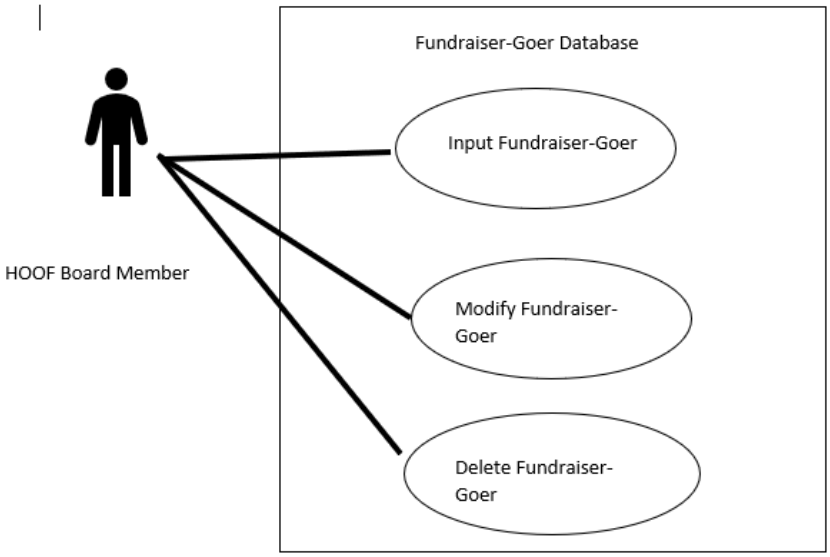
The use case diagram includes one actor, nine use cases and the system boarder surrounding the use cases. The actor represented in the use case diagram is the HOOF Ky staff, as they will be the ones inputting, modifying, and deleting information from their database. The nine use cases represented in the use case diagram represent the entering, modification, and deletion of information that HOOF Ky has collected on their board members, donors, and finances. The system boarder represents HOOF Ky's SQL Server Database, which they use to store all of their information. The actor will use SQL Server to input, modify, and delete all of the relevant business information into the database.

UC 24-26



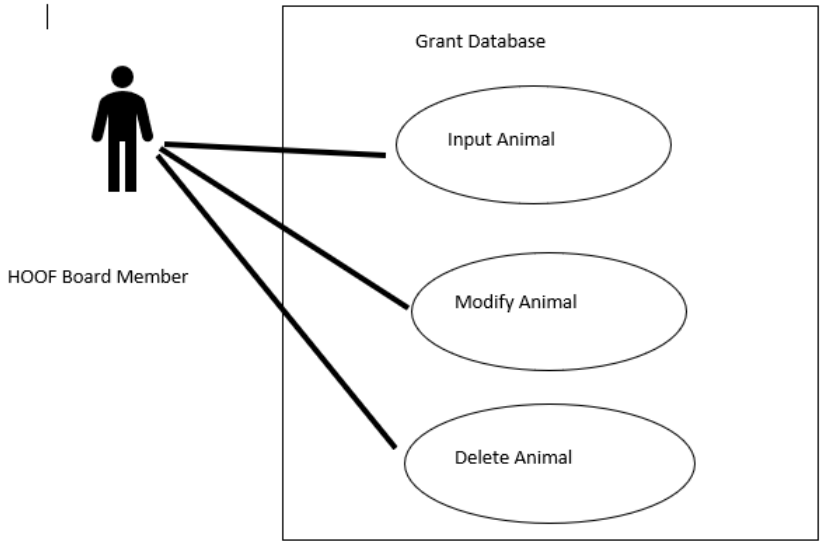
The board member is the sole person responsible for completing the form that ties back to the volunteer database that will input, modify, and delete volunteers from the database depending on what form was filled out and submitted.

UC 27-29



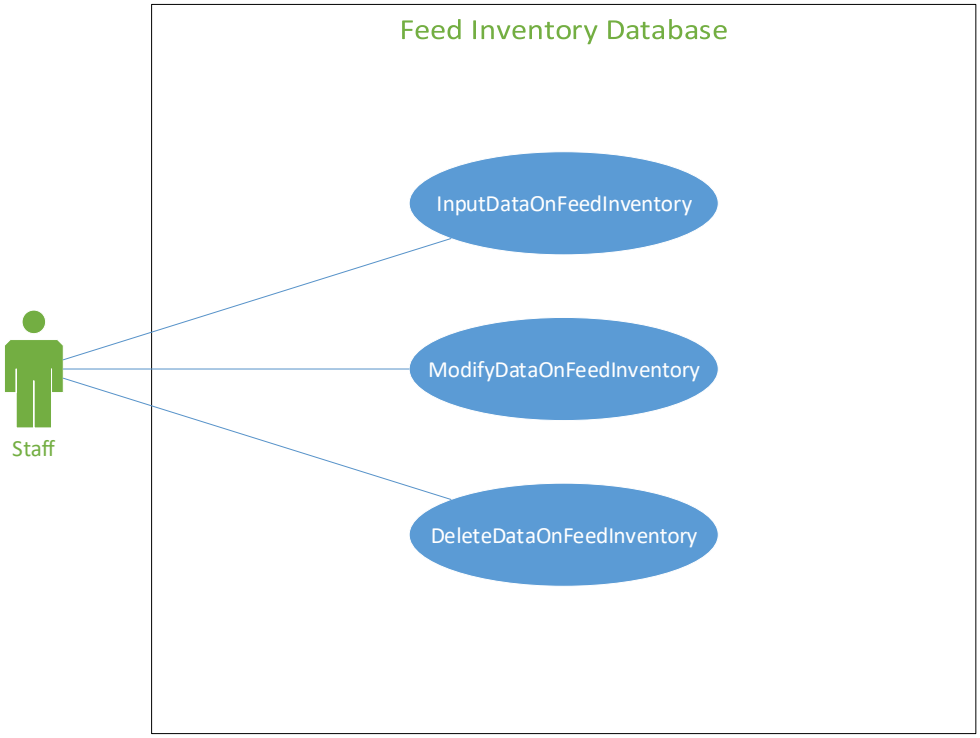
The board member is the sole person responsible for completing the form that ties back to the fundraiser-goer/attendee database that will input, modify, and delete attendees from the database depending on what form was filled out and submitted.

UC 30-32

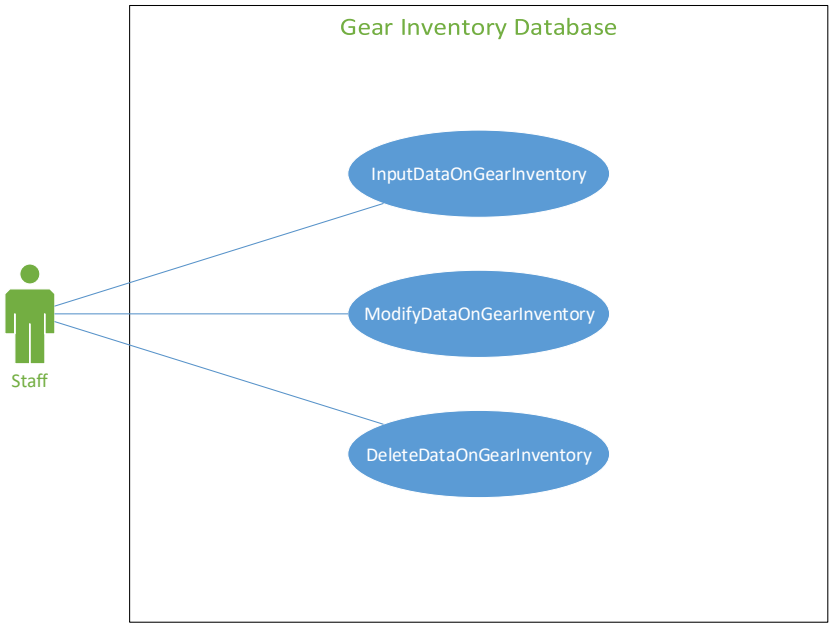


The board member is the sole person responsible for completing the form that ties back to the animal database that will input, modify, and delete animals from the database depending on what form was filled out and submitted.

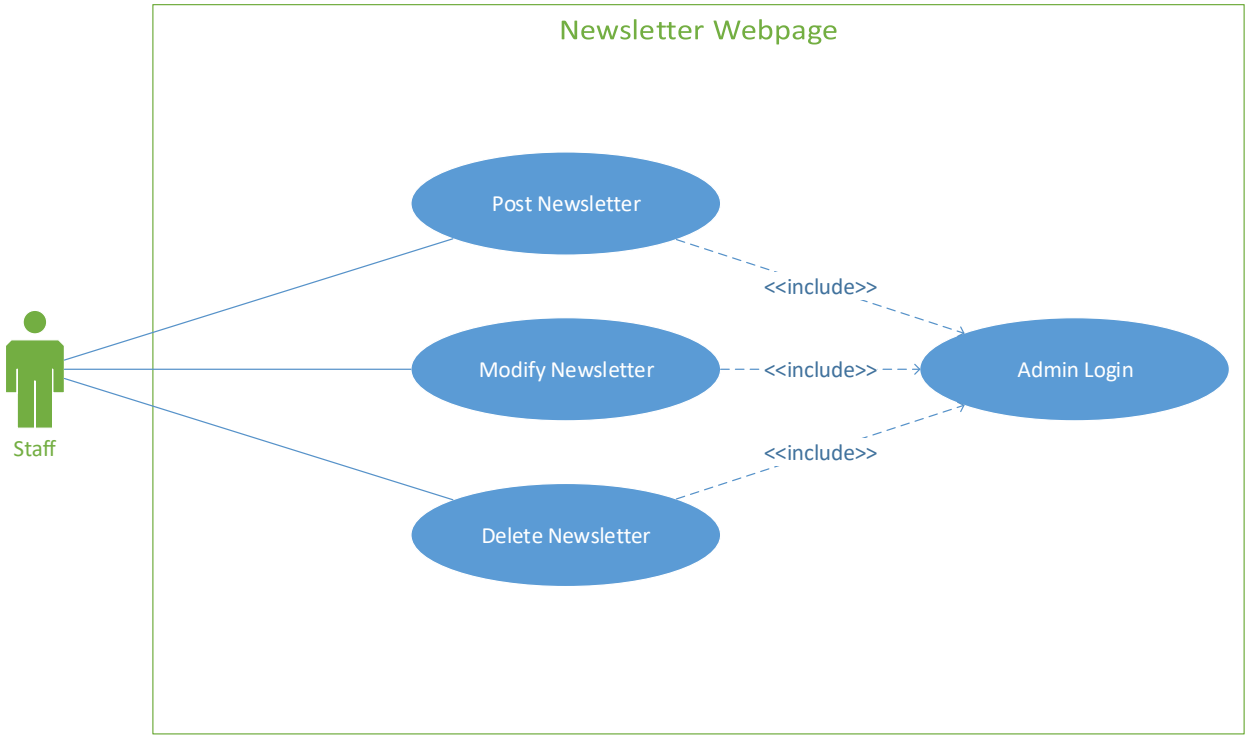
UC 33-35



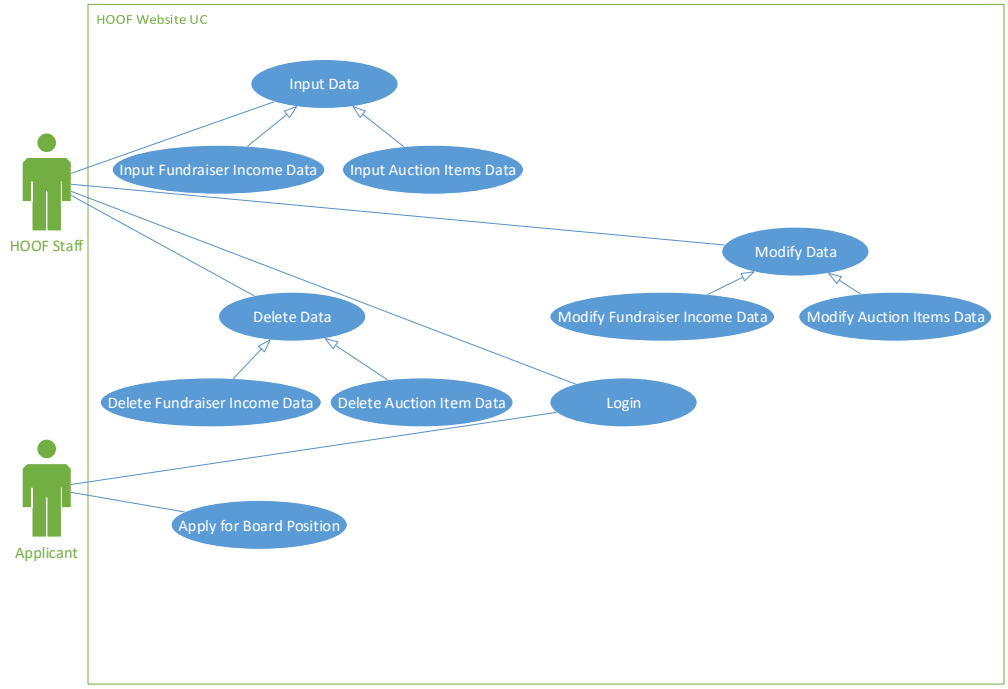
UC 36-38



UC 39 -41

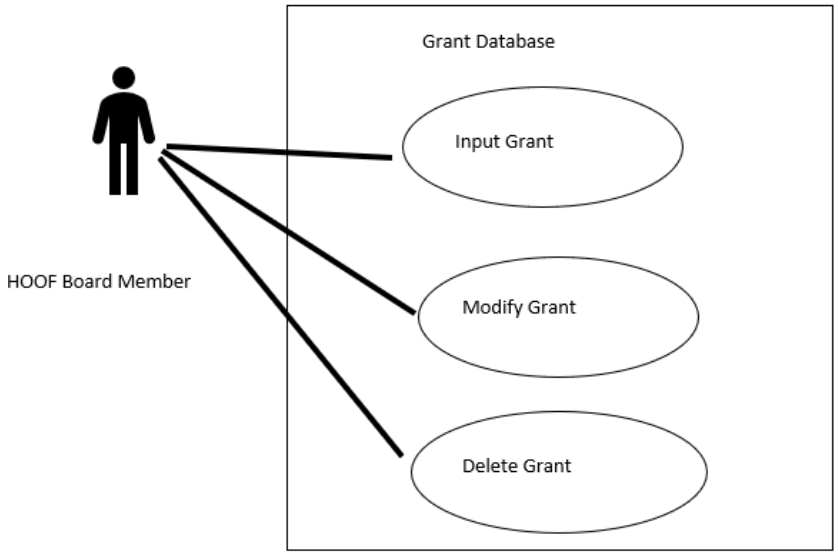


UC 8, 42-47



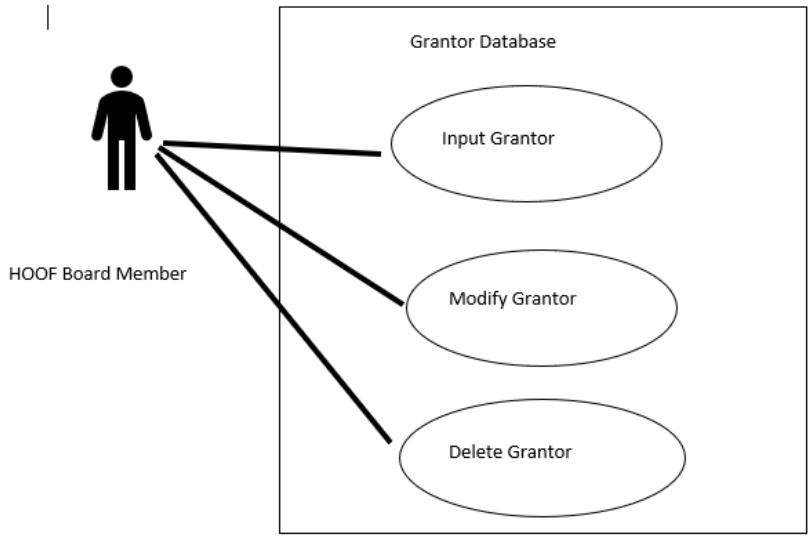
This use case diagram shows possible interactions with the HOOF website, when looking at HOOF staff members and board applicants. The staff member can input, modify, and delete data regarding to the fundraiser income database and the auction item database. The applicant can fill out an application for the board position

UC 48-50



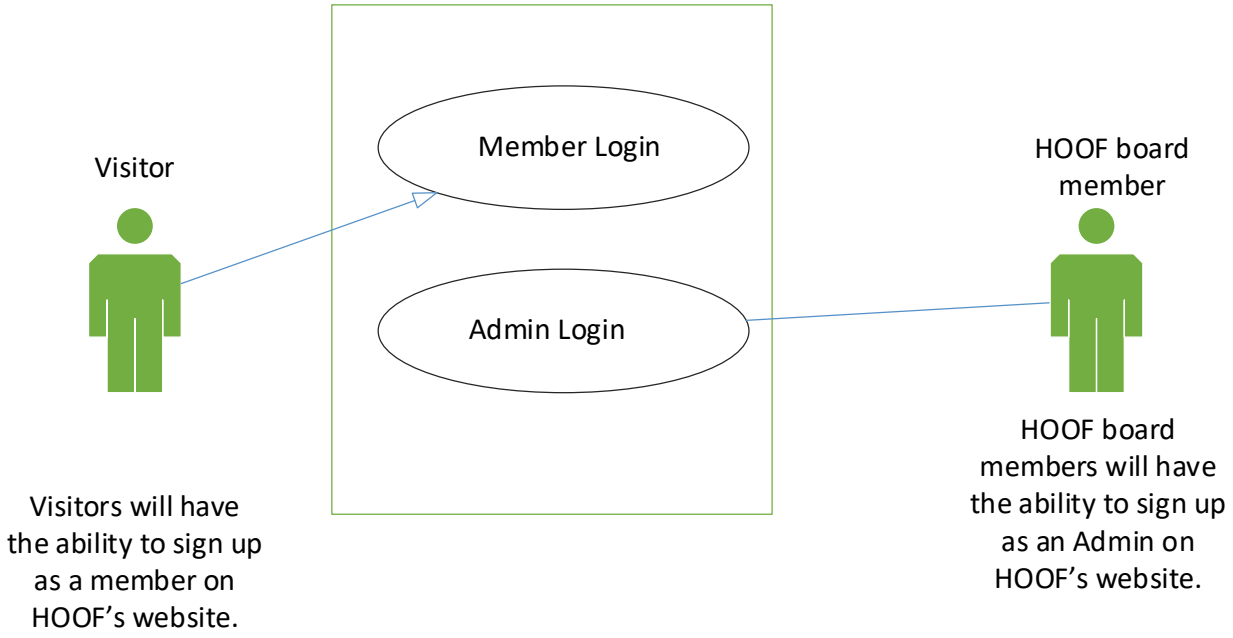
The board member is the sole person responsible for completing the form that ties back to the grant database that will input, modify, and delete grants from the database depending on what form was filled out and submitted.

UC 51-53



The board member is the sole person responsible for completing the form that ties back to the grantor database that will input, modify, and delete grantors from the database depending on what form was filled out and submitted.

UC 54-55


Visitor
Member Login
Admin Login
HOOF board member
Visitors will have the ability to sign up as a member on HOOF's website.
HOOF board members will have the ability to sign up as an Admin on HOOF's website.

Trace Matrix

	U1	U2	U3	U4	U5	U6	U7	U8	U9	U10	U11	U12	U13	U14	U15
Accept Online payments			X												
Anybody can donate		X	X												
Collect Camper Info												X	X	X	
Collect Volunteer Info					X										
Collect Donor Info			X												
Collect fundraiser info															
Events page									X	X	X				
Fundraising page		X													
About Us page				X		X	X								
Newsletter	X														
Volunteer signup				X	X										
Board member application								X							
Collect information about supplies															
Donate Item Capability			X												
Animal Information															
Modify Website									X	X	X				
Modify Database									X	X	X	X	X	X	

	U 16	U 17	U 18	U 19	U 20	U 21	U 22	U 23	U 24	U 25	U 26	U 27	U 28	U 29	U 30
Accept Online payments						X	X	X							
Anybody can donate															
Collect Camper Info															
Collect Volunteer Info									X	X	X				
Collect Donor Info			X	X	X										
Collect fundraiser info												X	X	X	
Events page															
Fundraising page															
About Us page															
Newsletter															
Volunteer signup									X	X	X				
Board member application															
Collect information about supplies															
Donate Item Capability															
Animal Information															X
Modify Website															
Modify Database	**X**	X				X	X	X				X	X	X	

	U 31	U 32	U 33	U 34	U 35	U 36	U 37	U 38	U 39	U 40	U 41	U 42	U 43	U 44	U 45
Accept Online payments															
Anybody can donate															
Collect Camper Info															
Collect Volunteer Info															
Collect Donor Info															
Collect fundraiser info															
Events page															X
Fundraising page												X	X	X	
About Us page															
Newsletter									X	X	X				
Volunteer signup															
Board member application															
Collect information about supplies			X	X	X	X	X	X							
Donate Item Capability															
Animal Information	**X**	X													
Modify Website															
Modify Database												X	X	X	X

	U46	U47	U48	U49	U50	U51	U52	U53	U54,55
Accept Online payments									
Anybody can donate									
Collect Camper Info									
Collect Volunteer Info									
Collect Donor Info						X	X	X	
Collect fundraiser info									
Events page	**X**	X							
Fundraising page									
About Us page									
Newsletter									
Volunteer signup									
Board member application									
Collect information about supplies									
Donate Item Capability									
Animal Information									
Modify Website									
Modify Database	**X**	X	X	X	X	X	X	X	X

Use Cases

Use Case Specification 1: SignUpForNewsletter

Use-Case Name

Brief Description

This use case allows users to sign up for HOOF's newsletter. By submitting their information HOOF can have a comprehensive list of everyone who wants to receive their newsletter by email. By having all this information in one place it will be easy for HOOF to send mass emails out to those who are interested in knowing more about their organization.

Flow of Events

Basic Flow

1. Visitor navigates to HOOF website
2. Visitor clicks "Sign Up for Newsletter"
3. Visitor enters first name
4. Visitor enters last name
5. Visitor enters email address
6. Visitor submits information by clicking "Submit"
7. System validates information
8. System verifies information
9. End use case

Alternative Flows

Visitor enters invalid email address

3a. System prompts visitor to reenter a correct email address

Visitor is already a member

8a. Members already are subscribed to the email automatically, so if the system finds a matching email address already in the system it will say "Email address already subscribed!"

Special Requirements

No special Requirements

Pre-conditions

User must have access to HOOF website.

Post-conditions

User newsletter signup information is stored in database.

Extension Points

No extension points

Use Case Specification 2: AccessDonationInformation

Brief Description

This use case describes how a site administrator can access information on about donations that have been made. Allowing administrators to access this will inform them on the things HOOF has received from its donors.

Flow of Events

Basic Flow

1. Admin navigates to HOOF website
2. Admin Logs in as a site Admin, execute Log In use case
3. Admin navigates to Administrator home page
4. Admin selects “Donations” page
5. Admin selects “Item Donations”
6. Admin selects “Accept” to accept an individual donation and keep the record permanently
7. End use case

Alternative Flows

No Donations have been made

If no donations have been made yet a message will appear where the donation records would be shown that displays “No donations have been made yet.”

Admin only wants to see donations made for a certain time period

The site administrator will have control to filter donation records by time period, item and donator.

Admin decides not to accept a donation

Admin may select “Decline” to decline an offer

Admin accidentally selects “Decline”

- Admin selects “Declined Items”
- Admin selects item they accidentally declined
- Admin selects “Move to Donations” and can do so within 24 hours of selecting “Decline”

Special Requirements

No Special Requirements

Pre-conditions

Visitor must have access to HOOF website , Visitor must have Administrator access

Post-conditions

Admin Selects “Accept”

If an admin accepts a donation an email will be sent to the donor alerting them that HOOF would like to accept their donation. It will alert of them of the different drop off locations where they can deliver their items.

Admin Selects “Decline”

If an admin declines a donation they have 24 hours to change their mind and place the item back in the donation records. After 24 hours an email will be sent to the donor alerting them that HOOF is no longer in need of their donation items.

Extension Points

No extension points

Use Case Specification 3: Sign Up to Donate Goods

Brief Description

This use case describes how a user signs up to donate different types of goods to HOOF on their website. This will allow users to easily let HOOF know what they would like to donate and gives HOOF all the information they need to contact them.

Flow of Events

Basic Flow

1. Visitor navigates to HOOF website
2. Visitor selects “Log In”, execute Log In use case
3. The member selects “Donate Goods” button
4. User selects from a drop down list of items to donate
5. User submits form by clicking the “Submit” button
6. Donation ID is automatically generated
7. System validates info
8. System verifies info
9. Use case ends

Alternative Flows

Visitor is not a member

2a. User cannot log in because they are not a member yet. The system will navigate to a page to let them create an account

Member decides not to donate

5a. Member selects “Cancel Donation”

Special Requirements

No special requirements

Pre-conditions

User must have access to the HOOF website.

System must have available list of items that HOOF is in need of that the user can select from.

4.3 Visitor must be a member

Post-conditions

The donation information is stored in the donation database.

Extension Points

No extension points

Use Case Specification 4: AccessVolunteerWorkInformation

Use-Case Name

Brief Description

This use case describes how site administrators can access volunteer work information. It will specify how many hours were worked by each volunteer as well as their HOOF volunteer history.

Flow of Events

Basic Flow

1. Admin navigates to HOOF website
2. Admin logs in as site Admin, execute Log In use case
3. Admin navigates to Administrator home page
4. Admin selects “Volunteer Records”
5. Admin enters volunteer ID, first name or last name into search field
6. Admin submits search
7. Admin selects correct result
8. End use case

Alternative Flows

Search cannot find Volunteer

If data is entered incorrectly or the search cannot find a matching volunteer record to match the administrator’s search it will display a message: “Cannot find matching volunteer”

Admin selects wrong Volunteer

If an admin selects the wrong volunteer they can select “Return to search” to return to the search page

Special Requirements

No Special Requirements

Pre-conditions

Visitor must have access to HOOF website

Visitor must have administrative access

Post-conditions

Admin can view Volunteer records

Extension Points

No extension points

Use Case Specification 5: SignUpForVolunteerWork

Use-Case Name

Brief Description

This use case describes how a user signs up for volunteer work for HOOF. This allows them to input their information as well as the times they are available to volunteer. By having all this information in one place it will be easy for HOOF to see who would like to volunteer at their camps and what times they can be scheduled.

Flow of Events

Basic Flow

1. User navigates to HOOF website
2. User selects “Log In”, execute LogIn use case
3. The use case starts when the user selects the “Volunteer” button
4. User selects the camp they would like to volunteer at from a drop down list of current camps
5. User enters available times they can volunteer in the form
6. User submits form by clicking “Submit” button
7. Volunteer ID is automatically generated
8. Confirmation email is sent to user’s email address
9. Use case ends

Alternative Flows

Member has already entered volunteer hours

4a. System asks member “Would you like to edit your volunteer availability?”

4a1. Member selects “Yes” and edits the form

4a2. Member selects “No” and is navigated back to home page

Special Requirements

No special requirements

Pre-conditions

User must have access to HOOF website

4.2 User must have logged in (Log In UC)

Post-conditions

Volunteer information is stored in database

Extension Points

No extension points

Use case: Access HOOF camp session information
ID: 6
Risk Level: High
Description: The purpose of this use case is to show how a visitor can gain access to information about HOOF camp sessions through the HOOF KY website.
Actors: Visitor
Preconditions: 1. The visitor must have access to the internet to view the HOOF KY website
Flow of events: 1. Visitor opens http://www.hoofky.org website 2. Visitor clicks on the tab labeled “Summer Camp” 3. Visitor gains information about HOOF KY camp sessions
Postconditions: 1. The person visiting the HOOF KY website has been directed to the camp session information page.
Alternative flow of events: 1. At any time, the visitor may choose to close the website.

Use case: Access information about HOOF board members
ID: 7
Risk Level: Low
Description: The purpose of this use case is to show how a visitor can gain access to information about HOOF KY board members through the HOOF KY website.
Actors: Visitor
Preconditions: 1. The visitor must have access to the internet to view the HOOF KY website
Flow of events: 1. Visitor opens http://www.hoofky.org website 2. Visitor clicks on the tab labeled “Our Board” 3. Visitor has access to information about the HOOF KY Board of Directors
Postconditions: 1. The person visiting the HOOF KY website has been directed to the board members information page.
Alternative flow of events: 1. At any time, the visitor may choose to close the website.

Use Case: **Apply for Board Position**
ID: UC8
Actors: Visitor
Description: This use case is intended to be used by a visitor to the HOOF website in order to apply for a board membership position.
Risk Level: medium
Preconditions: Must be able to open HOOF website
Flow of Events: Actor logs in Actor navigates to the application web page Actor inputs First Name Actor inputs Last Name Actor inputs email Actor inputs phone number Actor inputs birthdate Actor inputs previous job experience Actor inputs extra information Actor submits **Alternative Flow:** Actor can close out of web page at anytime
Post conditions: Hoof board members receive application

Use case: Create information about HOOF charity events
ID: 9
Risk Level: Medium
Description: The purpose of this use case is to show how a HOOF KY staff member can create information involving charity events.
Actors: Staff
Preconditions: 1. The visitor must have access to the internet to view the HOOF KY website, as well as access to configure the website
Flow of events: 1. Staff member visits WordPress website 2. Staff member enters administration username for HOOF KY website 3. Staff member enters administration password for HOOF KY website 4. Staff member is directed to WordPress home page for HOOF KY website 5. Staff member clicks on “Edit Page” to edit the website pages 6. Staff member chooses to edit the charity events tab 7. Staff member inputs new information regarding charity events into the editor 8. Staff member clicks the “Save” button to save the inputted information 9. Staff member created information about HOOF charity events
Postconditions: 1. The staff member that oversees editing of the website has created information about HOOF charity events
Alternative flow of events: 1. At any time, the staff member may choose to close the website 2. At any time, the staff member may choose to stop editing the page

Use case: Edit information about HOOF charity events
ID: 10
Risk Level: Medium
Description: The purpose of this use case is to show how a HOOF KY staff member can edit information about charity events.
Actors: Staff
Preconditions: 1. The visitor must have access to the internet to view the HOOF KY website, as well as access to configure the website
Flow of events: 1. Staff member visits WordPress website 2. Staff member enters administration username for HOOF KY website 3. Staff member enters administration password for HOOF KY website 4. Staff member is directed to WordPress home page for HOOF KY website 5. Staff member clicks on “Edit Page” to edit the website pages 6. Staff member chooses to edit the charity events tab 7. Staff member inputs new information regarding charity events into the editor, and edits current information to meet updates 8. Staff member clicks the “Save” button to save the inputted information 9. Staff member has created and edited information about HOOF charity events on the HOOF KY website under the charity events tab
Postconditions: 1. The staff member that oversees editing of the website has edited necessary information under the charity events tab on the HOOF KY website
Alternative flow of events: 1. At any time, the staff member may choose to close the website 2. At any time, the staff member may choose to stop editing the page

Use case: Delete information about HOOF charity events
ID: 11
Risk Level: Medium
Description: The purpose of this use case is to show how a HOOF KY staff member can delete information about charity events.
Actors: Staff
Preconditions: 2. The visitor must have access to the internet to view the HOOF KY website, as well as access to configure the website
Flow of events: 1. Staff member visits WordPress website 2. Staff member enters administration username for HOOF KY website 3. Staff member enters administration password for HOOF KY website 4. Staff member is directed to WordPress home page for HOOF KY website 5. Staff member clicks on “Edit Page” to edit the website pages 6. Staff member chooses to edit the charity events tab 7. Staff member selects the information that is to be deleted 8. Staff member deletes irrelevant information regarding charity events using the editor 9. Staff member clicks the “Save” button to save the inputted information 10. Staff member deletes information about HOOF charity events
Postconditions: 2. The staff member that oversees editing of the website has deleted irrelevant information about HOOF charity events
Alternative flow of event: 1. At any time, the staff member may choose to close the website 2. At any time, the staff member may choose to stop editing the page

Use case: Input data on children at camps
ID: 12
Risk Level: High
Description: The purpose of this use case is to show how a HOOF KY staff member can navigate through SQL Server database management system to input data on the children attending HOOF camps into the database.
Actors: Staff
Preconditions: 1. The staff member must have access to the database in which all the information about camp attendees is held and must have permissions to input data into the database
Flow of events: 1. Staff member opens SQL Server database manager 2. Staff member selects the server type 3. Staff member enters database server name 4. Staff member chooses authentication server 5. Staff member types in username 6. Staff member types in password 7. Staff member presses “connect” to connect to the HOOF KY database 8. Staff member navigates to the child attendee information database 9. Staff member inputs new data of children attending camps into the database
Postconditions: 1. The staff member has inputted new data into the camp attendee database
Alternative flow of events: 2. At any time, the visitor may choose to close the database

Use case: Modify data on children at camps
ID: 13
Risk Level: High
Description: The purpose of this use case is to show how a HOOF KY staff member can navigate through SQL Server database management system to modify current data of the children attending HOOF camps within the database.
Actors: Staff
Preconditions: 1. The staff member must have access to the database and must also have permissions to modify the current recorded data of the children attending camp
Flow of events: 1. Staff member opens SQL Server database manager 2. Staff member selects the server type 3. Staff member enters database server name 4. Staff member chooses authentication server 5. Staff member types in username 6. Staff member types in password 7. Staff member presses “connect” to connect to the HOOF KY database 8. Staff member navigates to the child attendee information database 9. Staff member updates and modifies data on children at camps
Postconditions: 3. The staff member has modified and updated the current data of children attending HOOF camps
Alternative flow of events: 3. At any time, the visitor may choose to close the database

Use case: Delete data on children at camps
ID: 14
Risk Level: High
Description: The purpose of this use case is to show how a HOOF KY staff member can navigate through SQL Server database management system to delete data on the children attending HOOF camps from the database.
Actors: Staff
Preconditions: 1. The staff member must have access to the database and must also have permissions to delete current recorded data of the children attending camp
Flow of events: 1. Staff member opens SQL Server database manager 2. Staff member selects the server type 3. Staff member enters database server name 4. Staff member chooses authentication server 5. Staff member types in username 6. Staff member types in password 7. Staff member presses “connect” to connect to the HOOF KY database 8. Staff member navigates to the child attendee information database 9. Staff member deletes necessary data on children at camps
Postconditions: 4. The staff member has deleted data that had been recorded for the children attending HOOF camps
Alternative flow of events: 4. At any time, the visitor may choose to close the database

Use case: Input Board Member Data
ID: 15
Risk Level: High
Description: The purpose of this use case is for a HOOF KY staff member to input important board member information such as name, contact information, and how long they have served on the board. Putting all of this information into one organized space in case they are ever in need of reporting this information.
Preconditions: - Staff member must have access to the database, will full ability to create, modify, and delete - Table to hold information in database must be created
Flow of Events: 1. Staff member opens SQL Server Database Manager 2. Staff member selects the server type 3. Staff member enters database server name 4. Staff member chooses authentication server 5. Staff member types in username 6. Staff member types in password 7. Staff member presses “connect” to connect to the HOOF KY database 8. Staff member connects to BOARD_MEMBER database 9. Staff member enters in necessary information
Postconditions: - Staff member validates the database has been populated with correct information
Alternative Flow of events: - At any time, the staff member may close the database

Use case: Modify Board Member Data
ID: 16
Risk Level: High
Description: The purpose of this use case is for a HOOF KY staff member to Modify important board member information such as name, contact information, and how long they have served on the board. Putting all of this information into one organized space in case they are ever in need of reporting this information.
Preconditions: - Staff member must have access to the database, will full ability to create, modify, and delete - Table to hold information in database must be created
Flow of Events: 1. Staff member opens SQL Server Database Manager 2. Staff member selects the server type 3. Staff member enters database server name 4. Staff member chooses authentication server 5. Staff member types in username 6. Staff member types in password 7. Staff member presses “connect” to connect to the HOOF KY database 8. Staff member connects to BOARD_MEMBER database 9. Staff member modifies the necessary information
Postconditions: - Staff member validates the database has been populated with correct information
Alternative Flow of events: - At any time, the staff member may close the database

Use case: Delete Board Member Data
ID: 17
Risk Level: High
Description: The purpose of this use case is for a HOOF KY staff member to delete expired or unneeded board member information such as name, contact information, and how long they have served on the board. Putting all of this information into one organized space in case they are ever in need of reporting this information.
Preconditions: - Staff member must have access to the database, will full ability to create, modify, and delete - Table to hold information in database must be created
Flow of Events: 1. Staff member opens SQL Server Database Manager 2. Staff member selects the server type 3. Staff member enters database server name 4. Staff member chooses authentication server 5. Staff member types in username 6. Staff member types in password 7. Staff member presses “connect” to connect to the HOOF KY database 8. Staff member connects to BOARD_MEMBER database 9. Staff member deltes the necessary information
Postconditions: - Staff member validates the correct data has been deleted
Alternative Flow of events: - At any time, the staff member may close the database

Use case: Input Donor Data
ID: 18
Risk Level: High
Description: The purpose of this use case is for a HOOF KY staff member to input important donor information such as name, contact information, how much they have donated, and what medium they use to donate. Putting all of this information into one organized space in case they are ever in need of reporting this information.
Preconditions: - Staff member must have access to the database, will full ability to create, modify, and delete - Table to hold information in database must be created
Flow of Events: 1. Staff member opens SQL Server Database Manager 2. Staff member selects the server type 3. Staff member enters database server name 4. Staff member chooses authentication server 5. Staff member types in username 6. Staff member types in password 7. Staff member presses “connect” to connect to the HOOF KY database 8. Staff member connects to DONOR database 9. Staff member enters in necessary information
Postconditions: - Staff member validates the database has been populated with correct information
Alternative Flow of events: - At any time, the staff member may close the database

Use case: Modify Donor Data
ID: 19
Risk Level: High
Description: The purpose of this use case is for a HOOF KY staff member to Modify important Donor information such as name, contact information, how much they donated, and what medium they use to donate. Putting all of this information into one organized space in case they are ever in need of reporting this information.
Preconditions: - Staff member must have access to the database, will full ability to create, modify, and delete - Table to hold information in database must be created
Flow of Events: 1. Staff member opens SQL Server Database Manager 2. Staff member selects the server type 3. Staff member enters database server name 4. Staff member chooses authentication server 5. Staff member types in username 6. Staff member types in password 7. Staff member presses “connect” to connect to the HOOF KY database 8. Staff member connects to DONOR database 9. Staff member modifies the necessary information
Postconditions: - Staff member validates the database has been populated with correct information
Alternative Flow of events: - At any time, the staff member may close the database

Use case: Delete Donor Data
ID: 20
Risk Level: High
Description: The purpose of this use case is for a HOOF KY staff member to delete expired or unneeded Donor information such as name, contact information, how much they donated, and what medium they use to donate. Putting all of this information into one organized space in case they are ever in need of reporting this information.
Preconditions: - Staff member must have access to the database, will full ability to create, modify, and delete - Table to hold information in database must be created
Flow of Events: 1. Staff member opens SQL Server Database Manager 2. Staff member selects the server type 3. Staff member enters database server name 4. Staff member chooses authentication server 5. Staff member types in username 6. Staff member types in password 7. Staff member presses “connect” to connect to the HOOF KY database 8. Staff member connects to DONOR database 9. Staff member deletes necessary information
Postconditions: - Staff member validates the correct data has been deleted
Alternative Flow of events: - At any time, the staff member may close the database

Use case: Input Financial Data
ID: 21
Risk Level: High
Description: The purpose of this use case is for a HOOF KY staff member to input important quarterly financial data like how much they received in donations, grants, camper revenue, and how much they spent on costs. Putting all of this information into one organized space in case they are ever in need of reporting this information.
Preconditions: - Staff member must have access to the database, will full ability to create, modify, and delete - Table to hold information in database must be created
Flow of Events: 1. Staff member opens SQL Server Database Manager 2. Staff member selects the server type 3. Staff member enters database server name 4. Staff member chooses authentication server 5. Staff member types in username 6. Staff member types in password 7. Staff member presses “connect” to connect to the HOOF KY database 8. Staff member connects to QUARTERLY_FINANCES database 9. Staff member modifies in necessary information
Postconditions: - Staff member validates the database has been populated with correct information
Alternative Flow of events: - At any time, the staff member may close the database

Use case: Modify Financial Data
ID: 22
Risk Level: High
Description: The purpose of this use case is for a HOOF KY staff member to modify important quarterly financial data like how much they received in donations, grants, camper revenue, and how much they spent on costs. Putting all of this information into one organized space in case they are ever in need of reporting this information.
Preconditions: - Staff member must have access to the database, will full ability to create, modify, and delete - Table to hold information in database must be created
Flow of Events: 1. Staff member opens SQL Server Database Manager 2. Staff member selects the server type 3. Staff member enters database server name 4. Staff member chooses authentication server 5. Staff member types in username 6. Staff member types in password 7. Staff member presses “connect” to connect to the HOOF KY database 8. Staff member connects to QUARTERLY_FINANCES database 9. Staff member modifies the necessary information
Postconditions: - Staff member validates the database has been populated with correct information
Alternative Flow of events: - At any time, the staff member may close the database

Use case: Delete Financial Data
ID: 23
Risk Level: High
Description: The purpose of this use case is for a HOOF KY staff member to delete expired or unneeded quarterly financial data like how much they received in donations, grants, camper revenue, and how much they spent on costs. Putting all of this information into one organized space in case they are ever in need of reporting this information.
Preconditions: - Staff member must have access to the database, will full ability to create, modify, and delete - Table must be populated with information - Table to hold information in database must be created
Flow of Events: 1. Staff member opens SQL Server Database Manager 2. Staff member selects the server type 3. Staff member enters database server name 4. Staff member chooses authentication server 5. Staff member types in username 6. Staff member types in password 7. Staff member presses “connect” to connect to the HOOF KY database 8. Staff member connects to QUARTERLY_FINANCES database 9. Staff member deletes the necessary information
Postconditions: - Staff member validates the correct data has been
Alternative Flow of events: - At any time, the staff member may close the database

Use Case Specification 24: Input Volunteer Use Case

Use-Case Name

Brief Description

The use case will enable users to enter information about the volunteers

Flow of Events

Basic Flow

- Enter volunteer database
- Enter first name
- Enter last name
- Enter street
- Enter city
- Enter state
- Enter zip
- Enter volunteer event name
- Enter volunteer day
    - Month, day, year
- Enter volunteer start time in hours and minutes
- Enter volunteer end time in hours and minutes
- Use case ends

Alternative Flows

< First Alternative Flow >

- The volunteer enters invalid data
    - Validate data
    - Send error to user on form that the input is incorrect
    - Error should let them know what valid data is/looks-like

Special Requirements

< First Special Requirement >

- The database will be able to store at least 1,000 volunteer records

<Second Special Requirement >

- the database will be available to board members 99% of the time

< Third Special Requirement >

- Data will be backed-up weekly

Pre-conditions

- The user is logged in

Post-conditions

- A volunteer is created

Use Case Specification 25: Modify Volunteer Use Case

Brief Description

The modify data on volunteers use case will enable board members to change data on volunteers in their database. This can be done in the event that the volunteer enters incorrect data.

Flow of Events

Basic Flow

- Staff searches database for field where data is to be modified
- The search returns the field to be modified
- The field is modified
- The users saves the modification to the database
- Use case ends

Alternative Flows

First Alternative Flow: User modifies incorrect record but is able to click undo

- User modifies incorrect record
- User notices immediately
- User clicks undo
- User locates correct record for modification
- User modifies correct record
- User saves changes
- User backs-up changes

Second Alternative Flow: User modifies incorrect record and has to restore from back-up

- User modifies incorrect record
- User doesn't notice until a later date
- User goes to back-up
- User restores data
- User modifies correct record from restored data
- User saves changes
- User backs-up changes

Special Requirements

First Special Requirement

- "Modify volunteer data" may apply up to 2,000 updates per work day

3.2 Second Special Requirement

- the database will be available to board members 99% of the time

3.3 Third Special Requirement

- Data is backed-up weekly

Pre-conditions

Pre-condition One

- User must be logged in to database

Pre-condition Two

- The record/field to be modified must exist

Post-conditions

Post-condition One

- The user has successfully updated a volunteer record

Use Case Specification 26: Delete Volunteer Use Case

Use-Case Name

Brief Description

This use case will enable members with access to the volunteer database the ability to delete volunteer records that are no longer needed. It will also provide them with an alternative flow if they delete the wrong record and how to correct that.

Flow of Events

Basic Flow

User queries database to record or field that needs to be deleted

Field/record is located

Field/record is deleted

Changes to database are saved

Database is backed-up

Use case ends

Alternative Flows

First Alternative Flow

- User accidently deleted unintended record.
- User goes to backed up data to recover record
- User uses the back to restore data
- User re-enters database
- User deletes appropriate record

An Alternative Subflow

Second Alternative Flow

- The user deletes a record
- The record wasn't the intended record to be deleted
- Mistake is realized immediately after action
- User clicks undo button to undo the deletion
- User finds the correct record
- Correct record is deleted

Special Requirements

First Special Requirement

- the database will be available to board members 99% of the time
- Able to delete 2,000 volunteer records a day
- Data is backed-up weekly

Pre-conditions

- User is logged into database
- The record to be delete exists

Post-conditions

- The user has successfully deleted the volunteer record that was indented to be deleted

Use case: Input Fundraiser Attendee
ID: 27
Brief Description: Add details about a fundraiser attendee to the database
Primary Actors: HOOF Board Member
Secondary Actors: None
Preconditions: 1. The board member is logged into the system
Main flow: 1. The Board member selects “add volunteer”. 2. The board member enters the first name of the fundraiser attendee. 3. The board member enters the last name of the fundraiser attendee. 4. The board member enters the address of the fundraiser attendee. 5. The board member enters the city. 6. The board member enters the state. 7. The board member enters the zip. 8. The board member enters the month, day, and year of the event attended. 9. The board member enters the name of the event attended. 10. The system creates a new fundraiser attendee.
Postcondition: 1. A new fundraiser attendee has been created
Alternative flows: FundraiserAttendeeAlreadyExists

Use case: Modify Fundraiser attendee
ID: 28
Brief Description: Edit details about a fundraiser attendee to the database
Primary Actors: HOOF Board Member
Secondary Actors: None
Preconditions: 1. The board member is logged into the system
Main flow: 1. The Board member selects “modify fundraiser attendee”. 2. The board member selects which fundraiser attendee to modify. 3. The board member selects which attribute to modify. 4. The system edits the fundraiser attendee
Postcondition: 1. A fundraiser attendee attribute has been modified
Alternative flows: AccidentallyModifyWrongFundraiser attendee

Use case: Delete Fundraiser attendee
ID: 29
Brief Description: Delete fundraiser attendee from the database
Primary Actors: HOOF Board Member
Secondary Actors: None
Preconditions: 1. The board member is logged into the system
Main flow: 1. The Board member selects “delete fundraiser attendee”. 2. The board member selects which fundraiser attendee to delete. 3. The system deletes the fundraiser attendee.
Postcondition: 1. A fundraiser attendee attribute has been deleted
Alternative flows: DeleteWrongFundraiser attendee

Use case: Input Animal
ID: 30
Brief Description: Add details about an animal to the animal database
Primary Actors: HOOF Board Member
Secondary Actors: None
Preconditions: 1. The board member is logged into the system
Main flow: 1. The Board member selects “add animal”. 2. The board member enters the name of the animal 3. The board member enters the first name of the owner of the animal. 4. The board member enters the last name of the owner of the animal. 5. The board member enters the address of the owner of the animal. 6. The board member enters the city of the owner of the animal. 7. The board member enters the state of the owner of the animal. 8. The board member enters the zip of the owner of the animal. 9. The board member enters the phone of the owner of the animal. 10. The board member enters the condition of the animal (ridable: Y/N). 11. The board member enters any veterinary work on the animal. 12. The board member enters any medication for the animal. 13. The system creates a new animal
Postcondition: 1. A new animal has been created
Alternative flows: AnimalAlreadyExists

Use case: Modify Animal
ID:31
Brief Description: Edit details about an animal to the animal database
Primary Actors: HOOF Board Member
Secondary Actors: None
Preconditions: 1. The board member is logged into the system
Main flow: 1.The Board member selects “modify animal”. 1. The board member selects which animal to modify. 2. The board member selects which attribute to modify. 3. The system edits the animal
Postcondition: 1. An animal attribute has been modified
Alternative flows: AccidentallyModifyWrongAnimal

Use case: Delete Animal
ID:32
Brief Description: Delete animal from the database
Primary Actors: HOOF Board Member
Secondary Actors: None
Preconditions: 1. The board member is logged into the system
Main flow: 1. The Board member selects “delete animal”. 2. The board member selects which animal to delete. 3. The system deletes the animal.
Postcondition: 1. An animal attribute has been deleted
Alternative flows: DeleteWrongAnimal

Use Case: Input Data on Feed Inventory
ID: 33
Risk Level: High
Description: This use case is for when new horse feed is obtained and a HOOF employee adds data concerning the new item to the database
Primary Actor: Staff
Preconditions: 1. New horse feed is obtained 2. The staff member has access to the database and permission to input data
Flow of Events: 1. Use case starts when new horse feed is obtained 2. Staff member opens the database manager 3. Staff member selects the server type 4. Staff member enters the database server name 5. Staff member choses authentication server 6. Staff member types in username 7. Staff member types in password 8. Staff member connects to the database 9. Staff member opens the feed inventory database 10. Staff member inputs data on new horse feed a. Staff member enters ID of horse feed b. Staff member enters name of horse feed c. Staff member enters description of horse feed d. Staff member enters amount of horse feed e. Staff member checks yes if the feed was donated; no if it was not donated i. If yes, staff member selects ID of the donor 11. Use case ends
Postconditions: 1. Data on the new horse feed is successfully added to the database
Alternative Flows: None

Use Case: Modify Data on Feed Inventory
ID: 34
Risk Level: High
Description: This use case is for when a HOOF employee needs to modify information (i.e. the amount of feed in inventory changes or the name needs to be changed) about a type of horse feed in the database.
Primary Actor: Staff
Preconditions: 1. Information about an item in the Feed Inventory Database needs to be updated 2. The staff member has access to the database and permission to modify data
Flow of Events: 1. Use case starts when the amount of a type of horse feed is increased or decreased 2. Staff member opens the database manager 3. Staff member selects the server type 4. Staff member enters the database server name 5. Staff member choses authentication server 6. Staff member types in username 7. Staff member types in password 8. Staff member connects to the database 9. Staff member opens the feed inventory database 10. Staff member modifies horse feed data a. Staff member selects the name and ID of the feed they intend to modify b. Staff member enters new ID of horse feed c. Staff member enters new name of horse feed d. Staff member enters new description of horse feed e. Staff member enters new amount of horse feed f. Staff member checks yes if the feed was donated; no if it was not donated i. If yes, staff member selects ID of the donor 11. Use Case ends
Postconditions: 1. Horse feed data is successfully modified
Alternative Flows: None

Use Case: Delete Data on Feed Inventory
ID: 35
Risk Level: High
Description: This use case is for when a type of horse feed runs out or is removed form inventory and a HOOF employee deletes its data from the database
Primary Actor: Staff
Preconditions: 1. A type of horse feed in inventory runs out OR is removed from inventory 2. The staff member has access to the database and permission to delete data
Flow of Events: 1. Use case starts when a type horse feed is no longer in inventory 2. Staff member opens the database manager 3. Staff member selects the server type 4. Staff member enters the database server name 5. Staff member choses authentication server 6. Staff member types in username 7. Staff member types in password 8. Staff member connects to the database 9. Staff member opens the feed inventory database 10. Staff member deletes horse feed data a. Staff member selects the name and ID of the feed they intend to delete
Postconditions: 1. Horse feed data is successfully deleted
Alternative Flows: None

Use Case: Input Data on Gear Inventory
IDi: 36
Risk Level: High
Description: This use case is for when new gear is obtained and a HOOF employee needs to add its data to the database
Primary Actor: Staff
Preconditions: 1. New gear is obtained 2. The staff member has access to the database and permission to input data
Flow of Events: 1. Use case starts when new gear is obtained 2. Staff member opens the database manager 3. Staff member selects the server type 4. Staff member enters the database server name 5. Staff member choses authentication server 6. Staff member types in username 7. Staff member types in password 8. Staff member connects to the database 9. Staff member opens the gear inventory database 10. Staff member inputs the gear data a. Staff member enters the ID of the gear b. Staff member enters the name of the gear c. Staff member enters the type of the gear d. Staff member checks yes if the gear was donated; no if it was not donated i. If yes, staff member selects the ID of the donor 11. Use case ends
Postconditions: 1. Data on the new gear is successfully added to the database
Alternative Flows: None

Use Case: Modify Data on Gear Inventory
ID: 37
Risk Level: High
Description: This use case is for when the information concerning a gear item changes and a HOOF employee needs to modify its data in the database
Primary Actor: Staff
Preconditions: 1. The staff member has access to the database and permission to input data
Flow of Events: 1. Use case starts when information concerning a type of gear needs to be changed 2. Staff member opens the database manager 3. Staff member selects the server type 4. Staff member enters the database server name 5. Staff member choses authentication server 6. Staff member types in username 7. Staff member types in password 8. Staff member connects to the database 9. Staff member opens the gear inventory database 10. Staff member modifies the gear data a. Staff member selects the name and ID of the gear they intend to modify b. Staff member enters the new ID of the gear c. Staff member enters the new name of the gear d. Staff member enters the new type of the gear e. Staff member checks yes if the gear was donated; no if it was not donated i. If yes, staff member selects the ID of the donor f. Use case ends
Postconditions: 1. Gear data is successfully modified
Alternative Flows: None

Use Case: Delete Data on Gear Inventory
ID: 38
Risk Level: High
Description: This use case is for when a type of gear is no longer inventory and a HOOF employee needs to delete its data from the database
Primary Actor: Staff
Preconditions: 1. A type of gear is no longer in inventory 2. The staff member has access to the database and permission to input data
Flow of Events: 1. Use case starts when a type of gear is no longer in inventory 2. Staff member opens the database manager 3. Staff member selects the server type 4. Staff member enters the database server name 5. Staff member choses authentication server 6. Staff member types in username 7. Staff member types in password 8. Staff member connects to the database 9. Staff member opens the gear inventory database 10. Staff member deletes the gear data a. Staff member selects the name and ID of the gear they intend to delete
Postconditions: 1. Gear data is successfully deleted
Alternative Flows: None

Use Case: Post Newsletter
ID: 39
Risk Level: Med
Description: This use case is for when a HOOF employee needs to post a new issue of a newsletter to the website.
Primary Actor: Staff
Preconditions: 3. Newsletter needs to be posted 4. The staff member has administrative access to the HOOF website
Flow of Events: 12. Use case starts when newsletter needs to be posted 13. Execute Admin Login use case 14. Staff member enters data for the newsletter a. Staff member selects the date the newsletter is being posted b. Staff member types in the content of the newsletter in the text box c. Staff member hits “Post Newsletter” button 15. Use case ends
Postconditions: 2. The newsletter issue is successfully posted to the website
Alternative Flows: 1. The staff member forgets their login information

Use Case: Modify Newsletter
ID: 40
Risk Level: Med
Description: This use case is for when a HOOF employee needs to modify an issue of a newsletter that has already been posted to the website
Primary Actor: Staff
Preconditions: 1. Newsletter needs to be modified 2. The staff member has administrative access to the HOOF website
Flow of Events: 1. Use case starts when newsletter needs to be modified 2. Execute Admin Login use case 3. Staff member modifies data for the newsletter a. Staff member selects the newsletter they intend to modify b. Staff member types in the content of the newsletter in the text box c. Staff member hits “Modify Newsletter” button 4. Use case ends
Postconditions: 1. The newsletter issue is successfully modified
Alternative Flows: 1. The staff member forgets their login information

Use Case: Delete Newsletter
ID: 41
Risk Level: Med
Description: This use case is for when a HOOF employee needs to delete an issue of a newsletter that has already been posted to the website
Primary Actor: Staff
Preconditions: 1. Newsletter needs to be deleted 2. The staff member has administrative access to the HOOF website
Flow of Events: 1. Use case starts when newsletter needs to be deleted 2. Execute Admin Login use case 3. Staff member modifies data for the newsletter a. Staff member selects the newsletter they intend to modify b. Staff member hits the “Delete Newsletter” button 4. Use case ends
Postconditions: 1. Issue of newsletter is successfully deleted from the website
Alternative Flows: 1. The staff member forgets their login information

Use Case: **Input data on fundraiser income**
ID: UC42
Actors: HOOF Staff
Description: Use case is designed for HOOF staff member to input data on income received from a particular fundraiser
Risk Level: High
Preconditions: Actor must have access to SQL server and authority to add information
Flow of Events: Actor logs in Actor navigates to the Fundraiser Income web page Actor inputs fundraiser title Actor inputs fundraiser date Actor inputs fundraiser guest count Actor inputs fundraiser location Actor inputs total income Actor inputs total expenses Actor submits **Alternative Flow:** Actor can close website at anytime
Post conditions: New field in Fundraiser Income database is created

Use Case: **Modify Data on fundraiser income**
ID: UC43
Actors: Hoof Staff Member
Description: This use case is intended to be used by a HOOF staff member to modify any data in the fundraiser income database
Risk Level: high
Preconditions: The desired field in the fundraiser income database must already exist and the actor must have the required level of access to modify it
Flow of Events: Actor logs in Actor navigates to the Fundraiser Income web page Actor inputs fundraiser title Actor inputs fundraiser date Actor selects field Actor inputs new data Actor submits **Alternative Flow:** Actor can close out of web page at anytime
Post conditions: Fundraiser Income database is updated

Use Case: **Delete Data on fundraiser income**
ID: UC44
Actors: Hoof Staff Member
Description: This use case is intended to be used by a HOOF staff member to delete any data in the fundraiser income database
Risk Level: high
Preconditions: The desired field in the fundraiser income database must already exist and the actor must have the required level of access to delete it
Flow of Events: Actor logs in Actor navigates to the Fundraiser Income web page Actor inputs fundraiser title Actor inputs fundraiser date Actor selects field Actor deletes field Actor saves **Alternative Flow:** Actor can close out of website at anytime
Post conditions: Fundraiser Income database is updated

Use Case: **Input data on Auction Items**
ID: UC45
Actors: HOOF Staff
Description: Use case is designed for HOOF staff member to input data on auction items received during silent auctions
Risk Level: Medium
Preconditions: Actor must have access to SQL server and authority to add information
Flow of Events: Actor logs in Actor navigates to the Auctions Items web page Actor inputs fundraiser title Actor inputs fundraiser date Actor inputs item name Actor inputs item donor's first name Actor inputs item donor's last name Actor inputs item donor's email Actor inputs item's final bid price Actor inputs winner's first name Actor inputs winner's last name Actor inputs winner's email address Actor saves **Alternative Flow:** Actor can close database at anytime
Post conditions: New field in Auction Item database is created

Use Case: **Modify data on Auction Items**
ID: UC46
Actors: HOOF Staff
Description: Use case is designed for HOOF staff member to modify data on auction items received during silent auctions
Risk Level: Medium
Preconditions: The desired field in the auction item database must already exist and the actor must have the required level of access to modify it
Flow of Events: Actor logs in Actor navigates to the Auctions Items web page Actor inputs fundraiser title Actor inputs fundraiser date Actor inputs item name Actor selects field Actor inputs new data Actor saves **Alternative Flow:** Actor can close website at anytime
Post conditions: Auction item database is updated

Use Case: **Delete data on Auction Items**
ID: UC47
Actors: HOOF Staff
Description: Use case is designed for HOOF staff member to delete data on auction items received during silent auctions
Risk Level: Medium
Preconditions: The desired field in the auction item database must already exist and the actor must have the required level of access to delete it
Flow of Events: Actor logs in Actor navigates to the Auction Item web page Actor inputs fundraiser title Actor inputs fundraiser date Actor inputs item name Actor selects field Actor deletes data Actor saves **Alternative Flow:** Actor can close database at anytime
Post conditions: Auction item database is updated

Use Case Specification 48: Input Grant

Brief Description

Input Grant will allow board members to enter grants that they receive into a database in order to track what they received, when they received it and how much the grant was.

Basic Flow

The board member is logged in

The board member enters the grant:

Grant name

- Grant amount
- Date received
- Grant description

Use case ends

Alternative Flows

First Alternative Flow: A grant has the same date received

The board member is logged in

The board member enters the grant:

Grant name

- Grant amount
- Date received
- Grant description
- The system validates that the grant name
- The system validates the date the grant was received
- If they're duplicates of another grant in the system then the system alters the board member that the grant they're trying to enter is a duplicate.

Second Alternative Flow: A user attempts to enter a grant without a description

The board member is logged in

The board member enters the grant:

Grant name

Grant amount

Date received

The user tries to enter a grant without a description

The system alters that each grant must have a description to clarify what type of grant it is.

Special Requirements

First Special Requirement

- The system shall keep grant info secure
- The system shall be able to keep all info inputted into it
- The system shall remain up 99% of the time

Pre-conditions

- The user is logged in

Post-conditions

- A grant has been created, A grant has been entered into the system

Use Case Specification 49: Modify grant Use Case

Brief Description

The modify data on grants use case will enable board members to change data on grants in their database. This can be done in the event that the grant enters incorrect data.

Flow of Events

Basic Flow

- Staff searches database for field where data is to be modified
- The search returns the field to be modified
- The field is modified
- The users saves the modification to the database
- Use case ends

Alternative Flows

First Alternative Flow: User modifies incorrect record but is able to click undo

- User modifies incorrect record
- User notices immediately
- User clicks undo
- User locates correct record for modification
- User modifies correct record
- User saves changes
- User backs-up changes

Second Alternative Flow: User modifies incorrect record and has to restore from back-up

- User modifies incorrect record
- User doesn't notice until a later date
- User goes to back-up
- User restores data
- User modifies correct record from restored data
- User saves changes
- User backs-up changes

Special Requirements

- "Modify grant data" may apply up to 2,000 updates per work day
- the database will be available to board members 99% of the time
- Data is backed-up weekly

Pre-conditions

- User must be logged in to database
- The record to be modified must exist

Post-conditions

- The user has successfully updated a grant record

Use Case Specification 50: Delete Grant Use Case

Brief Description

This use case will enable members with access to the Grant database the ability to delete grant records that are no longer needed. It will also provide them with an alternative flow if they delete the wrong record and how to correct that.

Flow of Events

Basic Flow

User queries database to record or field that needs to be deleted

Field/record is located

Field/record is deleted

Changes to database are saved

Database is backed-up

Use case ends

Alternative Flows

First Alternative Flow

- User accidently deleted unintended record.
- User goes to backed up data to recover record
- User uses the back to restore data
- User re-enters database
- User deletes appropriate record

An Alternative Subflow

Second Alternative Flow

- The user deletes a record
- The record wasn't the intended record to be deleted
- Mistake is realized immediately after action
- User clicks undo button to undo the deletion
- User finds the correct record
- Correct record is deleted

Special Requirements

First Special Requirement

- the database will be available to board members 99% of the time
- Able to delete 2,000 grant records a day
- Data is backed-up weekly

Pre-conditions

- User is logged into database
- The record to be delete exists

Post-conditions

- The user has successfully deleted the Grant record that was indented to be deleted

Use Case Specification 51: Input Grantor

Use-Case Name

Brief Description

Input Grantor will allow board members to enter grantors so they can track who they receive grants from and their contact information.

Flow of Events

Basic Flow

The board member is logged in

The board member enters the grantor:

- Grantor company name
- Grantor contact first name
- Grantor contact last name
- Grantor contact phone
- Grantor contact email
- Grantor contact address

Use case ends

Special Requirements

First Special Requirement

- The system shall keep grantor info secure
- The system shall be able to keep all info inputted into it
- The system shall remain up 99% of the time
- They system will allow duplicate company names but they can't have the same contact info

Pre-conditions

Pre-condition One

- The user is logged in

Post-conditions

Post-condition One

- A grantor has been created

Post-condition two

- A grantor has been entered into the system

Use Case Specification 52: Modify Grantor Use Case

Brief Description

The modify data on grantors use case will enable board members to change data on grantors in their database. This can be done in the event that the grantor enters incorrect data.

Flow of Events

Basic Flow

- Staff searches database for field where data is to be modified
- The search returns the field to be modified
- The field is modified
- The users saves the modification to the database
- Use case ends

Alternative Flows

First Alternative Flow: User modifies incorrect record but is able to click undo

- User modifies incorrect record
- User notices immediately
- User clicks undo
- User locates correct record for modification
- User modifies correct record
- User saves changes
- User backs-up changes

Second Alternative Flow: User modifies incorrect record and has to restore from back-up

- User modifies incorrect record
- User doesn't notice until a later date
- User goes to back-up
- User restores data
- User modifies correct record from restored data
- User saves changes
- User backs-up changes

Special Requirements

- "Modify grantor data" may apply up to 2,000 updates per work day
- the database will be available to board members 99% of the time
- Data is backed-up weekly

Pre-conditions

- User must be logged in to database
- The record/field to be modified must exist

Post-conditions

- The user has successfully updated a grantor record

Use Case Specification 53: Delete Grantor Use Case

Brief Description

This use case will enable members with access to the grantor database the ability to delete grantor records that are no longer needed. It will also provide them with an alternative flow if they delete the wrong record and how to correct that.

Flow of Events

Basic Flow

User queries database to record or field that needs to be deleted

Field/record is located

Field/record is deleted

Changes to database are saved

Database is backed-up

Use case ends

Alternative Flows

First Alternative Flow

- User accidently deleted unintended record.
- User goes to backed up data to recover record
- User uses the back to restore data
- User re-enters database
- User deletes appropriate record

An Alternative Subflow

Second Alternative Flow

- The user deletes a record
- The record wasn't the intended record to be deleted
- Mistake is realized immediately after action
- User clicks undo button to undo the deletion
- User finds the correct record
- Correct record is deleted

Special Requirements

- the database will be available to board members 99% of the time
- Able to delete 2,000 grantor records a day
- Data is backed-up weekly

Pre-conditions

- User is logged into database
- The record to be delete exists

Post-conditions

- The user has successfully deleted the grantor record that was indented to be deleted

Use case: Memeber Login
ID: 54
Brief Description: This use case describes how a user can create a user name and password to log in
Primary Actors: HOOF Website visitor
Secondary Actors: None
Preconditions: The user has access to the HOOF website
Main flow: 1.The user selects “Create an account” 2.The user creates a login user name 3.User creates password and confirms the password 4.The user selects “Member” 5.The user fills out first and last name 6.The user inputs their phone number 7.The user inputs their email 8.The user selects whether they’d like to receive a newsletter 9. User is automatically generated a user ID
Postcondition: A new user account has been created
Alternative flows: Forgot Password

Use case: Admin Login
ID: 55
Brief Description: This use case describes how a HOOF board member can create a user name and password to log in
Primary Actors: HOOF Board Member
Secondary Actors: None
Preconditions: The board member has access to the HOOF website
Main flow: The user selects “Create an account” 2.The user creates a login user name 3.User creates password and confirms the password 4.The user selects “Admin” 5.The user fills out first and last name 6.The user inputs their phone number 7.The user inputs their email 8.The user receives a message that they will be contacted when their admin status has been approved 9. User is automatically generated a user ID
Postcondition: A new admin account has been created
Alternative flows: Forgot Password

Sequence Diagrams

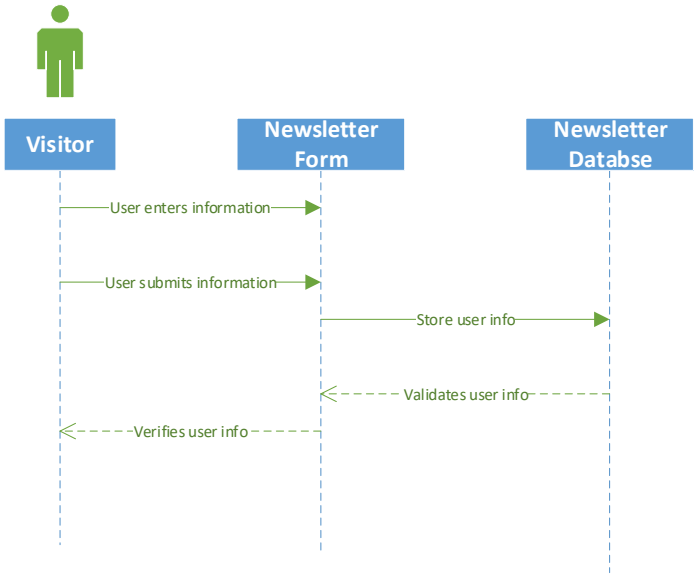
Sequence Diagrams

A sequence diagram displays a timeline of each object's behavior within a use case.

UC1: SignUpForNewsletter

Main Flow:

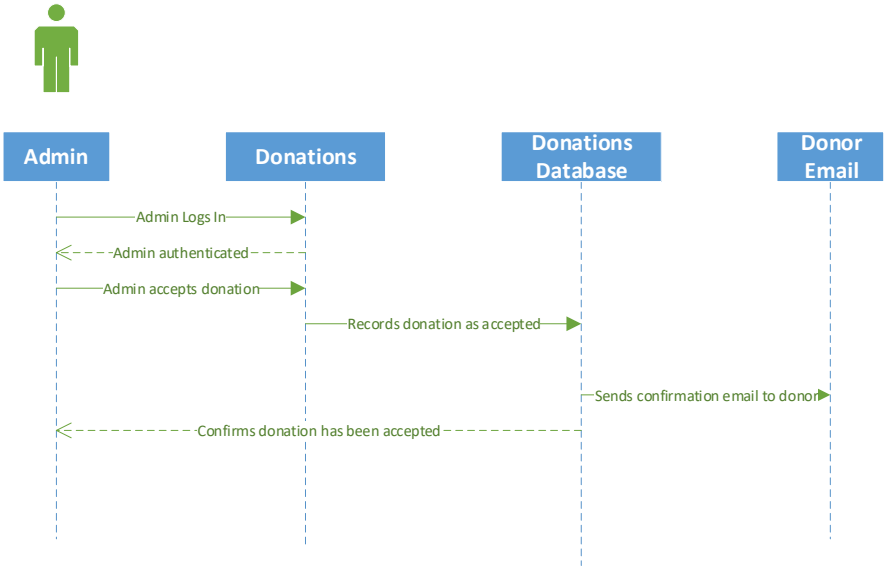
1. Visitor navigates to HOOF website
2. Visitor clicks "Sign Up for Newsletter"
3. Visitor enters first name
4. Visitor enters last name
5. Visitor enters email address
6. Visitor submits information by clicking "Submit"
7. System validates information
8. System verifies information
9. End use case



UC2: AccessDonationInformation

Main Flow:

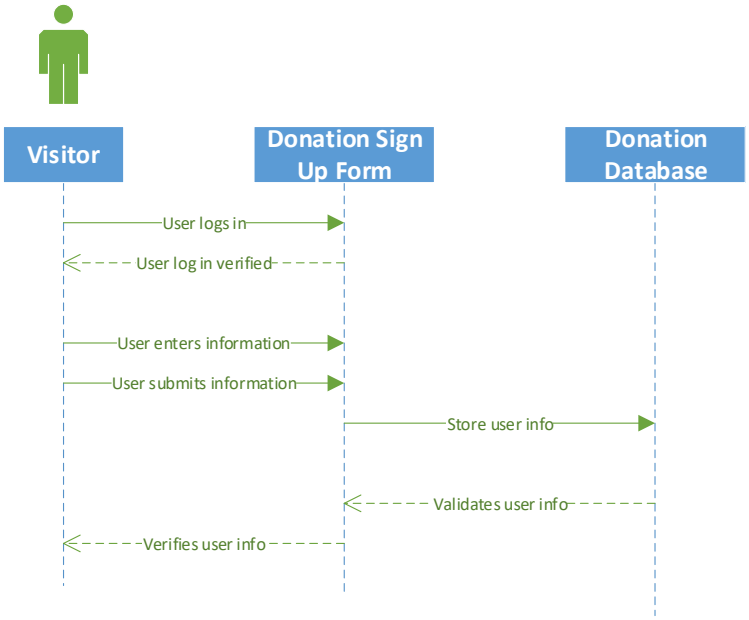
1. Admin navigates to HOOF website
2. Admin Logs in as a site Admin, execute Log In use case
3. Admin navigates to Administrator home page
4. Admin selects "Donations" page
5. Admin selects "Item Donations"
6. Admin selects "Accept" to accept an individual donation and keep the record permanently



UC3: SignUpToDonateGoods

Main Flow:

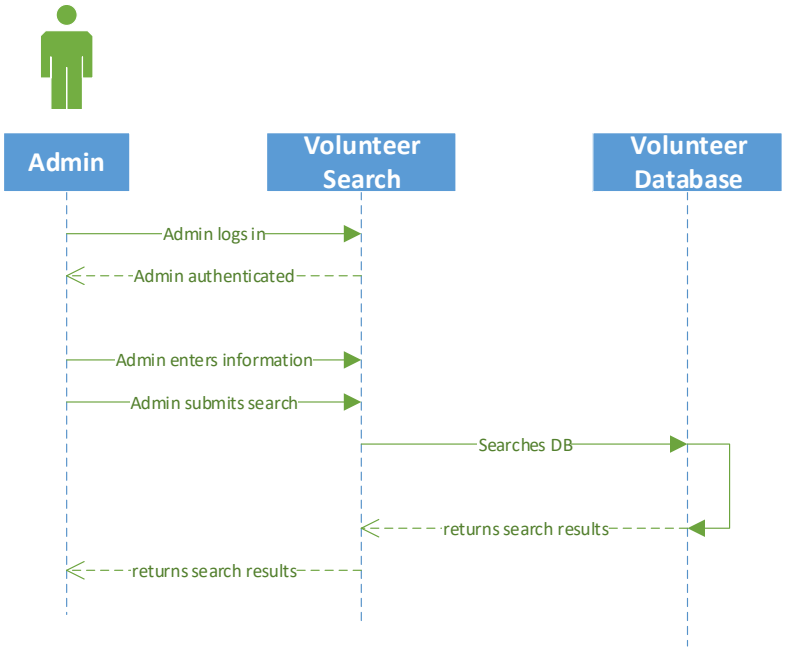
1. Visitor navigates to HOOF website
2. Visitor selects “Log In”, execute Log In use case
3. The member selects “Donate Goods” button
4. User selects from a drop down list of items to donate
5. User submits form by clicking the “Submit” button
6. Donation ID is automatically generated
7. System validates info
8. System verifies info
9. Use case ends



UC4: AccessVolunteerWorkInformation

Main Flow:

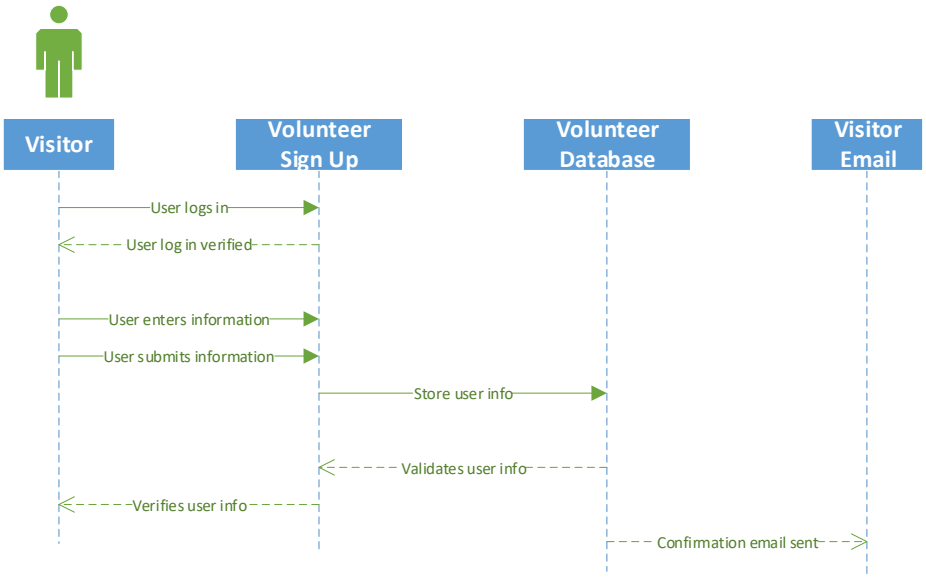
1. Admin navigates to HOOF website
2. Admin logs in as site Admin, execute Log In use case
3. Admin navigates to Administrator home page
4. Admin selects “Volunteer Records”
5. Admin enters volunteer ID, first name or last name into search field
6. Admin submits search
7. Admin selects correct result
8. End use case



UC5: SignUpForVolunteerWork

Main Flow:

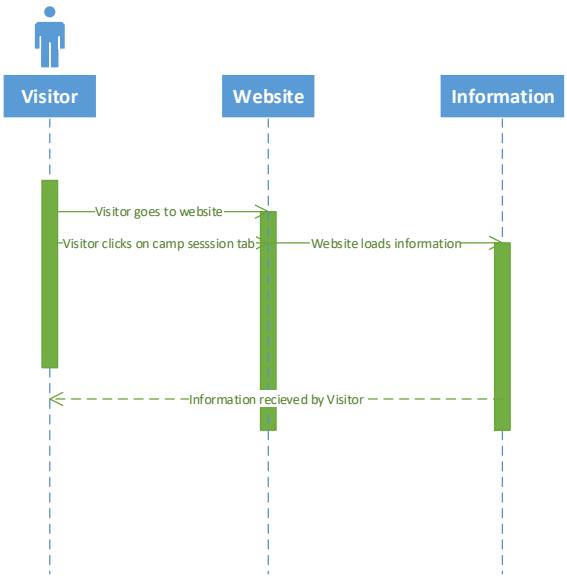
1. User navigates to HOOF website
2. User selects “Log In”, execute LogIn use case
3. The use case starts when the user selects the “Volunteer” button
4. User selects the camp they would like to volunteer at from a drop down list of current camps
5. User enters available times they can volunteer in the form
6. User submits form by clicking “Submit” button
7. Volunteer ID is automatically generated
8. Confirmation email is sent to user’s email address
9. Use case ends



UC6: Access HOOF camp session information

Flow of events:

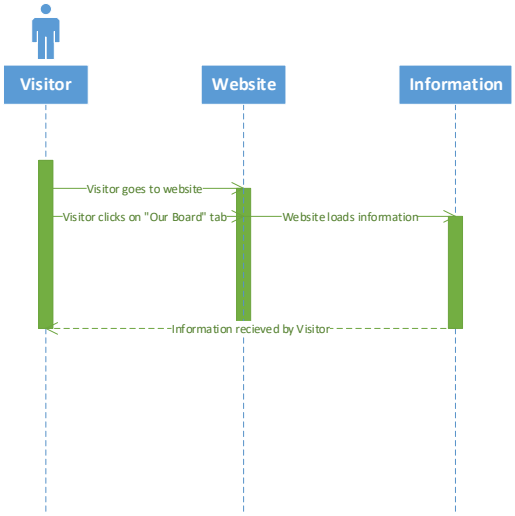
1. Visitor opens http://www.hoofky.org website
2. Visitor clicks on the tab labeled “Summer Camp”
3. Visitor gains information about HOOF KY camp sessions



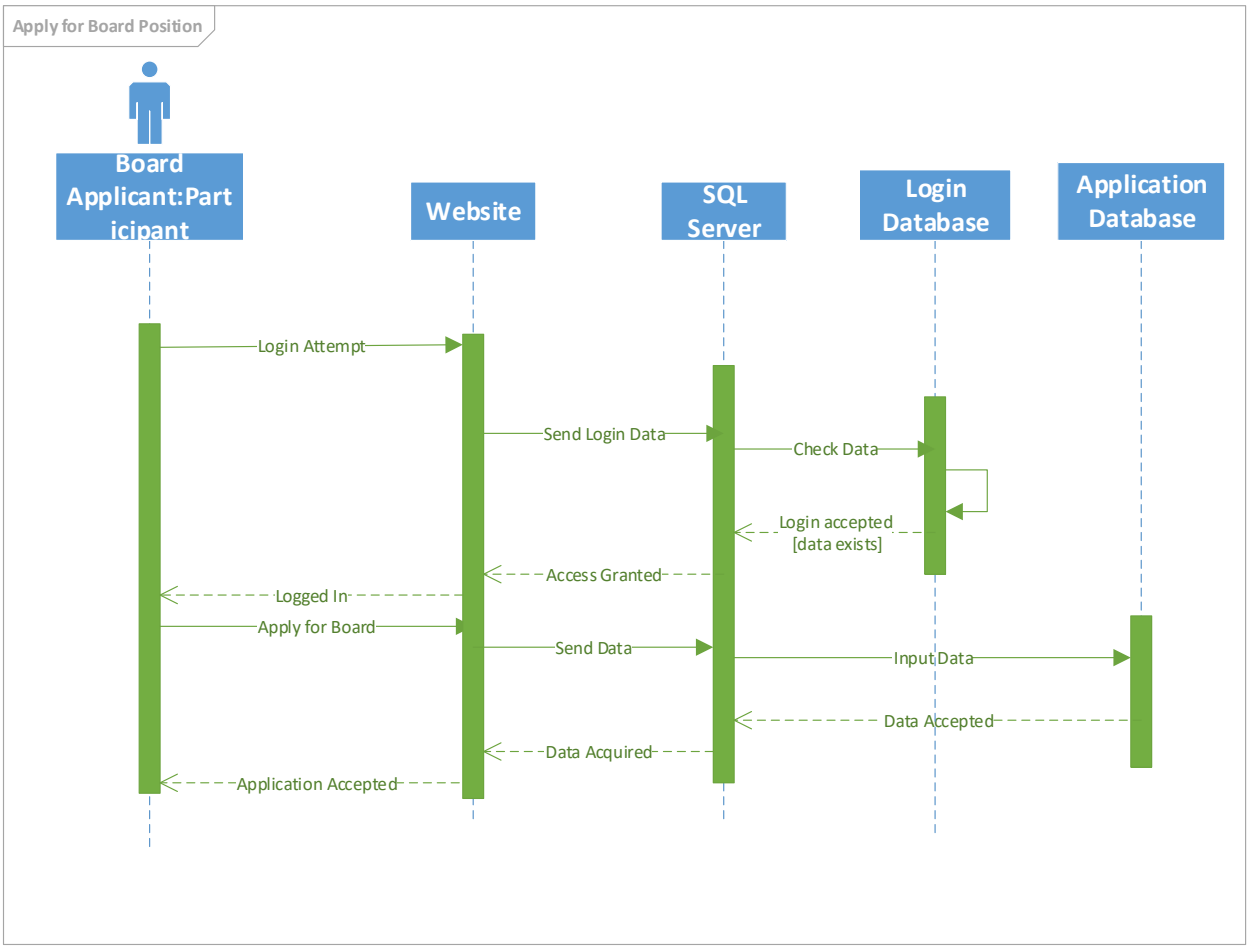
UC7: Access information about HOOF board members

Flow of events:

1. Visitor opens http://www.hoofky.org website
2. Visitor clicks on the tab labeled “Our Board”
3. Visitor has access to information about the HOOF KY Board of Directors

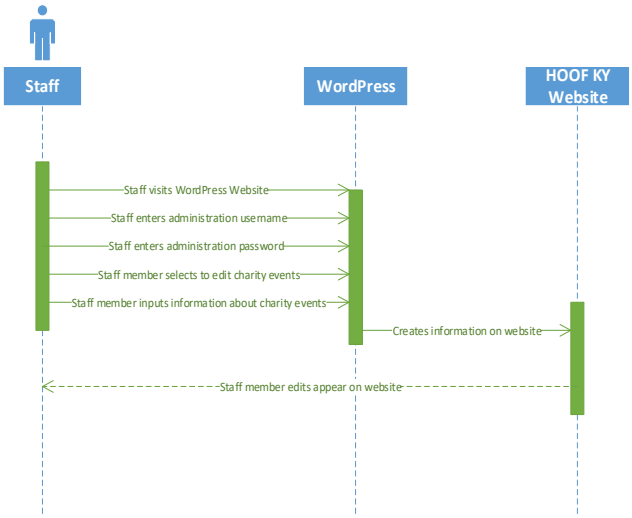


UC8: Apply for board position



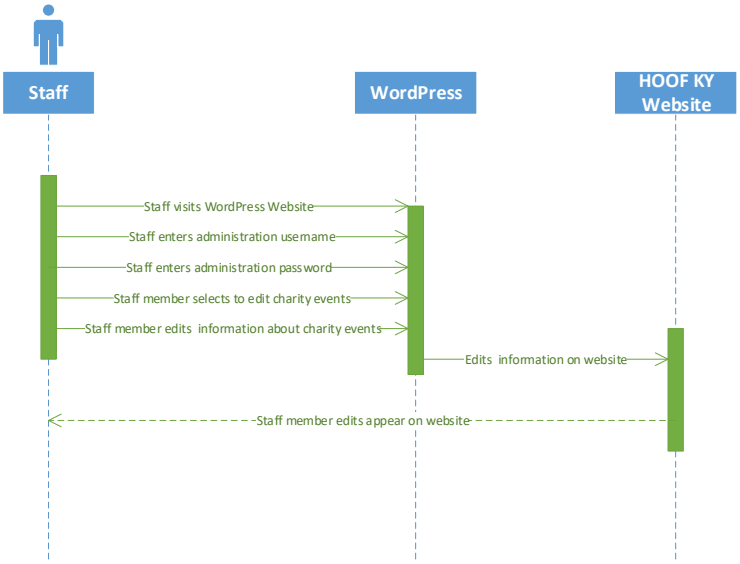
UC9: Create information about HOOF charity events

The staff member logins in to WordPress, edits the HOOF KY WordPress website and adds new information about charity events



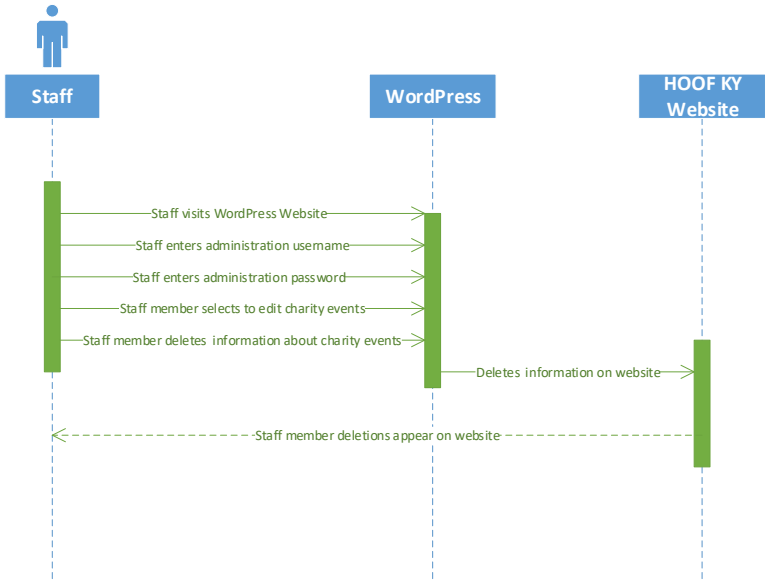
UC10: Modify information about HOOF charity events

The staff member logins in to WordPress, edits the HOOF KY WordPress website, and replaces old information with new information



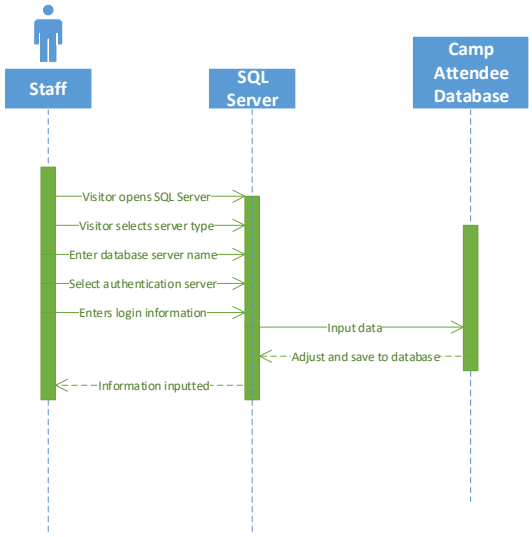
UC11: Delete information about HOOF charity events

The staff member logins in to WordPress, edits the HOOF KY WordPress website with deletions, and then the deletions appear on the website



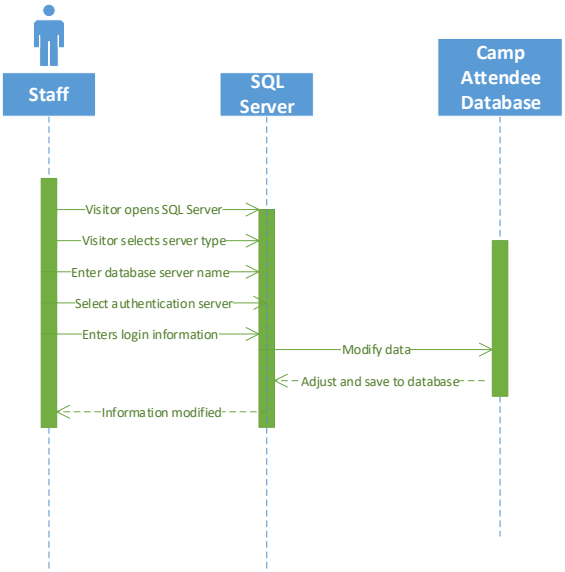
UC12: Input data on children at camps

The staff member signs in to SQL Server and connects to HOOF KY database, and inputs data into the child attendee database



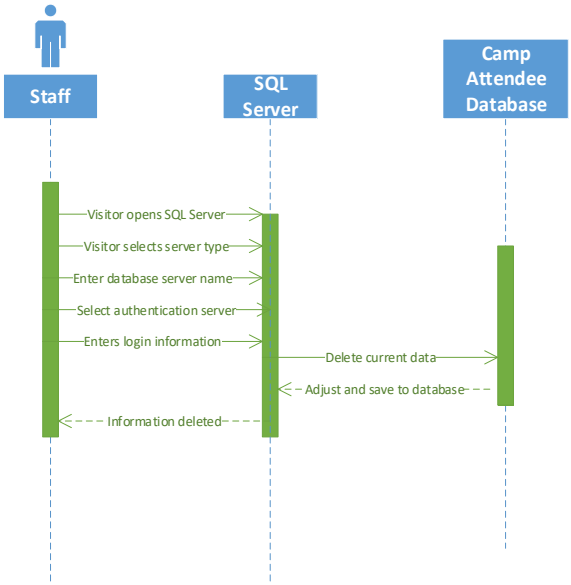
UC13: Modify data on children at camps

The staff member signs in to SQL Server and connects to HOOF KY database, and modifies data that is present within the database

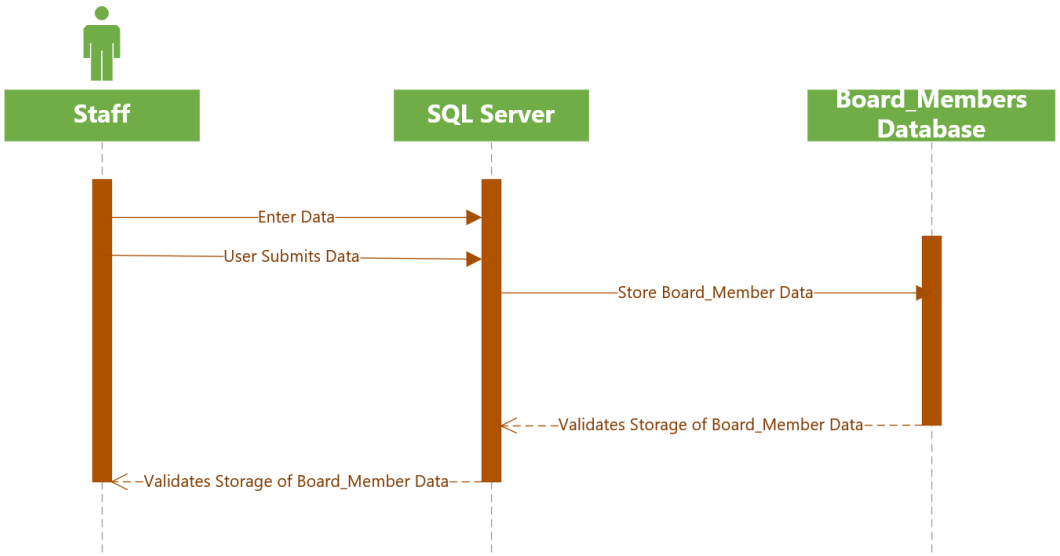


UC14: Delete data on children at camps

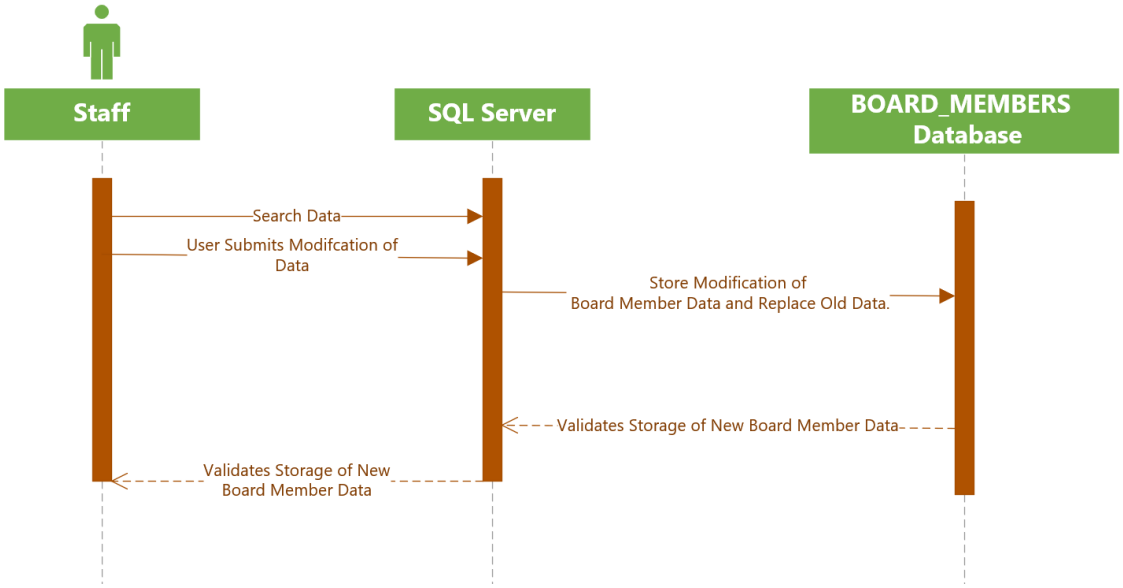
The staff member signs in to SQL Server and connects to HOOF KY database, and deletes data that is present within the database



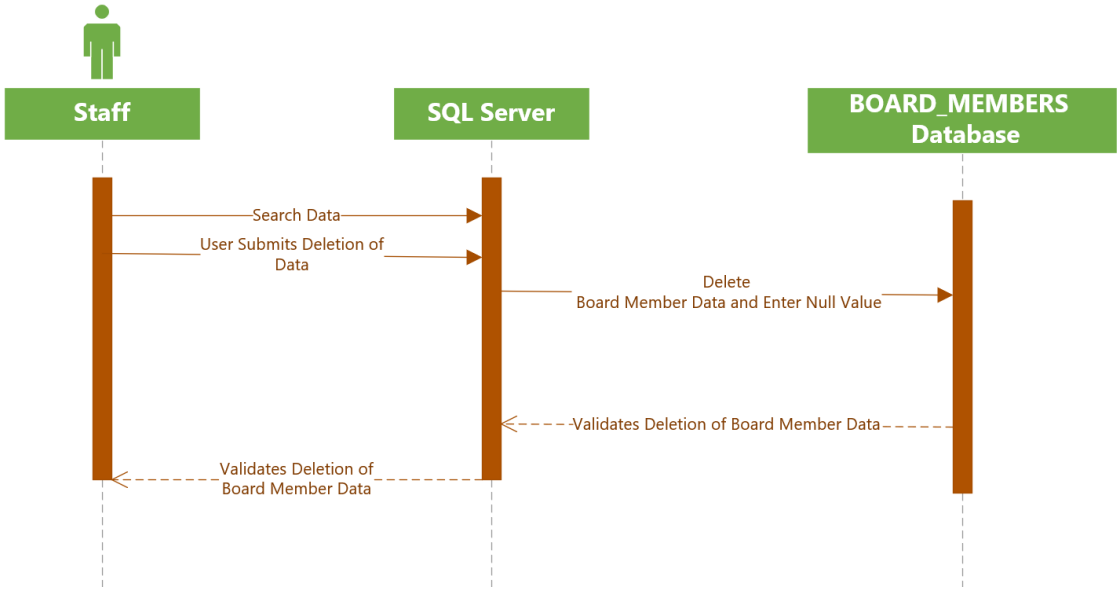
UC15: Input Board Member data



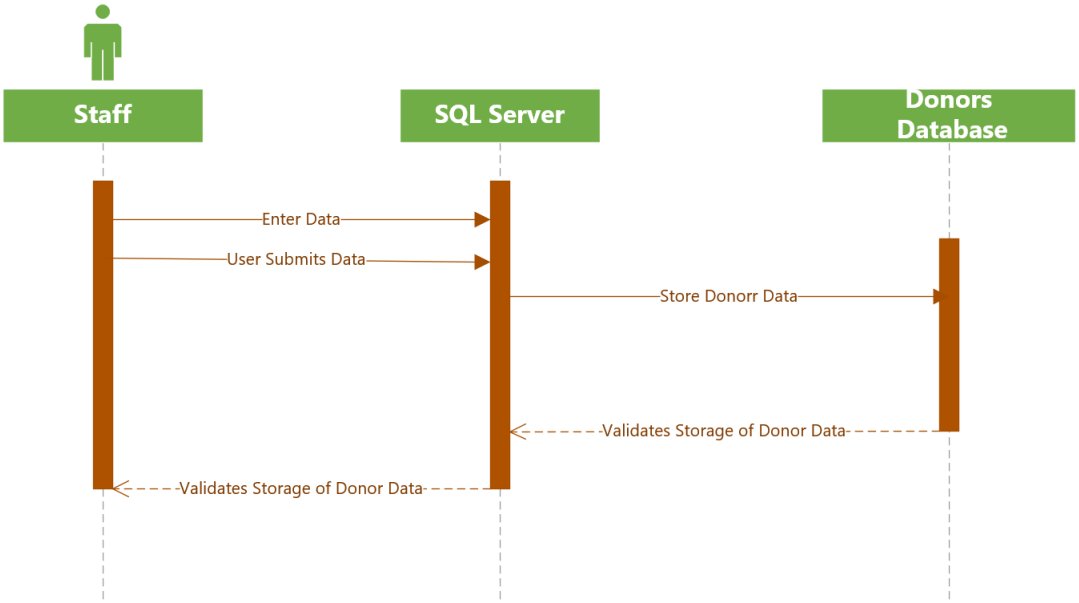
UC16: Modify Board Member data



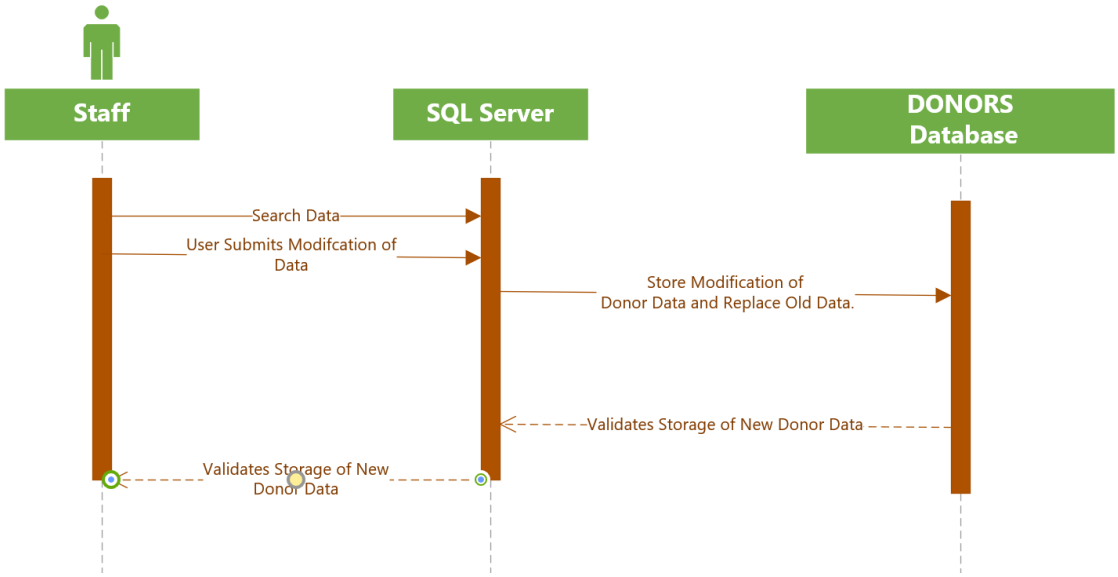
UC17: Delete Board Member data



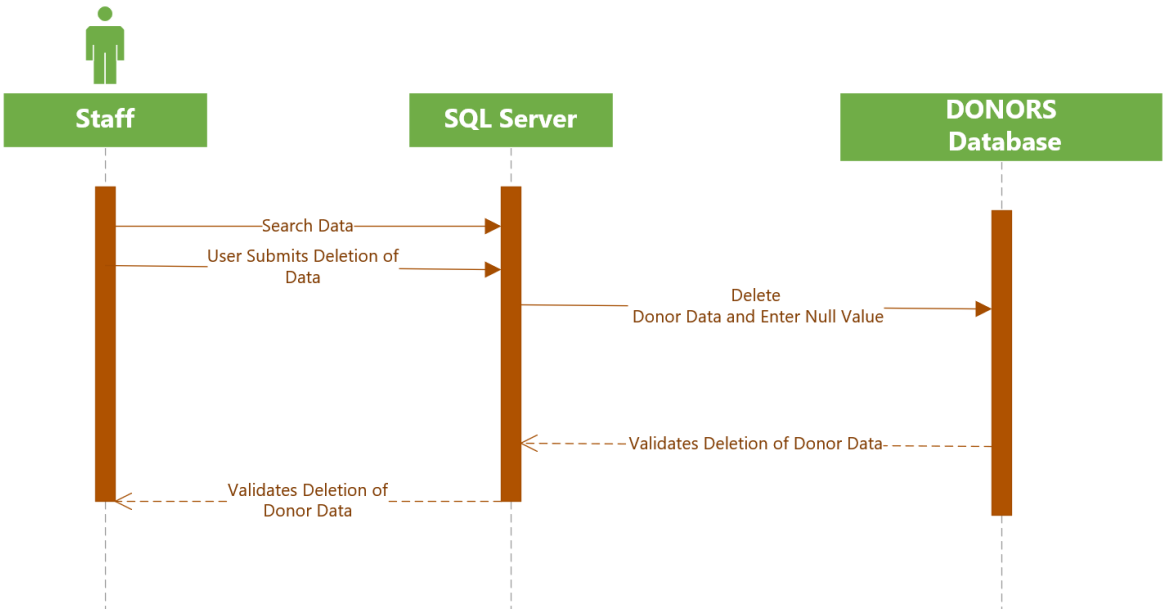
UC18: Input Donor Data



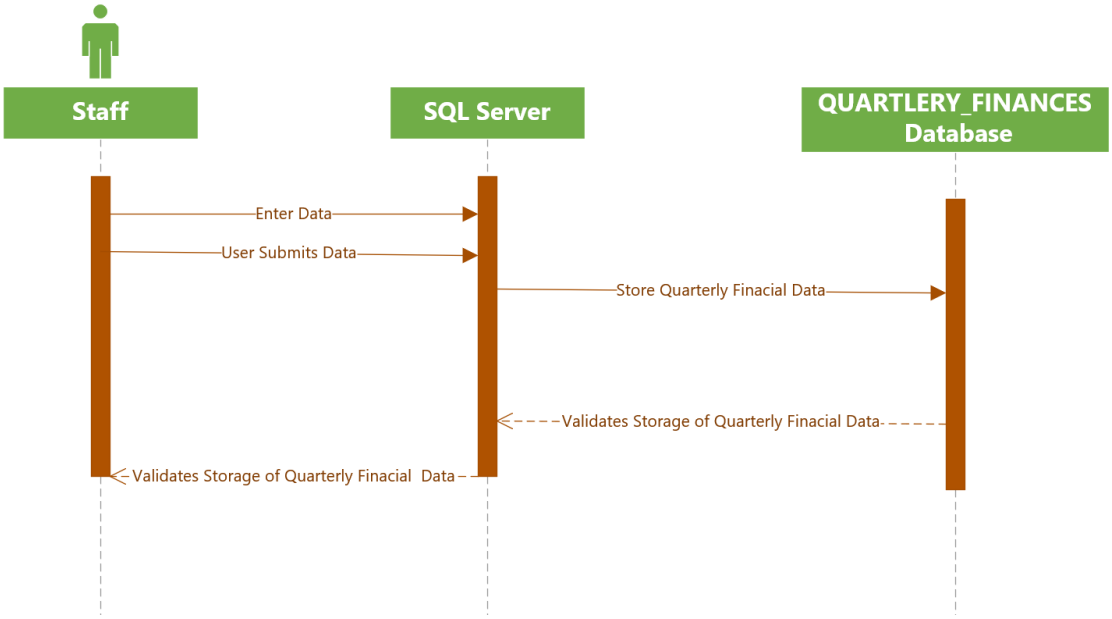
UC19: Modify Donor Data



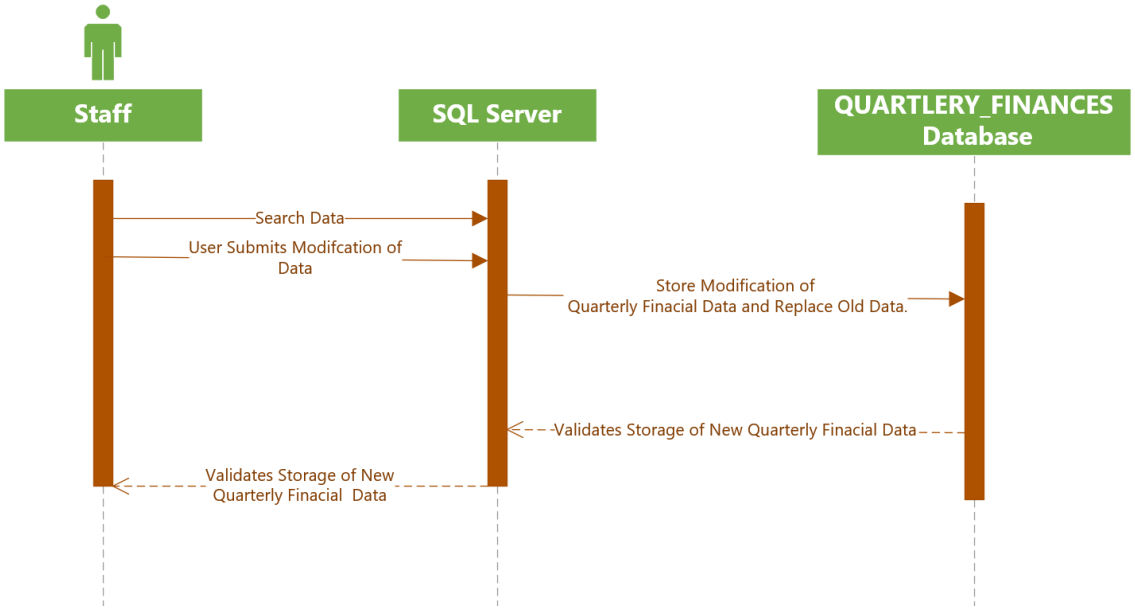
UC20: Delete Donor Data



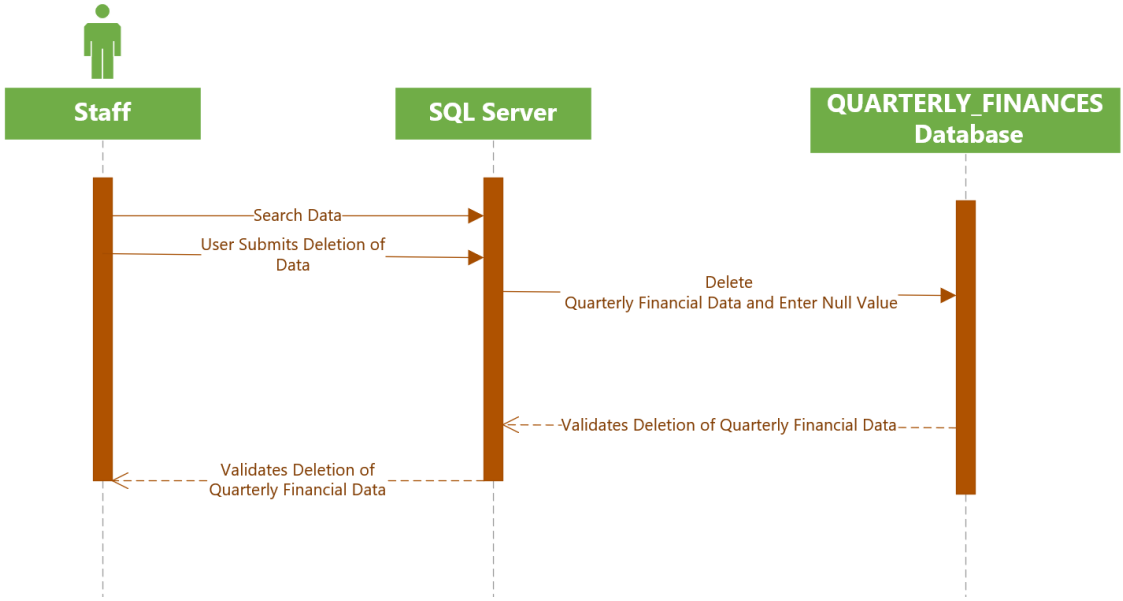
UC21: Input Financial Data



UC22: Modify Financial Data



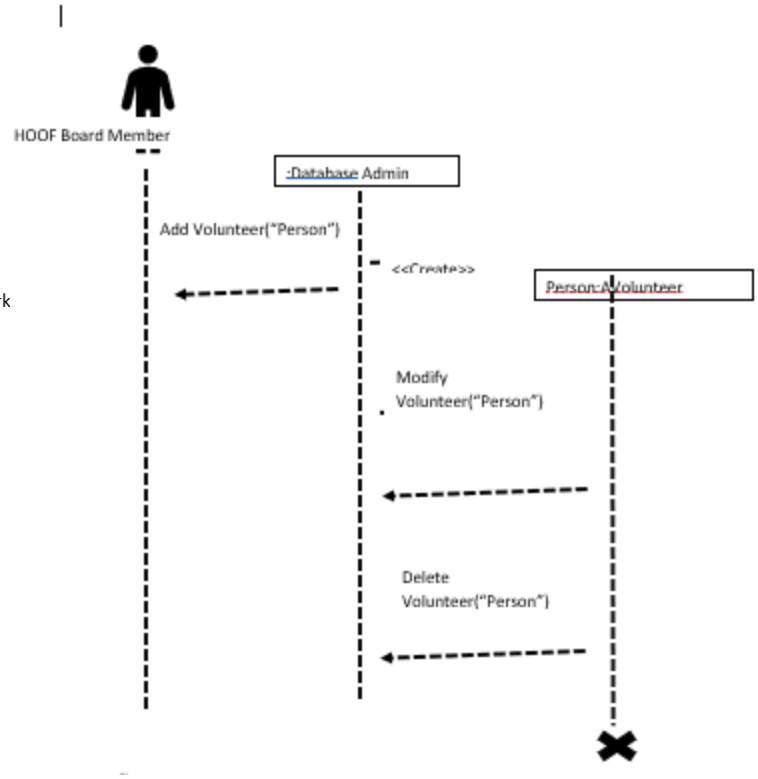
UC23: Delete Financial Data



UC24-26: Data on Volunteers

Main flow:

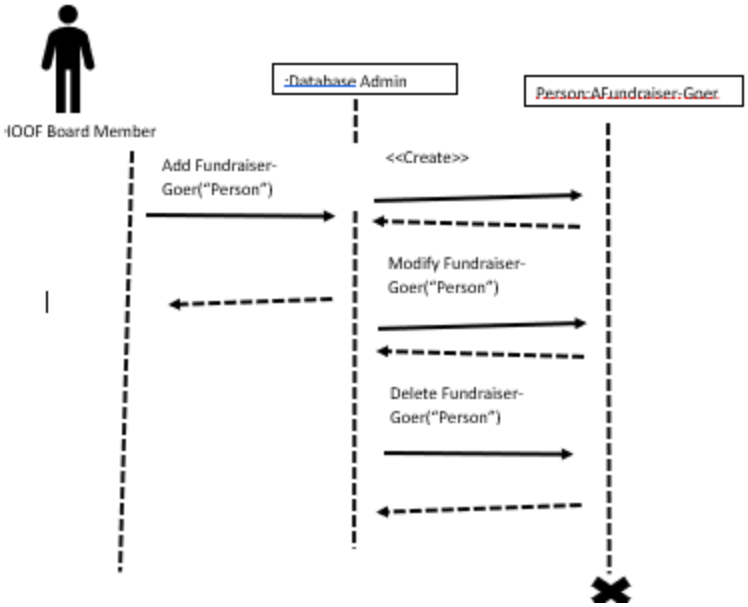
1. The Board member selects “add volunteer”.
2. The board member enters the first name of the volunteer.
3. The board member enters the last name of the volunteer.
4. The board member enters the address of the volunteer.
5. The board member enters the city of the volunteer.
6. The board member enters the state.
7. The board member enters the zip.
8. The board member enters the phone.
9. The board member enters the month, day, and year of the event work
10. The board member enters the name of the event worked.
11. The board member enters the hours worked.
12. The system creates a new volunteer.



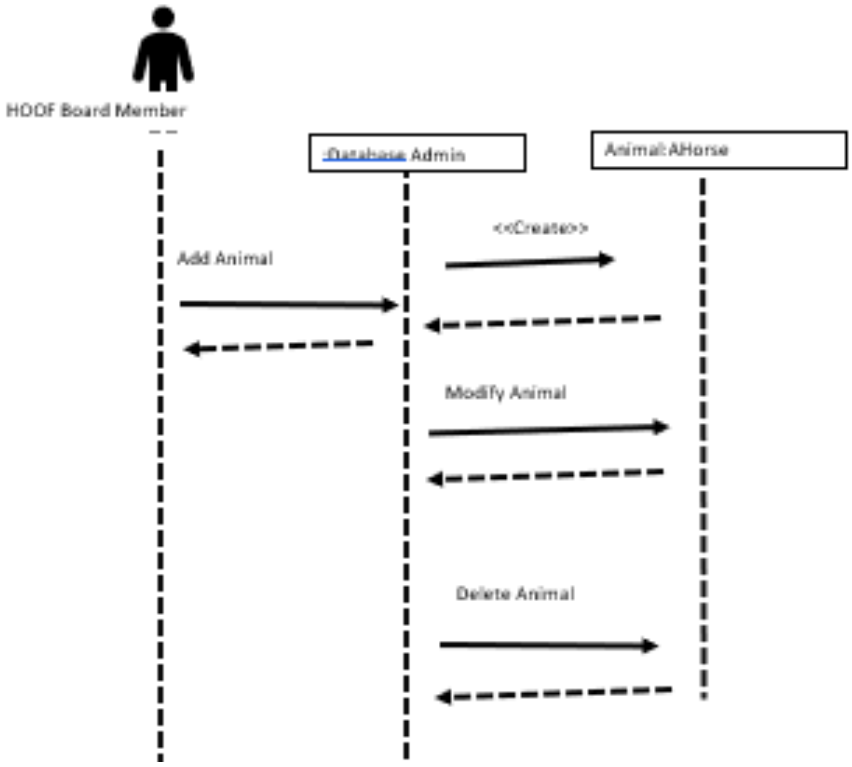
UC27-29: Fundraising Event-Goer Data

Main flow:

1. The Board member selects “add volunteer”.
2. The board member enters the first name of the fundraiser attendee.
3. The board member enters the last name of the fundraiser attendee.
4. The board member enters the address of the fundraiser attendee.
5. The board member enters the city.
6. The board member enters the state.
7. The board member enters the zip.
8. The board member enters the month, day, and year of the event attended.
9. The board member enters the name of the event attended.
10. The system creates a new fundraiser attendee.



UC30-32: Animal Data



Main flow:

1. The Board member selects “add animal”.
2. The board member enters the name of the animal
3. The board member enters the first name of the owner of the animal.
4. The board member enters the last name of the owner of the animal.
5. The board member enters the address of the owner of the animal.
6. The board member enters the city of the owner of the animal.
7. The board member enters the state of the owner of the animal.
8. The board member enters the zip of the owner of the animal.
9. The board member enters the phone of the owner of the animal.
10. The board member enters the condition of the animal (ridable: Y/N).
11. The board member enters any veterinary work on the animal.
12. The board member enters any medication for the animal.
13. The system creates a new animal

Use case: Modify Animal

Main flow:

1.The Board member selects “modify animal”.

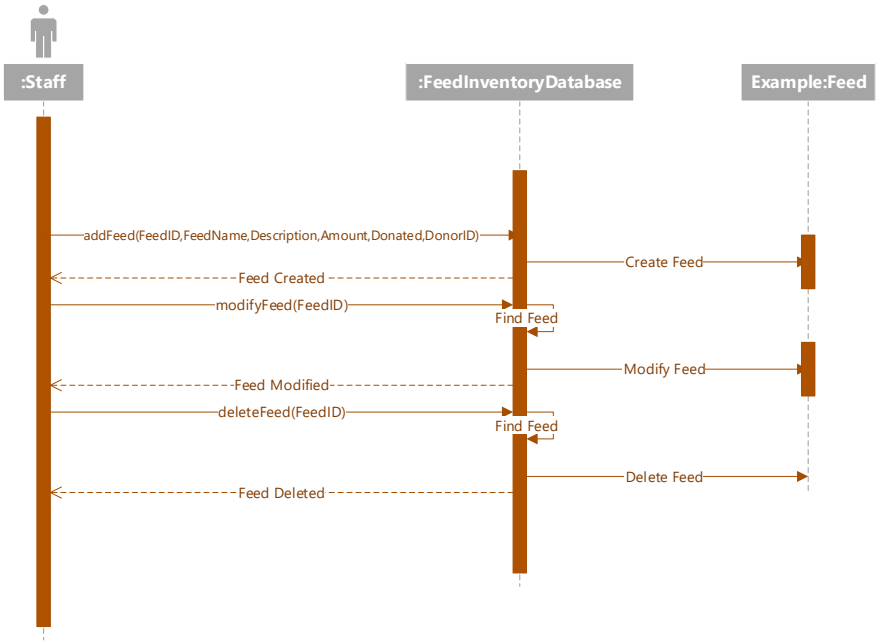
2. The board member selects which animal to modify.
3. The board member selects which attribute to modify.
4. The system edits the animal

Use case: Delete Animal

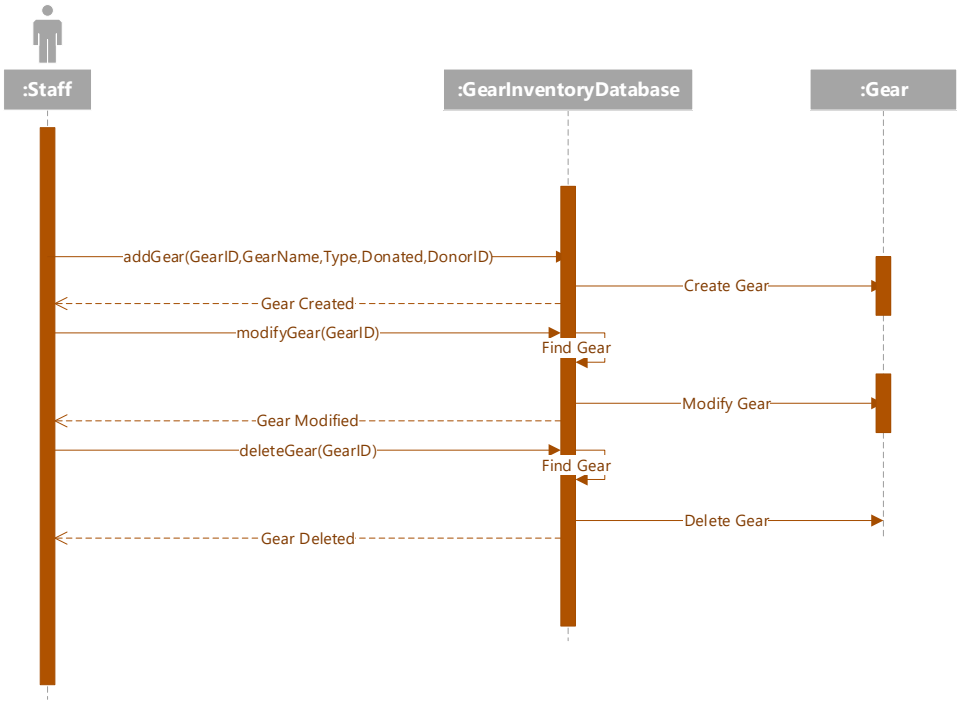
Main flow:

1. The Board member selects “delete animal”.
2. The board member selects which animal to delete.
3. The system deletes the animal.

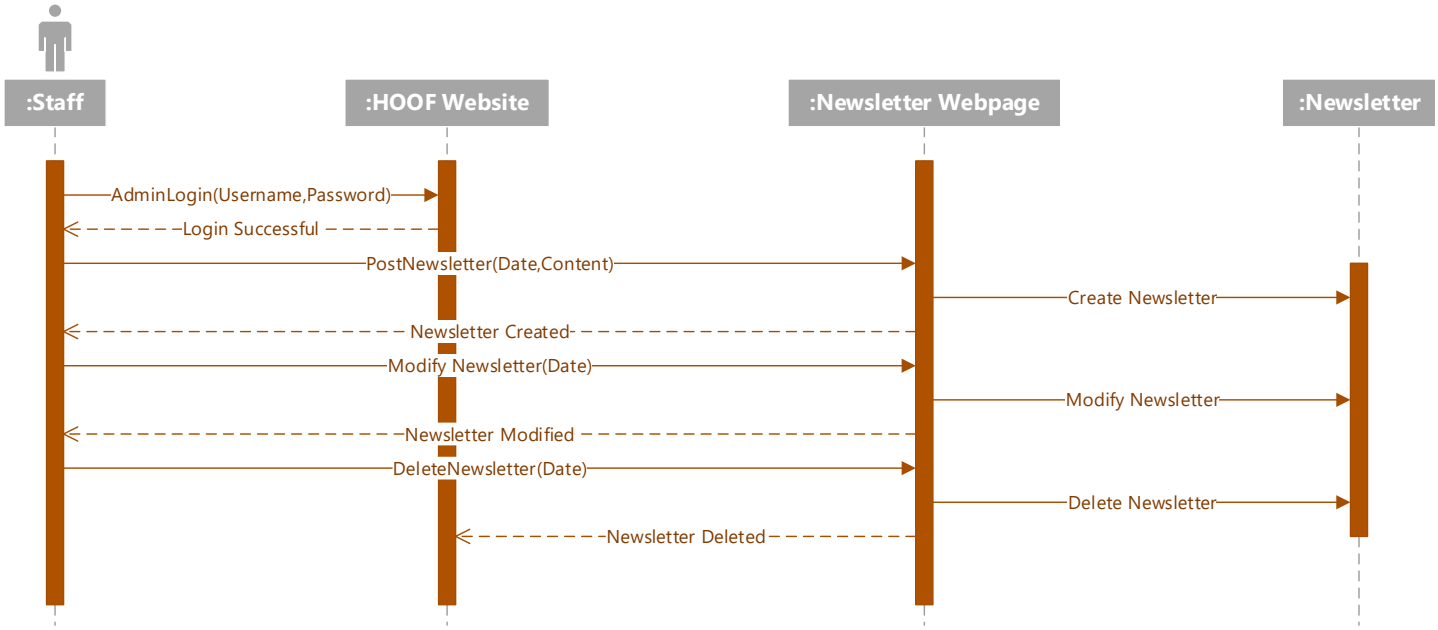
UC33-35: Post, Modify, Delete Feed Inventory



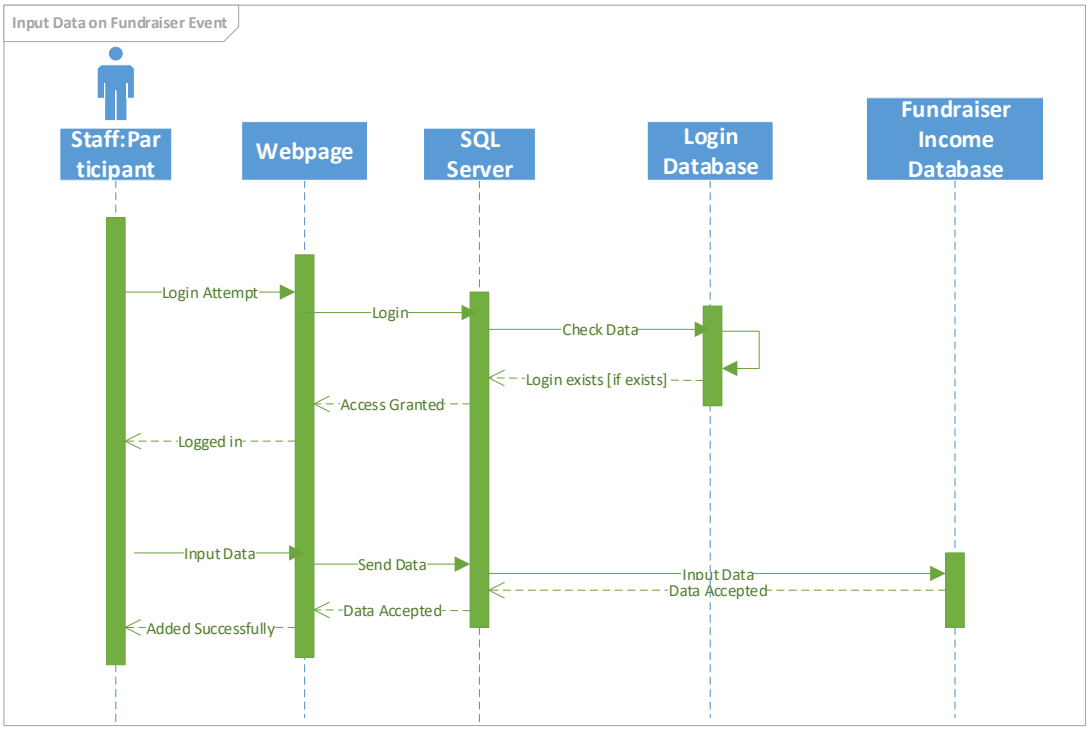
UC36-38: Post, Modify, Delete Gear Inventory



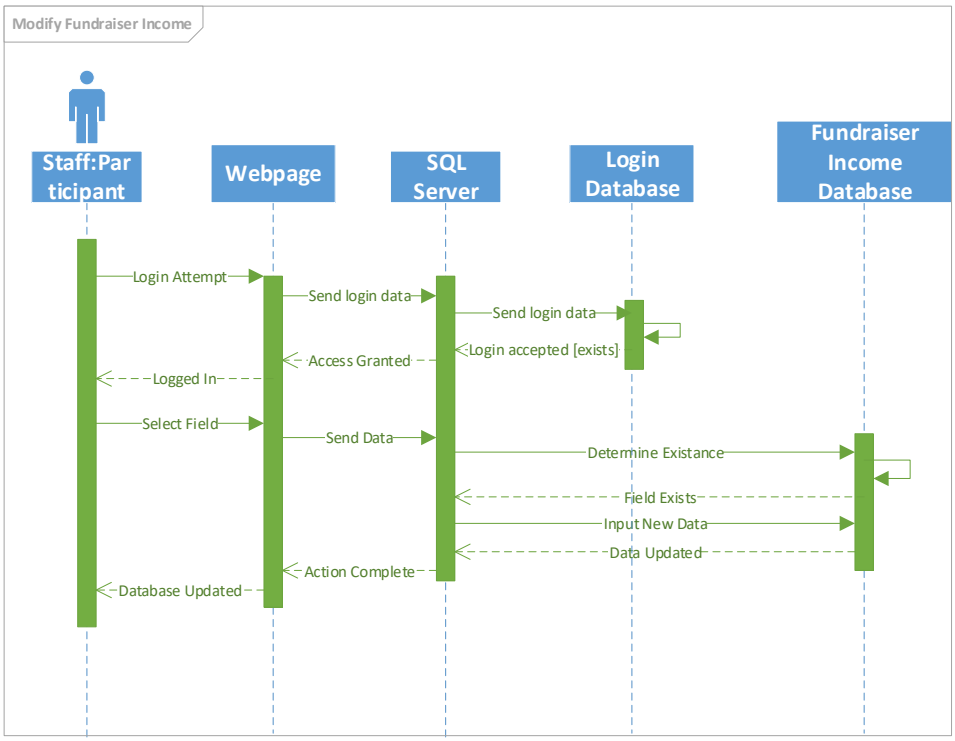
UC39-41 Post, Modify, Delete Newsletter



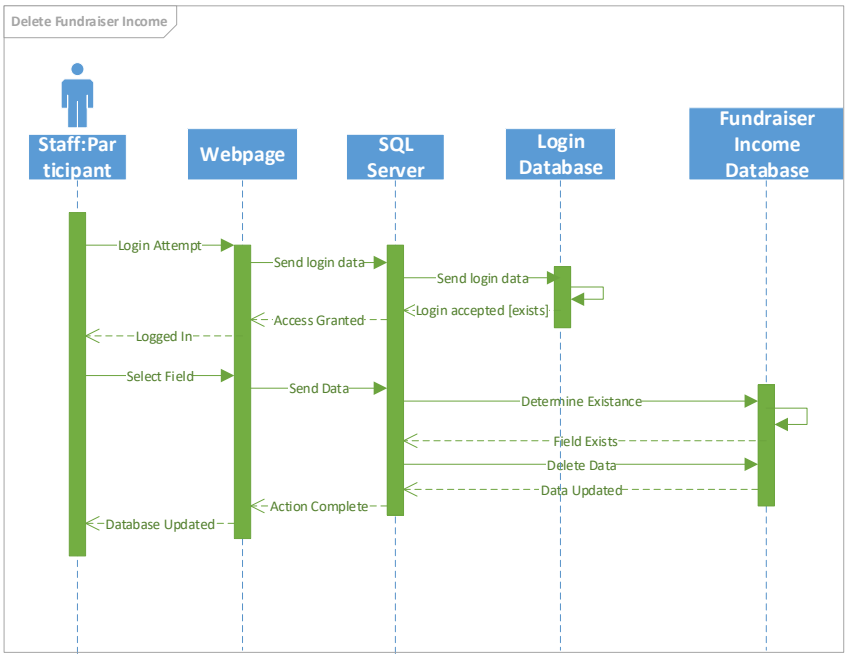
UC42: Input data on fundraiser income



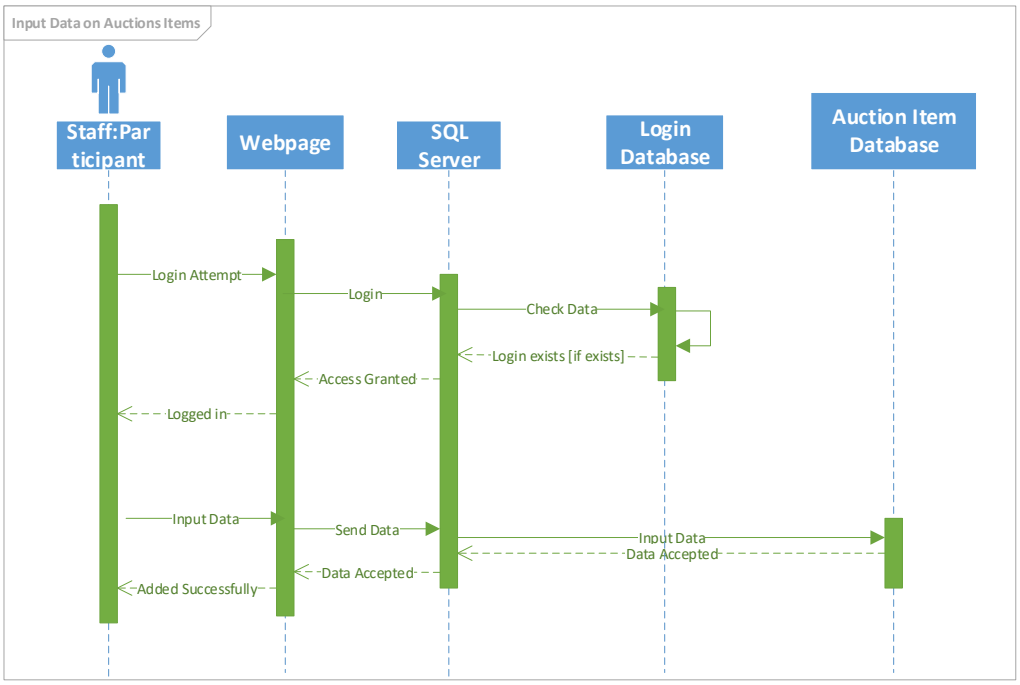
UC43: Modify data on fundraiser income



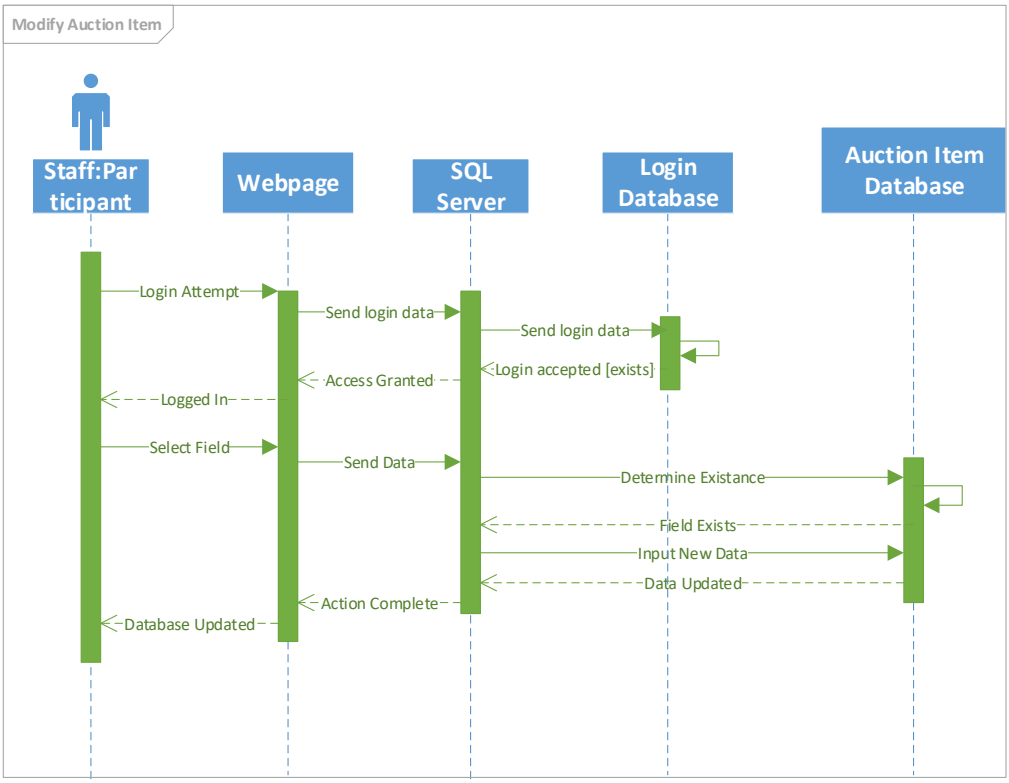
UC44: Delete data on fundraiser income



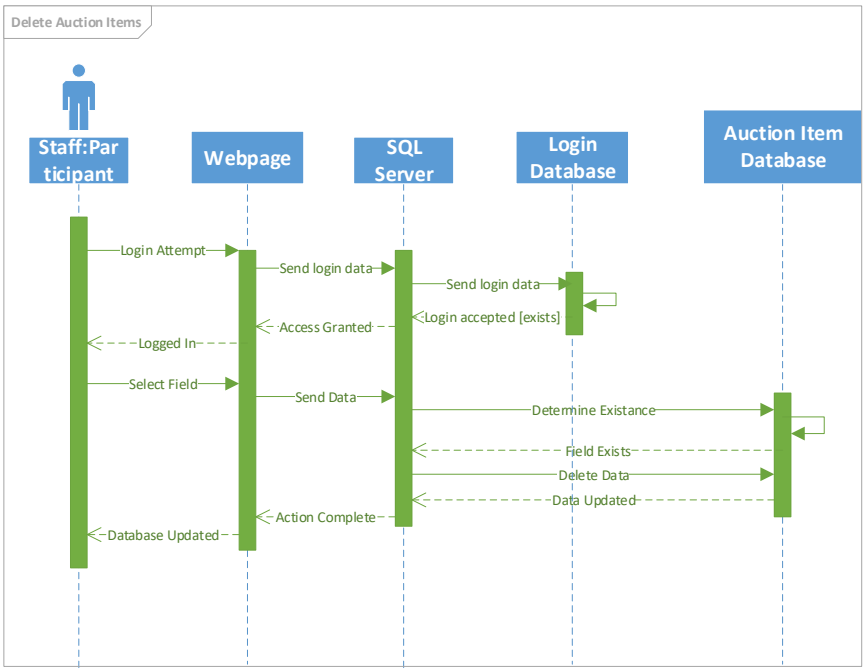
UC45: Input Data on Auction Items



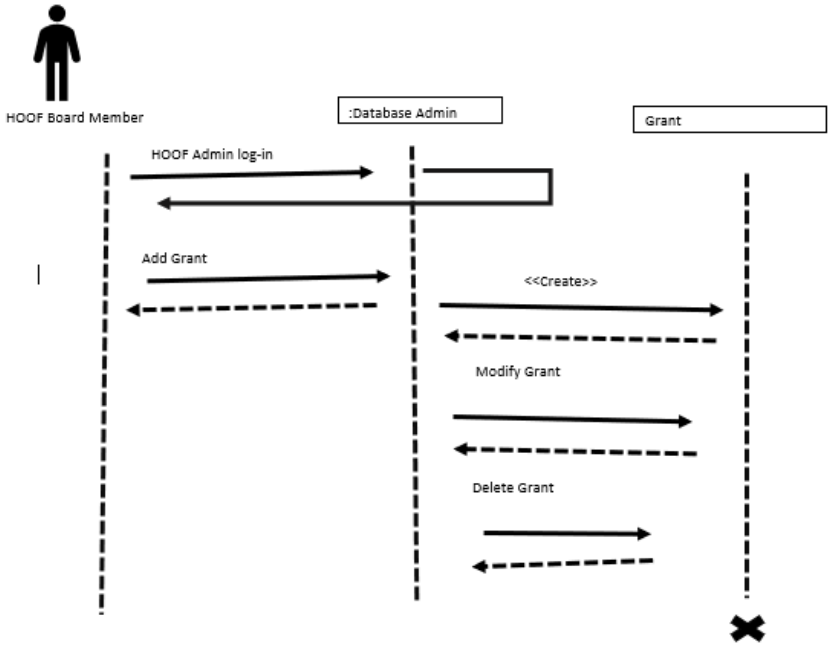
UC46: Modify Data on Auction Items



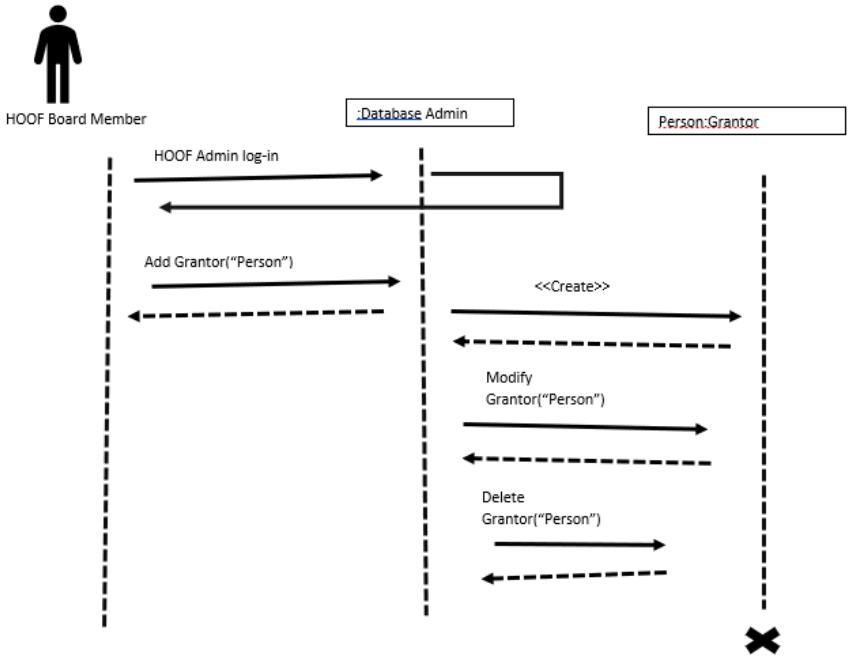
UC47: Delete Data on Auction Items



UC48-50: Input, Modify and Delete data on Grants



UC51-53: Input, Modify and Delete data on Grantors



UC 54: Member Login Data

Main flow:

1.The user selects “Create an account”

2.The user creates a login user name

3.User creates password and confirms the password

4.The user selects “Member”

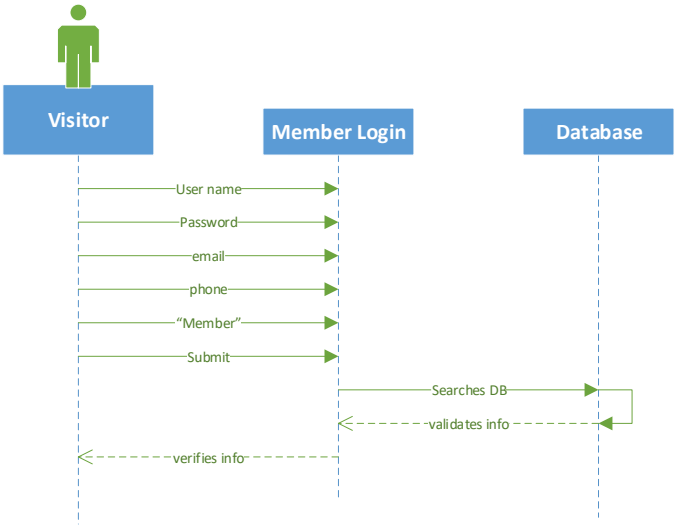
5.The user fills out first and last name

6.The user inputs their phone number

7.The user inputs their email

8.The user selects whether they’d like to receive a newsletter

9. User is automatically generated a user ID



UC55: Admin Login

Main flow:

The user selects “Create an account”

2.The user creates a login user name

3.User creates password and confirms the password

4.The user selects “Admin”

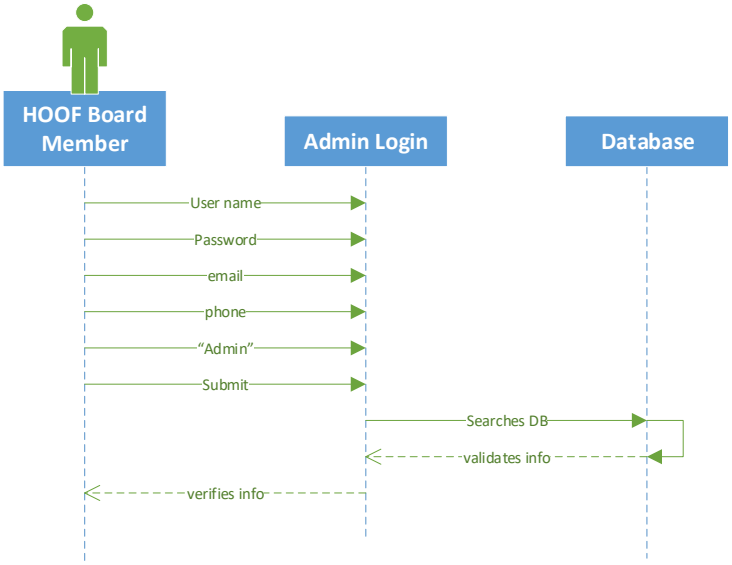
5.The user fills out first and last name

6.The user inputs their phone number

7.The user inputs their email

8.The user receives a message that they will be contacted when their admin status has been approved

9. User is automatically generated a user ID

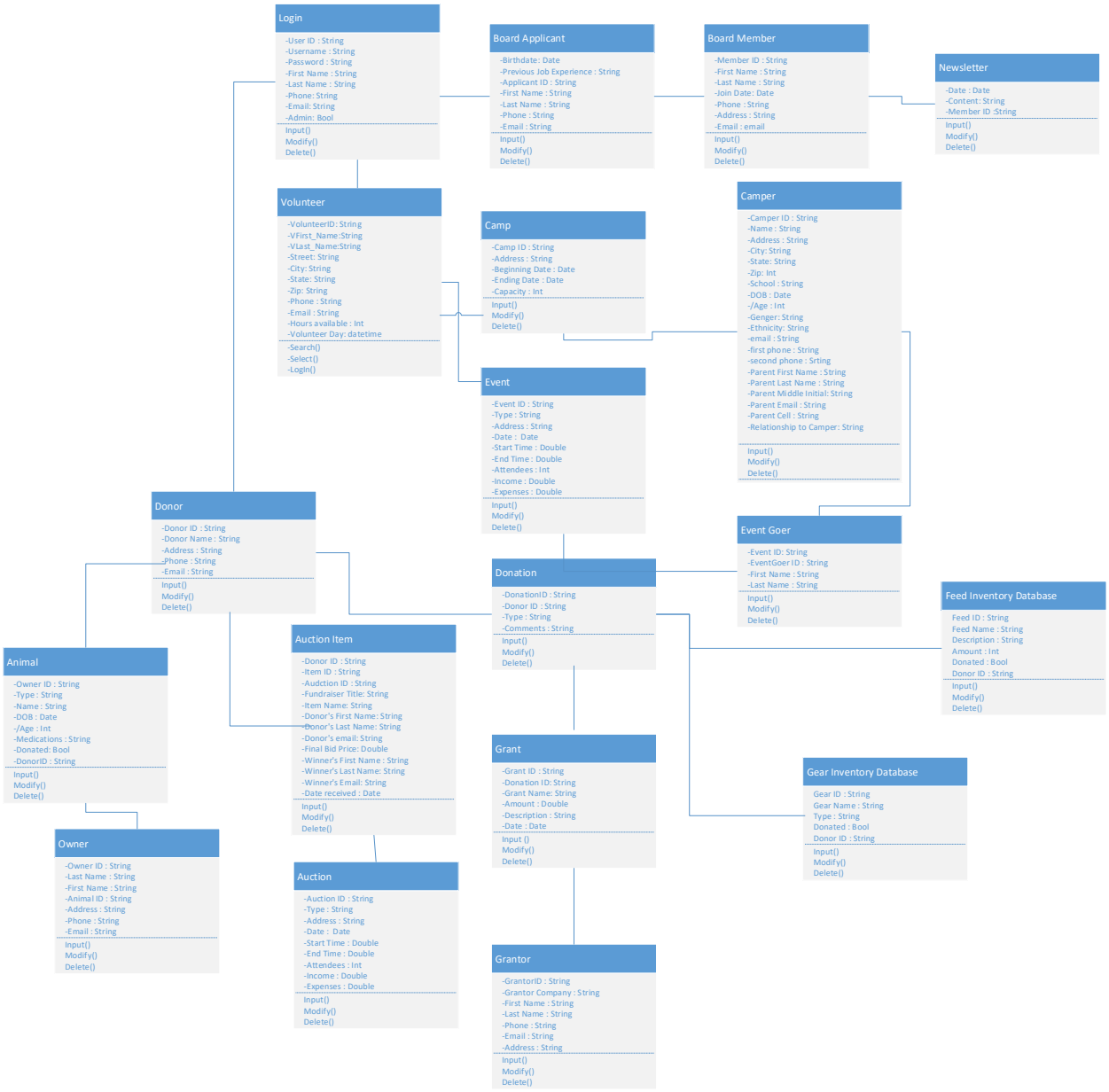


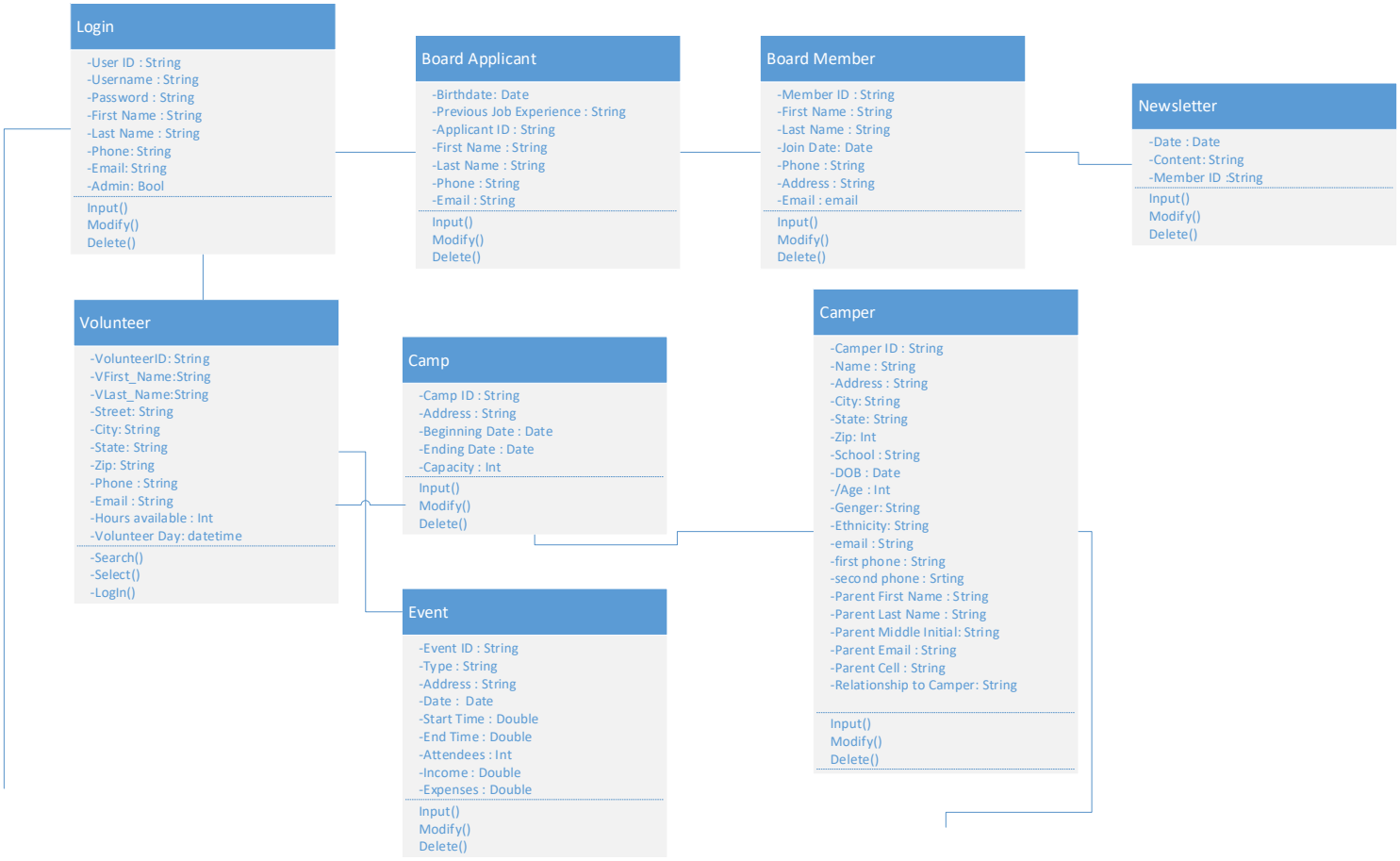
Class Diagrams

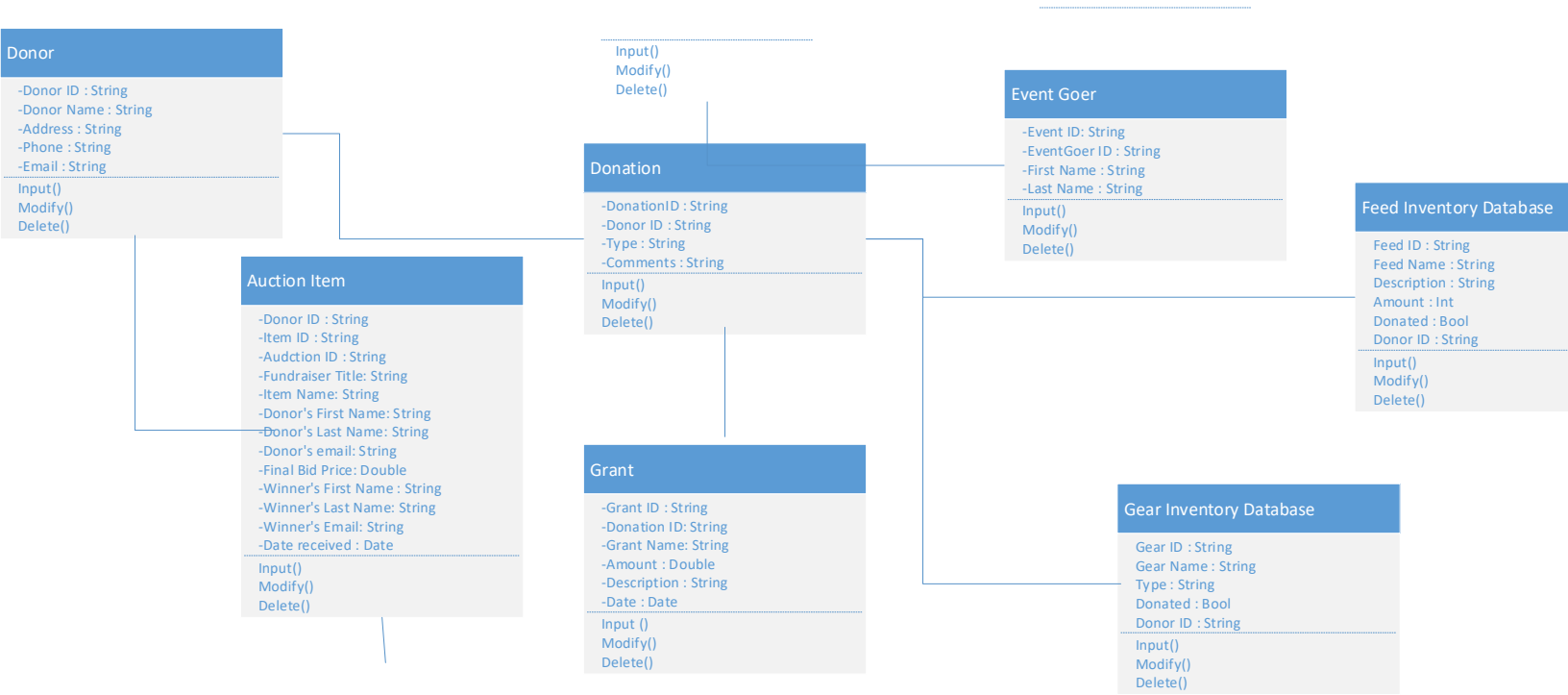
Class Diagram

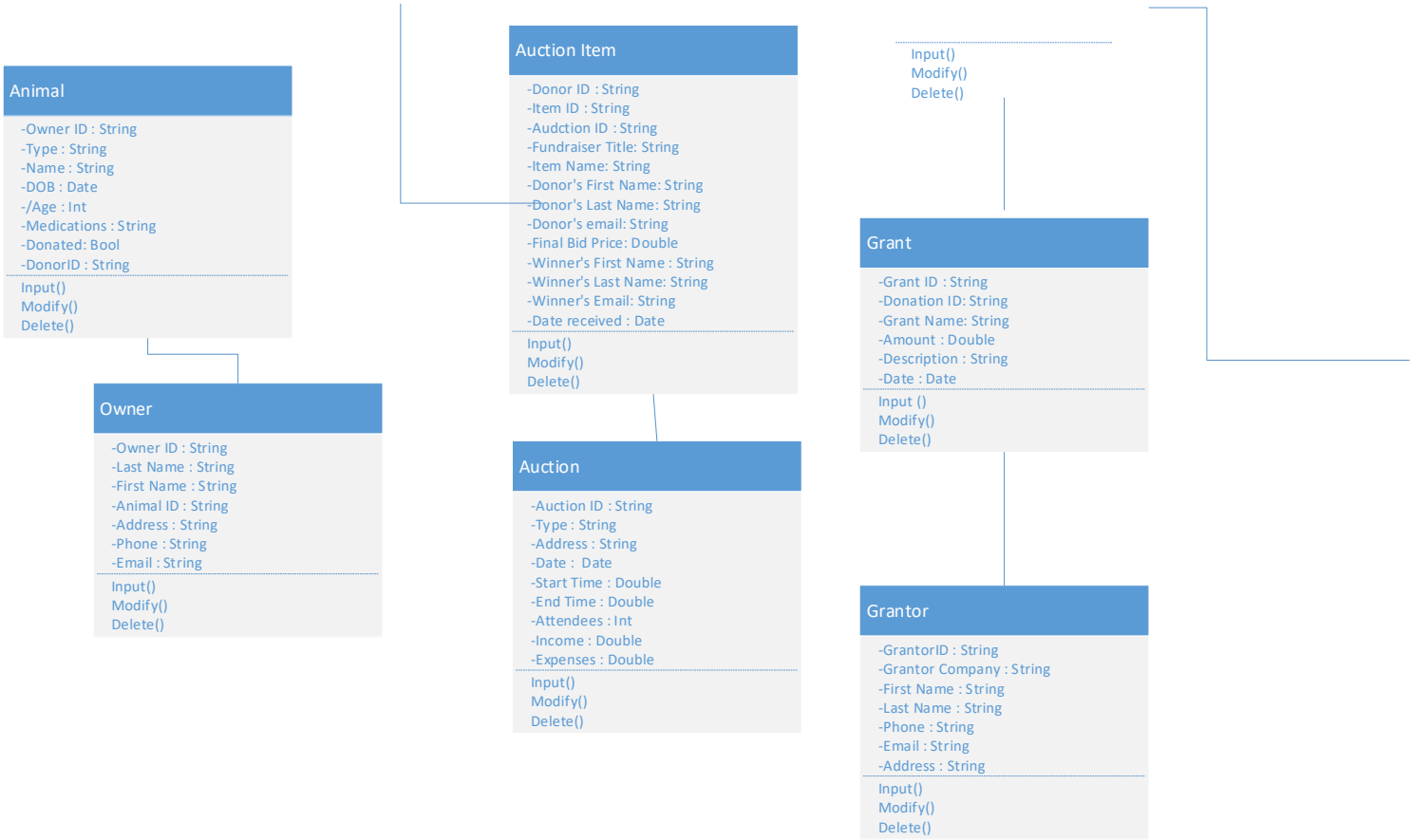
This diagram visualizes the relationships between system objects. Each object presented here has a name and performs a specific function on the website. Each object is connected to another object using a line.

Presented here is a general overview of the whole class diagram followed by three pictures that zoom in on the individual tables.




Login
-User ID : String
-Username : String
-Password : String
-First Name : String
-Last Name : String
-Phone: String
-Email: String
-Admin: Bool
Input()
Modify()
Delete()
Board Applicant
-Birthdate: Date
-Previous Job Experience : String
-Applicant ID : String
-First Name : String
-Last Name : String
-Phone : String
-Email : String
Input()
Modify()
Delete()
Board Member
-Member ID : String
-First Name : String
-Last Name : String
-Join Date: Date
-Phone : String
-Address : String
-Email : email
Input()
Modify()
Delete()
Newsletter
-Date : Date
-Content: String
-Member ID :String
Input()
Modify()
Delete()
Volunteer
-VolunteerID: String
-VFirst_Name:String
-VLast_Name:String
-Street: String
-City: String
-State: String
-Zip: String
-Phone : String
-Email : String
-Hours available : Int
-Volunteer Day: datetime
-Search()
-Select()
-LogIn()
Camp
-Camp ID : String
-Address : String
-Beginning Date : Date
-Ending Date : Date
-Capacity : Int
Input()
Modify()
Delete()
Camper
-Camper ID : String
-Name : String
-Address : String
-City: String
-State: String
-Zip: Int
-School : String
-DOB : Date
-/Age : Int
-Genger: String
-Ethnicity: String
-email : String
-first phone : String
-second phone : Srting
-Parent First Name : String
-Parent Last Name : String
-Parent Middle Initial: String
-Parent Email : String
-Parent Cell : String
-Relationship to Camper: String
Input()
Modify()
Delete()
Event
-Event ID : String
-Type : String
-Address : String
-Date : Date
-Start Time : Double
-End Time : Double
-Attendees : Int
-Income : Double
-Expenses : Double
Input()
Modify()
Delete()

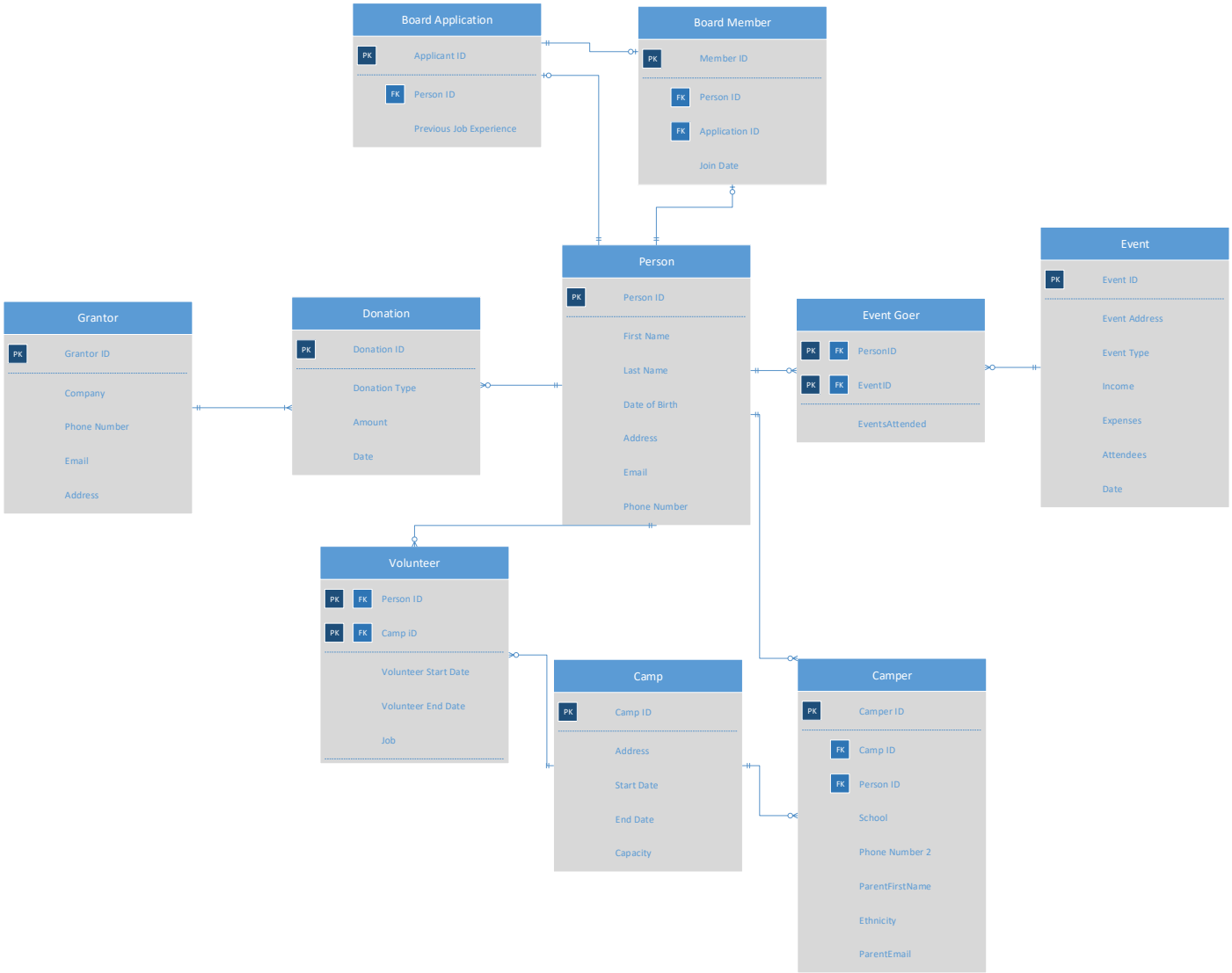

Donor
-Donor ID : String
-Donor Name : String
-Address : String
-Phone : String
-Email : String
Input()
Modify()
Delete()
Input()
Modify()
Delete()
Donation
-DonationID : String
-Donor ID : String
-Type : String
-Comments : String
Input()
Modify()
Delete()
Event Goer
-Event ID: String
-EventGoer ID : String
-First Name : String
-Last Name : String
Input()
Modify()
Delete()
Feed Inventory Database
Feed ID : String
Feed Name : String
Description : String
Amount : Int
Donated : Bool
Donor ID : String
Input()
Modify()
Delete()
Auction Item
-Donor ID : String
-Item ID : String
-Audction ID : String
-Fundraiser Title: String
-Item Name: String
-Donor's First Name: String
-Donor's Last Name: String
-Donor's email: String
-Final Bid Price: Double
-Winner's First Name : String
-Winner's Last Name: String
-Winner's Email: String
-Date received : Date
Input()
Modify()
Delete()
Grant
-Grant ID : String
-Donation ID: String
-Grant Name: String
-Amount : Double
-Description : String
-Date : Date
Input ()
Modify()
Delete()
Gear Inventory Database
Gear ID : String
Gear Name : String
Type : String
Donated : Bool
Donor ID : String
Input()
Modify()
Delete()


Animal
-Owner ID : String
-Type : String
-Name : String
-DOB : Date
-/Age : Int
-Medications : String
-Donated: Bool
-DonorID : String
Input()
Modify()
Delete()
Owner
-Owner ID : String
-Last Name : String
-First Name : String
-Animal ID : String
-Address : String
-Phone : String
-Email : String
Input()
Modify()
Delete()
Auction Item
-Donor ID : String
-Item ID : String
-Audction ID : String
-Fundraiser Title: String
-Item Name: String
-Donor's First Name: String
-Donor's Last Name: String
-Donor's email: String
-Final Bid Price: Double
-Winner's First Name : String
-Winner's Last Name: String
-Winner's Email: String
-Date received : Date
Input()
Modify()
Delete()
Auction
-Auction ID : String
-Type : String
-Address : String
-Date : Date
-Start Time : Double
-End Time : Double
-Attendees : Int
-Income : Double
-Expenses : Double
Input()
Modify()
Delete()
Input()
Modify()
Delete()
Grant
-Grant ID : String
-Donation ID: String
-Grant Name: String
-Amount : Double
-Description : String
-Date : Date
Input ()
Modify()
Delete()
Grantor
-GrantorID : String
-Grantor Company : String
-First Name : String
-Last Name : String
-Phone : String
-Email : String
-Address : String
Input()
Modify()
Delete()

Database Design

Database Design

The database design is created using an entity relationship diagram (ERD). This ERD is a normalized version of the class. This is done so that data redundancy are minimized and cardinalities are used to define table relationships. This illustration includes how tables will interact with one another and lists fields that will be included within each object.



Data Dictionary

Data Dictionary

The table below can be used to define each field within the database.

Entity	Field Name	Data Type	Field Length	Constraint	Description
Login User	User ID	Int	4	Primary Key	User ID, auto generated
	Username	String	25		A username used to validate user
	Password	String	25		A password used to validate user
	First_Name	String	25		First name of user logging in
	Last_Name	string	25		Last name of user logging in

Entity	Field Name	Data Type	Field Length	Constraint	Description
Board Applicant	ApplicantID	Int	4	Primary Key	Applicant ID, auto generated
	A.First_Name	String	25		First name of applicant
	A.Last_Name	String	25		Last Name of applicant
	Birthdate	Date	8		Birthdate, used to determine age of applicant
	A.PhoneNumber	Int	10		Phone number of applicant
	A.EmailAddress	String	50		Email Address of applicant

Entity	Field Name	Data Type	Field Length	Constraint	Description
Board Member	MemberID	Int	4	Primary Key	Member ID, auto generated
	B.First_Name	String	25		First name of board member
	B.Last_Name	String	25		Last name of board member
	Join_Date	Date	8		Date the board member joined the board
	B.PhoneNumer	Int	10		Phone number of board member
	B.EmailAddress	String	50		Email Address of board member

Entity	Field Name	Data Type	Field Length	Constraint	Description
Volunteer	VolunteerID	Int	4	Primary Key	Volunteer ID, auto generated
	V.First_Name	String	25		First name of volunteer
	V.Last_Name	String	25		Last name of volunteer
	V.Address	String	50		Home address of volunteer
	V.PhoneNumber	Int	10		Phone number of volunteer
	V.EmailAddress	String	50		Email Address of volunteer
	CampID	Int	4	Foreign Key	Camp ID of the camp worked
	Hours Available	Int	2		Hours available for volunteer to work in a week

Entity	Field Name	Data Type	Field Length	Constraint	Description
Camp	CampID	Int	4	Primary Key	Camp ID, auto generated
	C.Address	String	50		Address of where the camp will be held
	Beginning_Date	Date	8		Starting date of camp
	Ending_Date	Date	8		Date the camp ends
	Capacity	Int	3		Maximum Number of campers allowed

Entity	Field Name	Data Type	Field Length	Constraint	Description
Camper	CamperID	Int	4	Primary Key	Camper ID, auto generated
	Ca.First_Name	String	25		First name of camper
	Ca.Last_Name	String	25		Last name of camper
	Ca.Address	String	50		Home address of camper
	Ca.School	String	50		Name of school camper attends
	Ca.Birthdate	Date	8		Date of birth of camper
	Ca.Age	Int	3		Age of camper
	Ca.EmailAddress	String	50		Email address of camper
	Ca.FirstPhoneNumber	Int	10		Primary phone number of camper
	Ca.SecondPhonerNumber	Int	10		Secondary phone number of camper
	Ca.ParentFirstName	String	25		First name of camper's parent
	Ca.ParentLastName	String	25		Last name of camper's parent

Entity	Field Name	Data Type	Field Length	Constraint	Description
Event	EventID	Int	4	Primary Key	Event ID, auto generated
	Event_Type	String	25		A descrption of the type of event
	E.Address	string	50		Address of where the event is being held
	E.Date	Date	8		Date of the event
	E.Start_Time	Double	5		Starting time of the event
	E.End_Time	Double	5		Ending time of the event
	E.Attendees	Int	4		Number of attendees at the event
	E.Income	Double	10		Money made from the event
	E.Expenses	Double	10		Money spent on the event

Entity	Field Name	Data Type	Field Length	Constraint	Description
Event Goer	EventGoerID	Int	4	Primary Key	Event Goer ID, auto generated
	EventID	Int	4	Foreign Key	Event ID of the event the event goer is attending
	EG.FirstName	String	25		First name of event goer
	EG.LastName	String	25		Last name of event goer

Entity	Field Name	Data Type	Field Length	Constraint	Description
Donor	DonorID	Int	4	Primary Key	Donor ID, auto generated
	D.FirstName	String	25		First name of donor
	D.LastName	String	25		Last name of donor
	D.PhoneNumber	Int	10		Phone number of donor
	D.EmailAddress	String	50		Email address of donor

Entity	Field Name	Data Type	Field Length	Constraint	Description
Donation	DonationID	Int	4	Primary Key	Donation ID, auto generated
	DonorID	Int	4	Foreign Key	Donor ID of the donor who made the donation
	Do.Type	String	50		Type of donation made
	Do.Description	String	300		Description of the donation

Entity	Field Name	Data Type	Field Length	Constraint	Description
Feed Inventory	FeedID	Int	4	Primary Key	Feed ID, auto generated
	FeedName	String	25		Name of the feed
	F.Description	String	300		Description of the feed
	F.Donated	Bool	N/A		Was the feed donated?
	DonorID	Int	4	Foreign Key	DonorID of the person who donated the feed

Entity	Field Name	Data Type	Field Length	Constraint	Description
Animal	AnimalID	Int	4	Primary Key	Animal ID, Auto Generated
	OwnerID	Int	4	Foreign Key	Owner ID of the owner of the animal
	A.Name	String	25		The name of the animal
	A.Birthdate	Date	8		Date the animal was born
	A.Age	Int	3		The age of the animal
	A.Medications	String	300		The medivines that the animal is currently taking.
	A.Donated	Bool	N/A		Was the animal Donated?
	DonorID	Int	4	Foreign Key	Donor ID of the donor of the animal

Entity	Field Name	Data Type	Field Length	Constraint	Description
Owner	OwnerID	Int	4	Primary Key	Owner ID, Auto Generated
	O.LastName	String	25		Last name of the owner
	O.FirstName	String	25		First name of the owner
	AnimalID	Int	4	Foreign Key	Animal ID of the animal the owner owns
	O.Address	String	50		Home address of the owner
	O.PhoneNumber	Int	10		Phone number of the owner
	O.EmailAddress	String	50		Email address of the owner

Entity	Field Name	Data Type	Field Length	Constraint	Description
Auction Item	ItemID	Int	4	Primary Key	Item ID, Auto Generated
	AuctionID	Int	4	Foregin Key	Auction ID of the auction the item has been donated for
	DonorID	Int	4	Foregin Key	Donor ID of the donor of the item
	AI.Type	String	50		The type of item that has been donated
	AI.DateRecieved	Date	8		The date the item was received

Entity	Field Name	Data Type	Field Length	Constraint	Description
Auction	AuctionID	Int	4	Primary Key	Auction ID, Auto Generated
	A.Type	String	50		Type of auction that is taking place
	A.Address	String	50		Address of where the auction is being held
	A.Date	Date	8		The date the auction is taking place
	A.Start_Time	Double	5		Starting time of the auction
	A.End_Time	Double	5		Ending time of the auction
	A.Attendees	Int	4		Number of attendees at the auction
	A.Income	Double	10		Money made from the auction
	A.Expenses	Double	10		Money spent on the auction

Entity	Field Name	Data Type	Field Length	Constraint	Description
Grant	GrantID	Int	4	Primary Key	Grant ID, Auto Generated
	DonationID	Int	4	Foreign Key	Donation ID associated with this received grant
	G.Amount	Double	10		The amount of the money the grant is worth
	G.Description	String	300		Description of the grant
	G.Date	Date	8		The date the grant was received

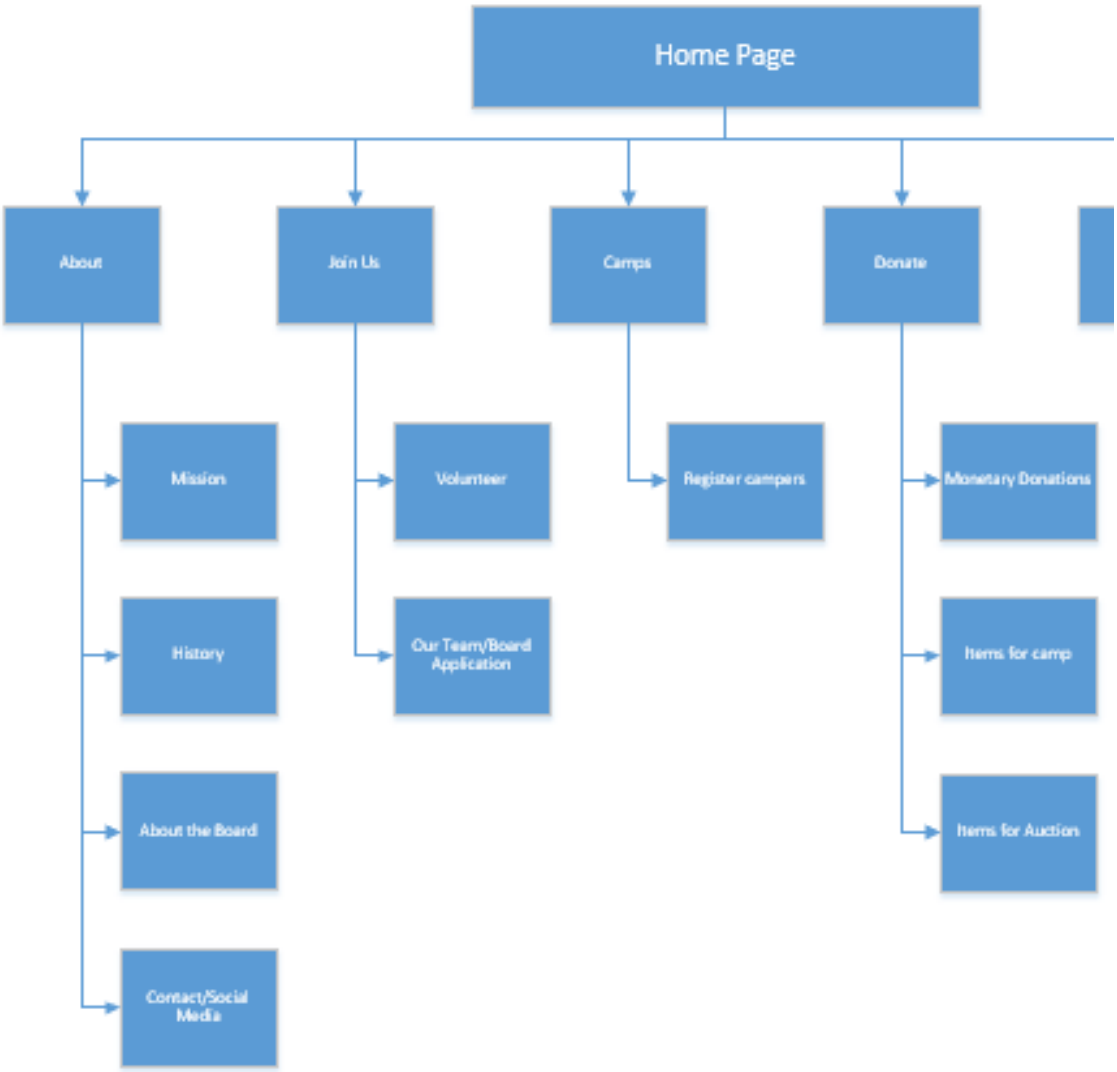
Entity	Field Name	Data Type	Field Length	Constraint	Description
Grantor	GrantorID	Int	4	Primary Key	Grantor ID, Auto Generated
	GrantorCompany	String	25		Name of the grantor's company
	Gr.FirstName	String	25		First name of the grantor
	Gr.LastName	String	25		Last name of the grantor
	Gr.PhoneNumber	Int	10		Phone number of the grantor
	Gr.EmailAddress	String	50		Email Address of the grantor
	Gr.Address	String	50		Address of the grantor

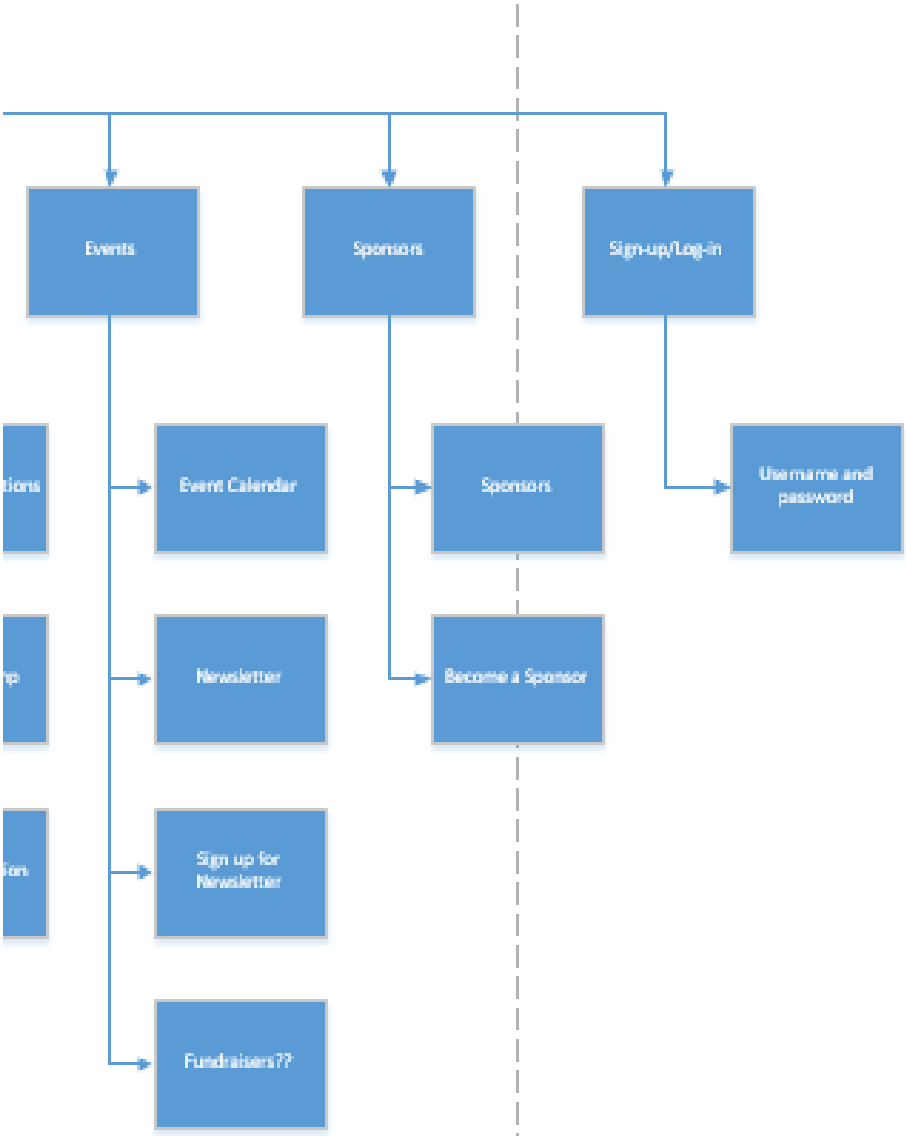
Entity	Field Name	Data Type	Field Length	Constraint	Description
Gear Inventory	GearID	Int	4	Primary Key	Gear ID, Auto Generated
	GearName	String	25		The name of the gear
	Ge.Type	String	50		The type of gear
	Ge.Donated	Bool	N/A		Was the gear donated?
	DonorID	Int	4	Foreign Key	DonorID of the person who donated the gear

Entity	Field Name	Data Type	Field Length	Constraint	Description
Newsletter Subscribers	SubscriberID	Int	4	Primay Key	Subscriber ID, Auto Generated
	SubscriberFirstName	String	25		The first name of the subscriber
	SubscriberLastName	String	25		The last name of the subscriber
	SubscriberEmail	String	50		Email Address of the subscriber

User Interface Navigation Diagram

User Interface Navigation Diagram



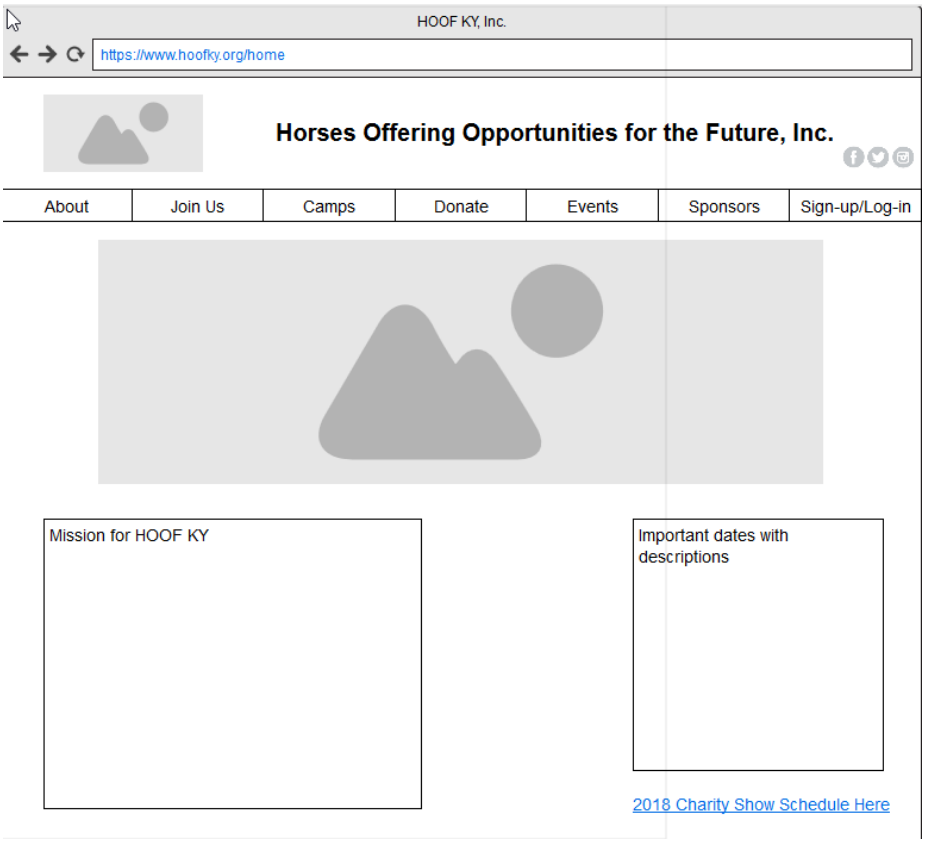

Events
Sponsors
Sign-up/Log-in
Event Calendar
Newsletter
Sign up for Newsletter
Fundraisers??
Sponsors
Become a Sponsor
Username and password

Screen Layouts

Use Interface Prototypes

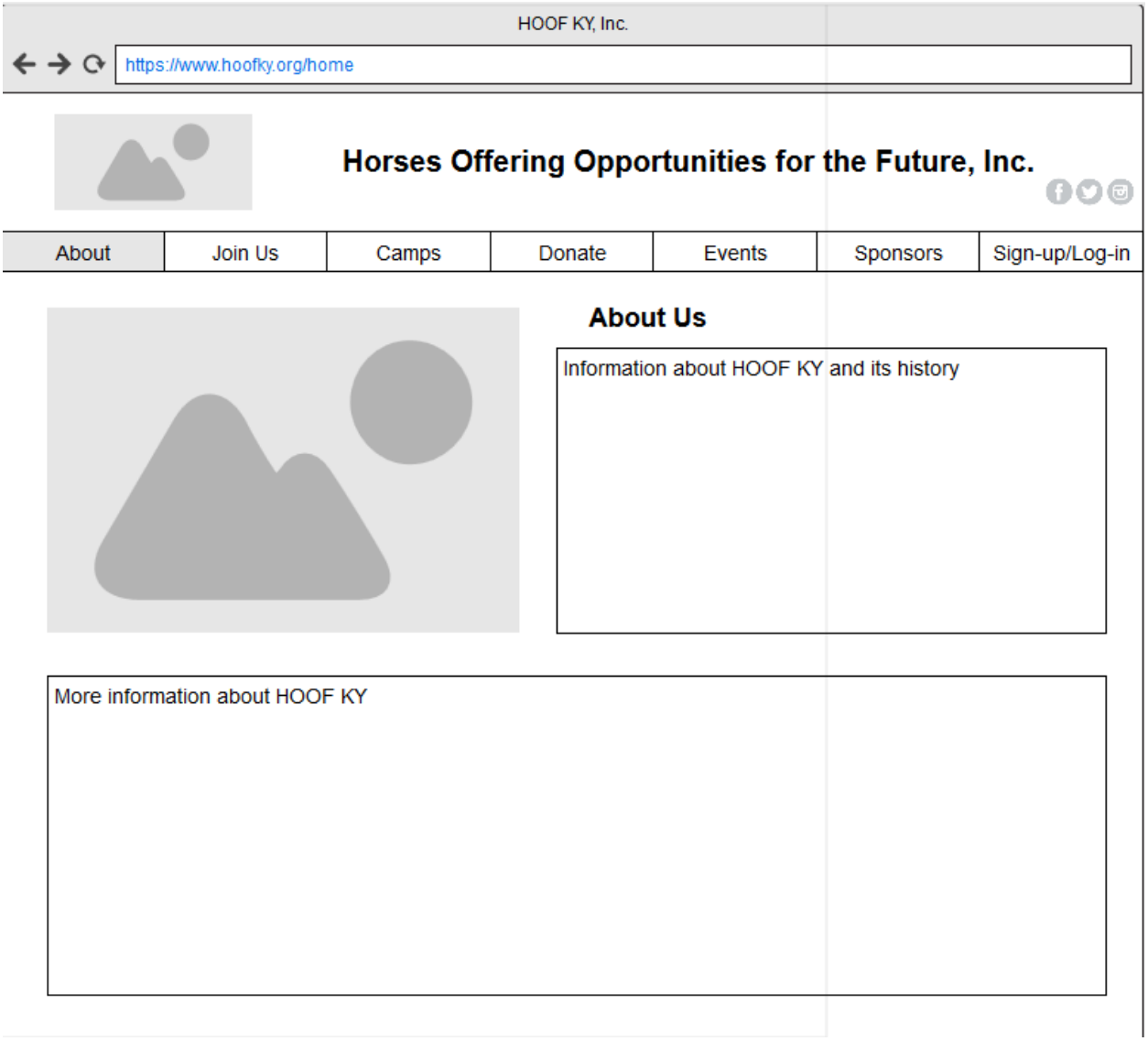
Prototypes are visual representations that show how the web page elements will function. These allow the user to understand how to interact with the web page. These high risk prototypes correspond with their respective high risk use case on pages 6-36.

Home Page:



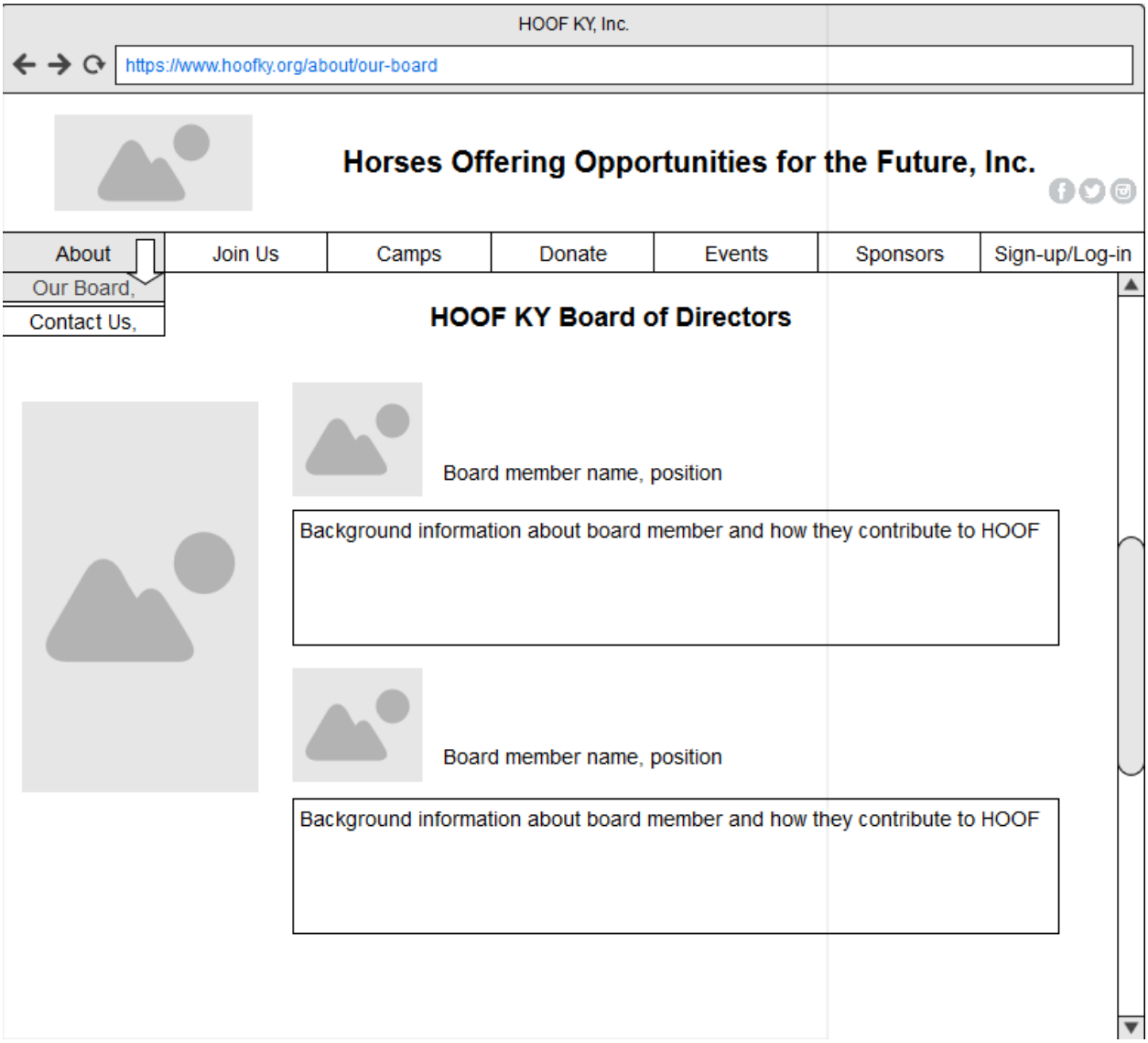
This screen layout represents what the home page for HOOF KY website could look like when someone visits the HOOFKY website. The home page has tabs labeled About, Join Us, Camps, Donate, Events, Sponsors, and Sign-up/Log-in. These tabs can be designed to have dropdown style boxes for more specific topics of the tab category. At the top of the page is also a social media bar that the user of the website can click on to be navigated to HOOF KY's Facebook, Twitter, or Instagram page.

About Us Page:



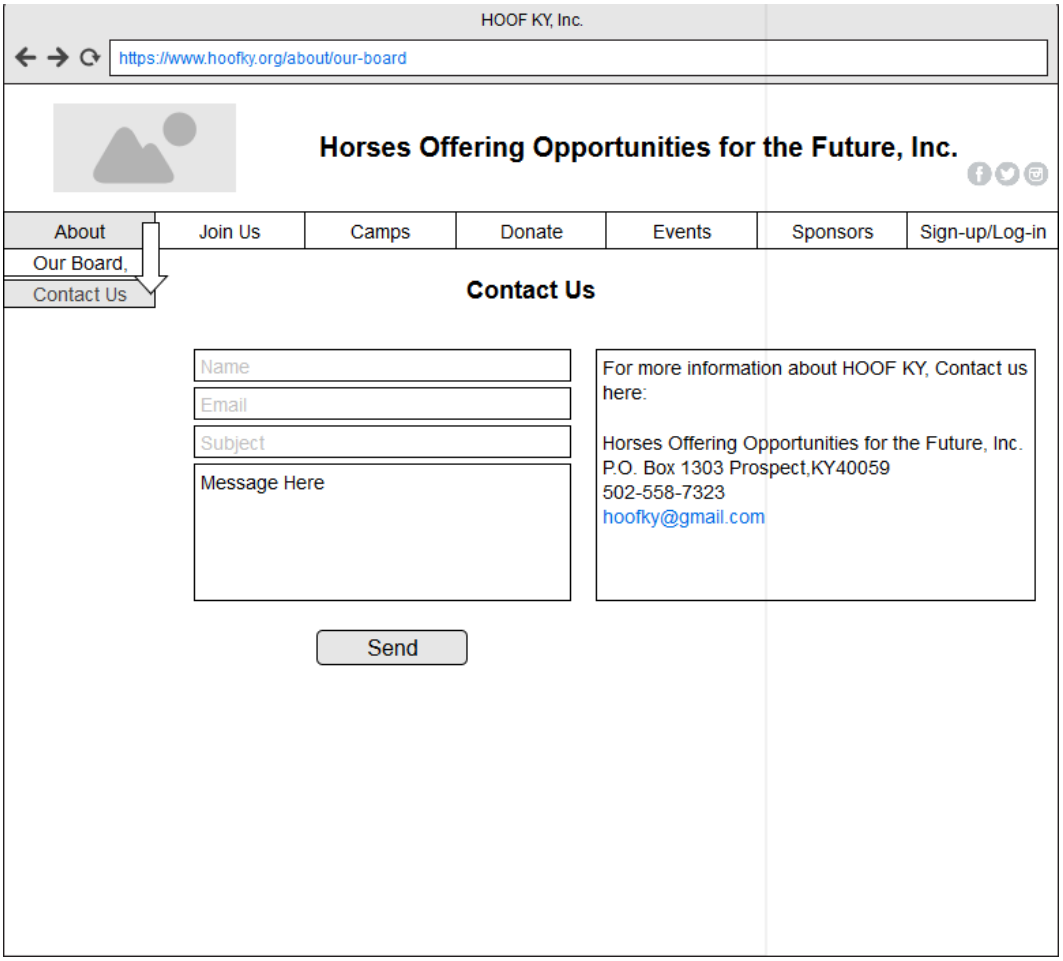
This screen layout represents what occurs when a user clicks on the About tab on the HOOF KY website. The About page contains information about HOOF KY and the history of HOOF KY as a non-profit organization.

Our Board Page:



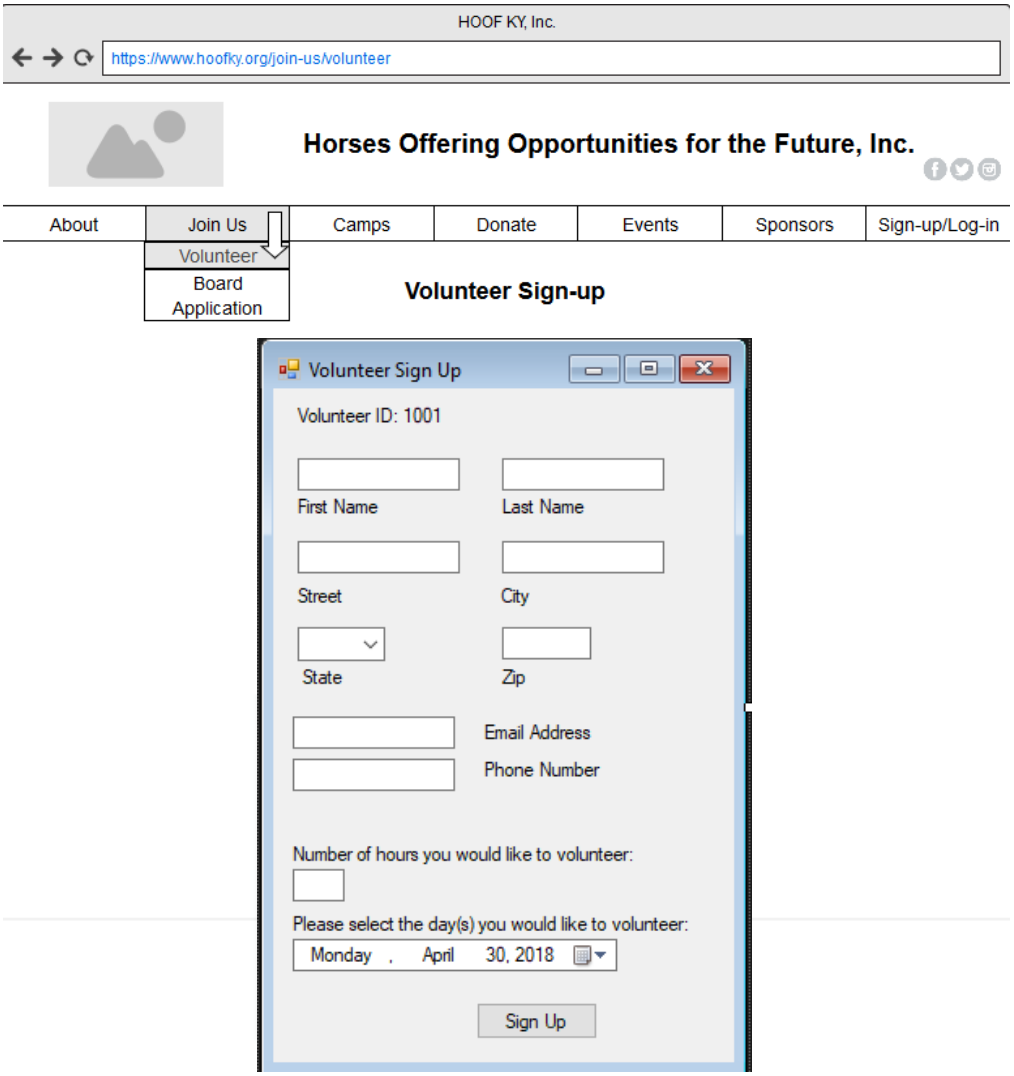
This screen layout represents what occurs when a user hovers over the About tab, and clicks on the Our Board tab on the dropdown menu. This page contains a list of the Board of Directors, along with pictures and information pertaining to each member.

Contact Us Page:



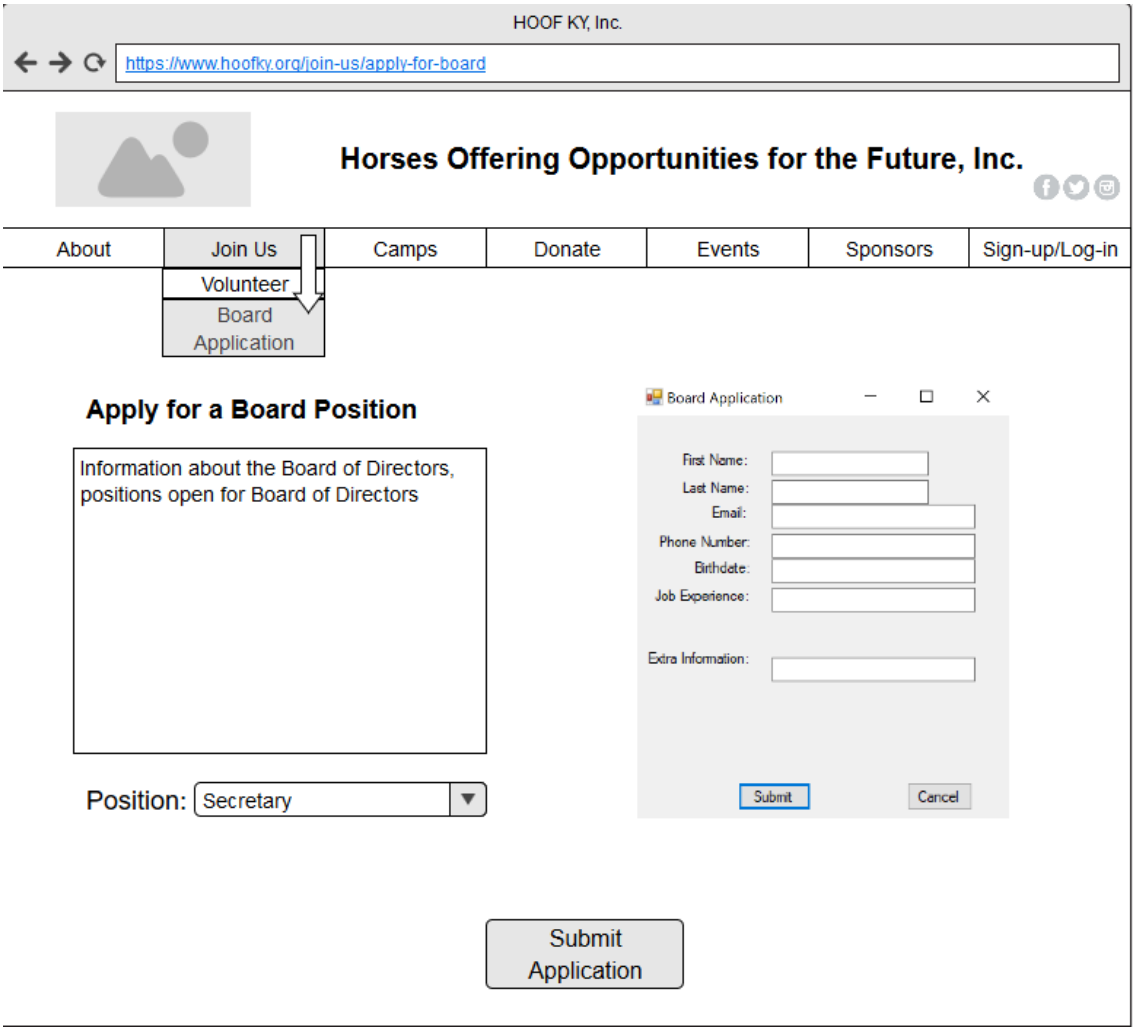
This screen layout represents what occurs when a user hovers over the About tab, and clicks on the Contact Us tab on the dropdown menu. This page contains a form that the user can fill out if they want to contact HOOF KY administration about something. The user inputs their name, email, a subject, and the message that they want to send. This page also contains essential contact information for HOOF KY including HOOF KY's P.O. BOX, business phone number, and email.

Volunteer Sign-up Page:



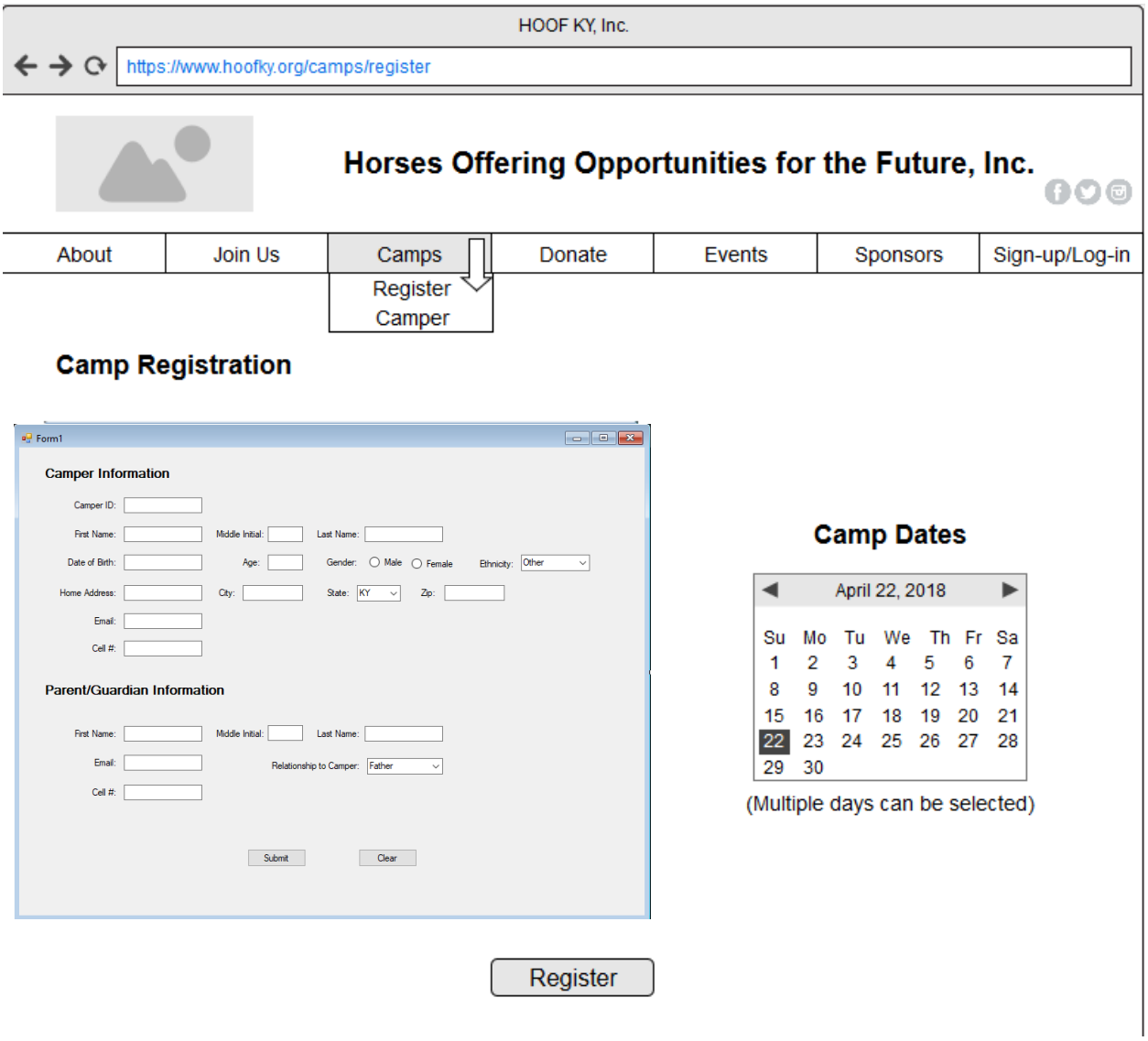
This screen layout represents what occurs when a user hovers over the Join Us tab, and selects the Volunteer dropdown box. The volunteer sign-up page contains a form that someone could use to sign up for volunteering by inputting the event that they wish to sign up for, the dates that they are available, as well as the amount of available work hours per week.

Apply for Board Position Page:



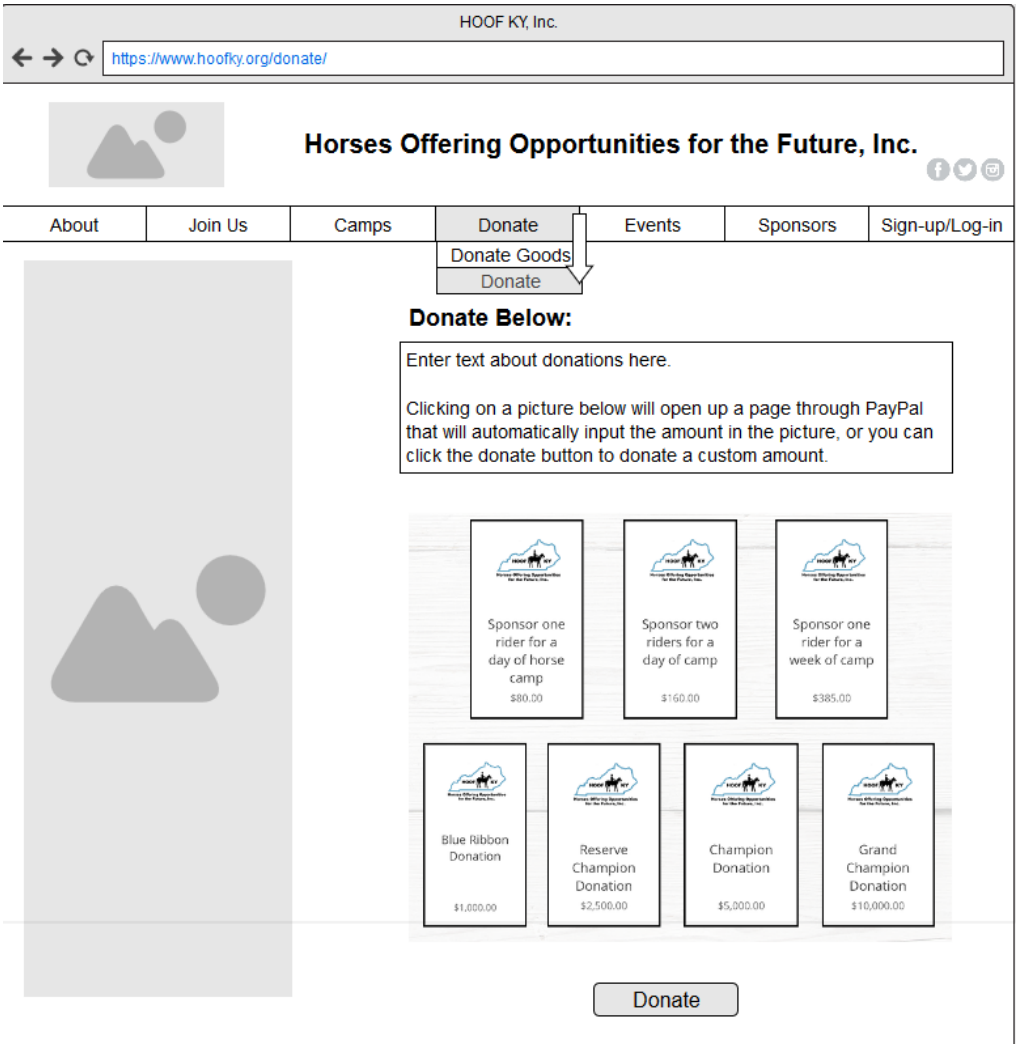
This screen layout represents what occurs when a user hovers over the Join us tab, and selects the Board Application dropdown box. The Board Application page contains information about the Board of Directors, as well as positions that are open. The user can fill out the board application and select the position that they wish to apply for, and can submit the application on HOOF KY's website.

Register Campers Page:



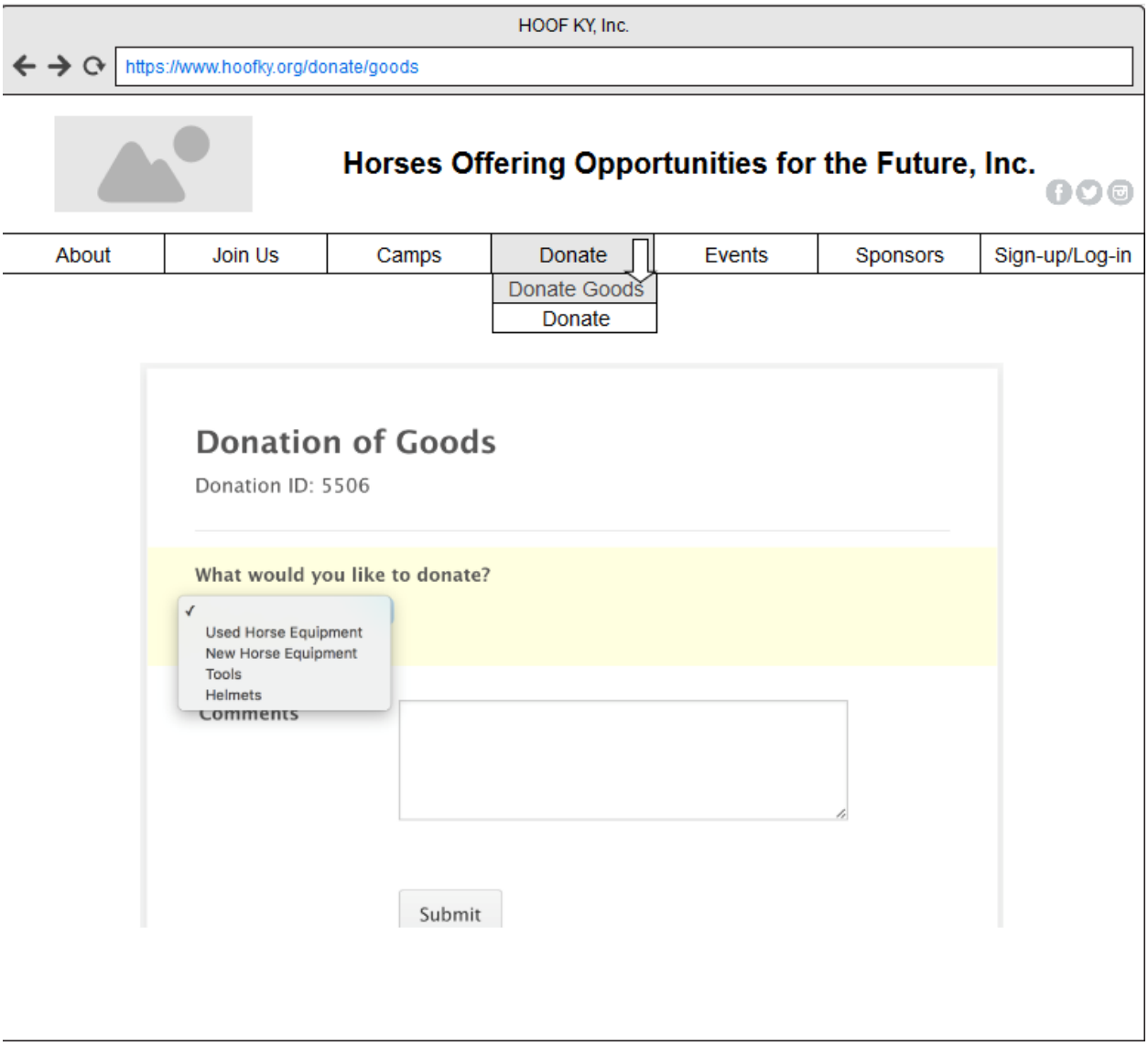
This screen layout represents what occurs when a user clicks on the Register Camper dropdown box under Camps. The register camper page has a form that a user can fill out, and also has a calendar set on the right side for the user to choose the camp dates that they wish to attend.

Donate Page:



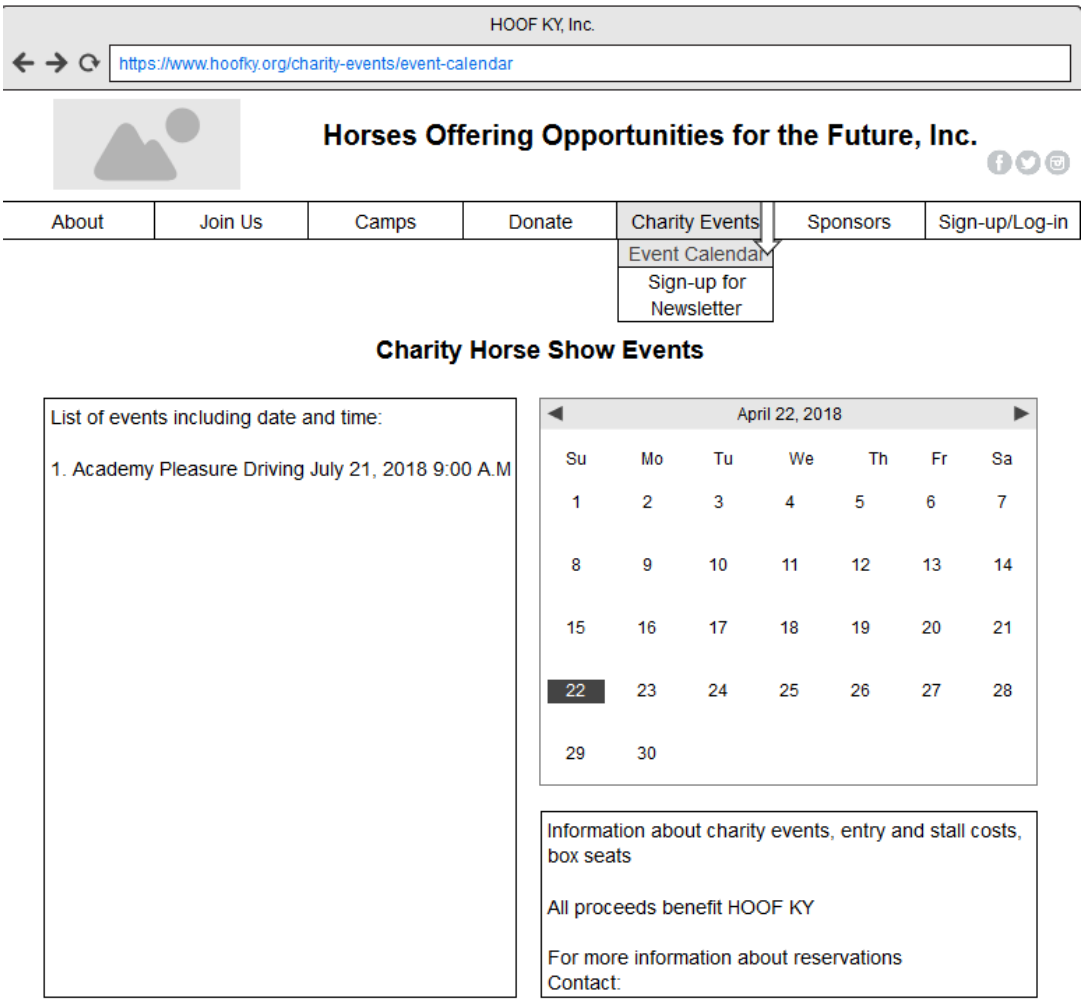
This screen layout represents what occurs when a user clicks on the Donate dropdown box under the Donation tab. The donation page has pictures for set donation amounts that the user can click on to donate that amount, or the user can click on the Donate button to donate a custom amount. This donation process uses PayPal, and each picture can redirect you to a PayPal page for each amount given on the pictures.

Donate Goods Page:



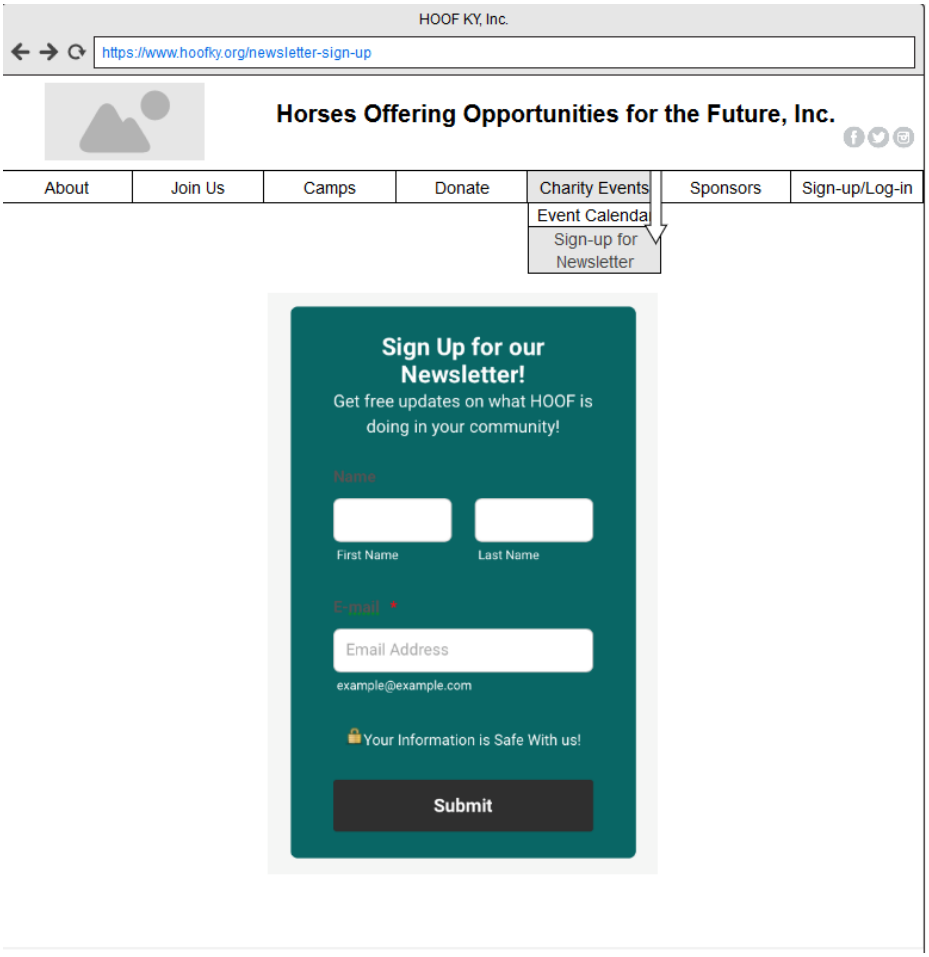
This screen layout represents what occurs when a user clicks on the Donate Goods dropdown box under Donate. The Donate Goods page is used to submit a request for the donation of goods such as horse equipment, tools, or helmets.

Charity Events Calendar Page:



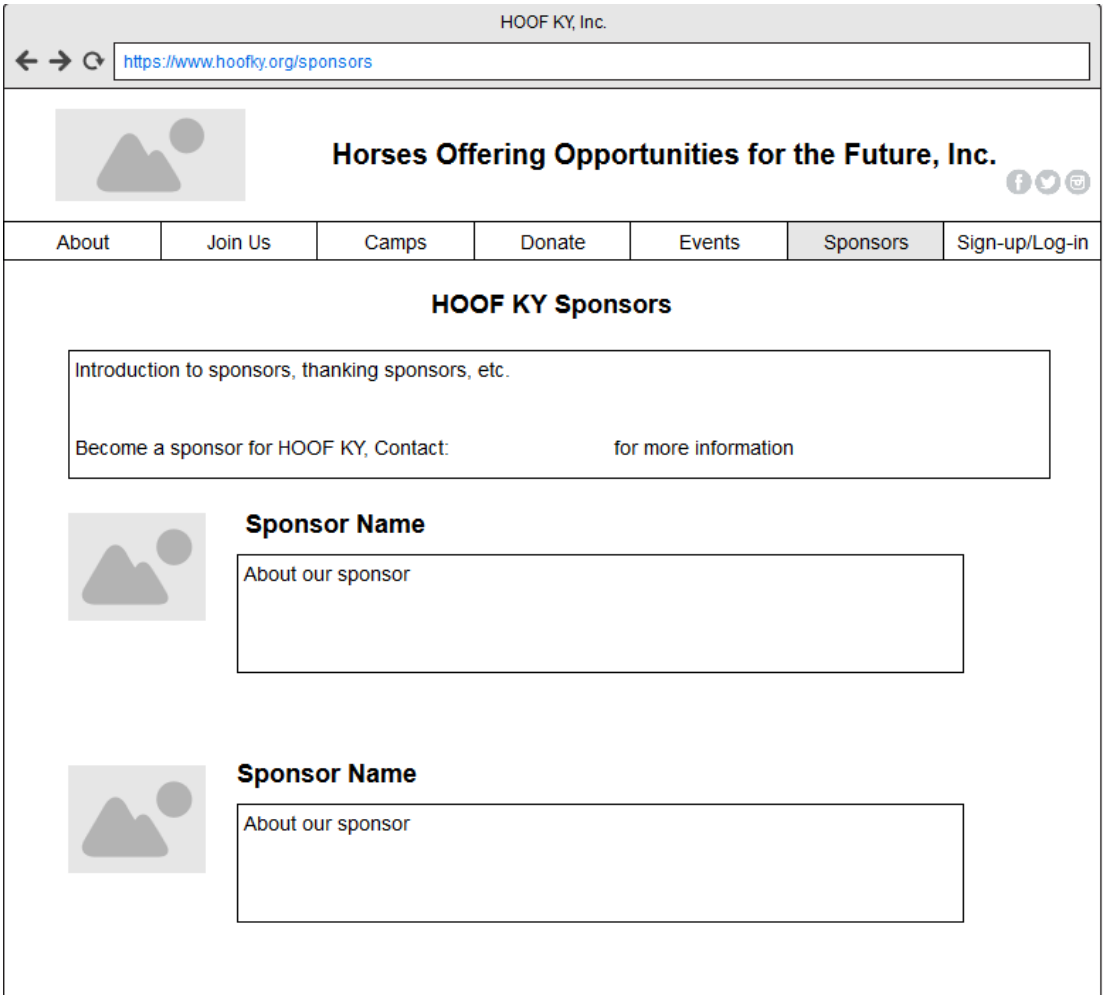
This screen layout represents what occurs when a user clicks on the Event Calendar dropdown box under Charity Events. The Charity Events Calendar page can contain a list of the charity events that HOOF KY participates in, as well as a calendar that shows all the dates and times for the charity events. This page also could contain information about the charity events such as admission costs or stall costs.

Sign-up for Newsletter Page:



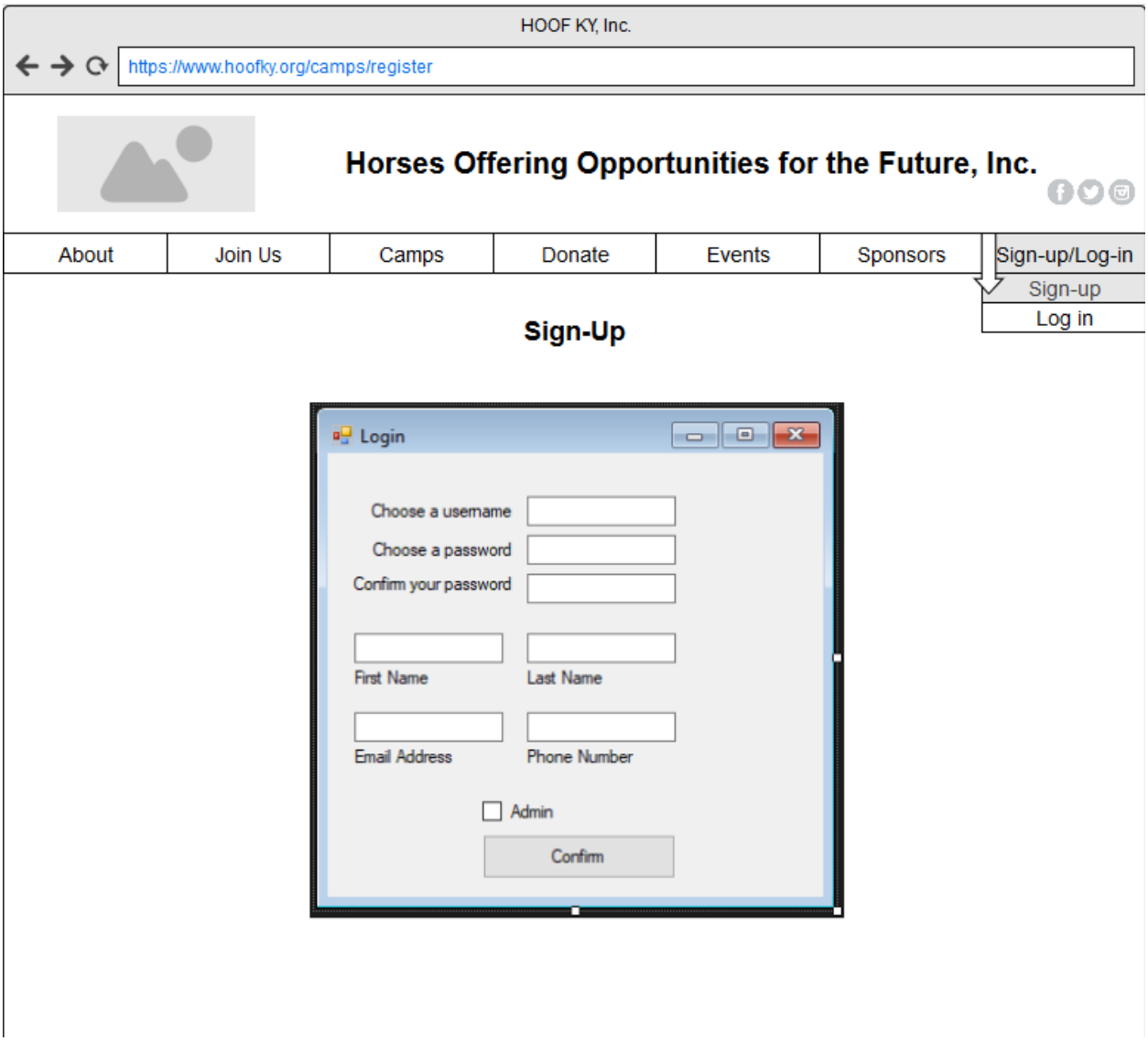
This screen layout represents what occurs when a user clicks on the Sign-up for Newsletter dropdown box under Charity Events. This page contains a form that can be filled out by the user to receive email updates on what HOOF is doing in your community.

Sponsors Page:



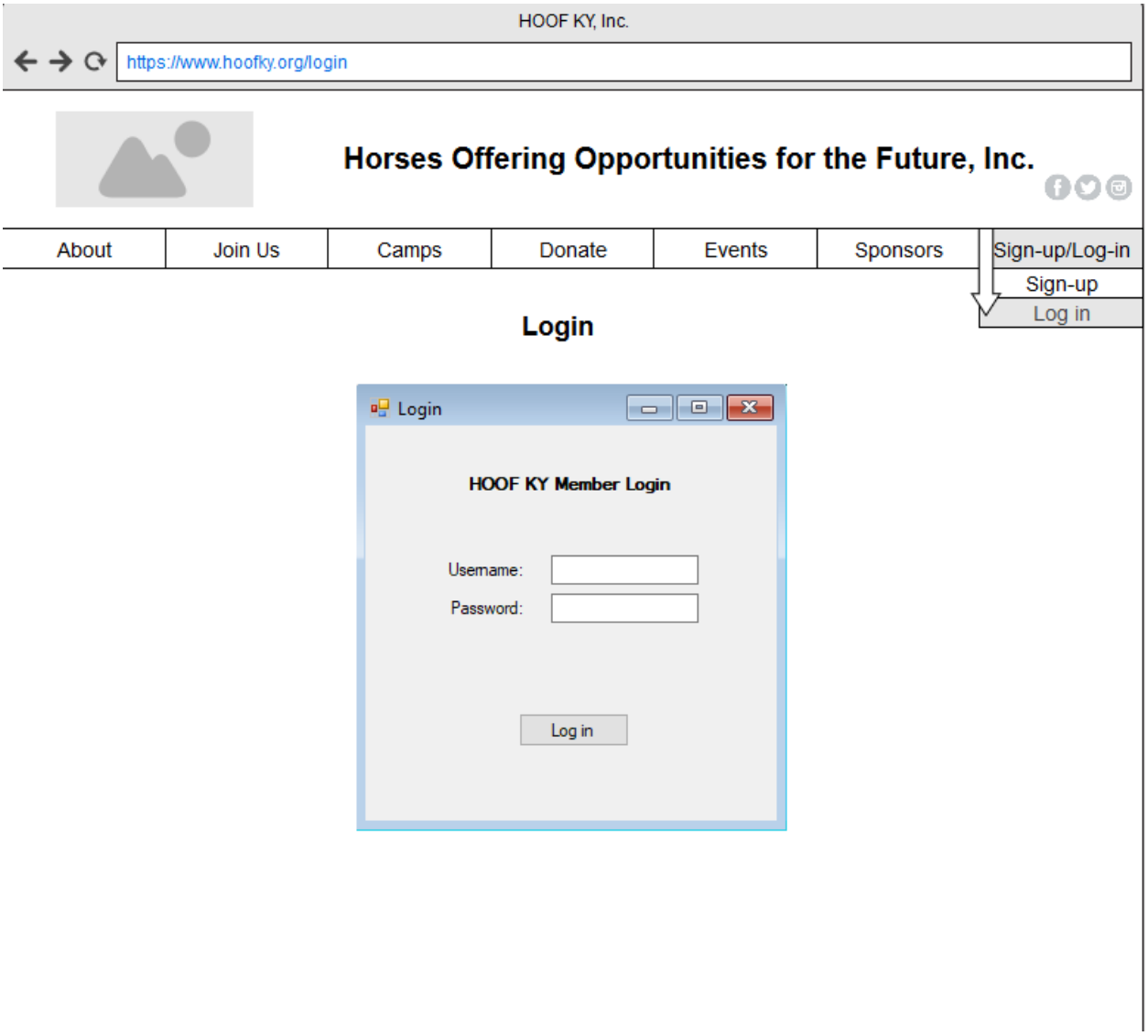
This screen layout represents what occurs when a user clicks on the Sponsors tab. The Sponsors page could be used to show thanks for the sponsors, as well as list all of the sponsors that are contributing to HOOF KY.

Sign-up Page:



This screen layout represents what occurs when a user clicks on the Sign-up dropdown box under Sign-up/Log-in. This page contains a form that is used to sign up for the HOOF KY website for easier access to other features on the website

Log-in Page:



This screen layout represents what occurs when a user clicks on the Log in dropdown box under Sign-up/Log-in. This page contains a form that the user may use to sign in to the HOOF KY website if they want to sign up for camps, events, or donate.

Physical Architecture Design

Physical Architecture Design

HOOF will benefit the most from obtaining a content management system (CMS), an online payment system (PS), a database management system (DBMS) stored on a cloud and a stronger social media presence via Facebook. We will address these needs from two different viewpoints: the design viewpoint and the realization viewpoint. The design viewpoint will outline the general need of each component and the realization viewpoint will go into further detail about the model of each component we have chosen.

Design Viewpoint

A content management system is a software application that allows a user to consolidate, create and update a website. A CMS is a primary need of HOOF to further their online presence and obtain an additional donor base. Utilizing a CMS will allow HOOF to also offer a platform for a PS which will make it easier for donors to make donations thereby increasing the likelihood that people will donate.

An online payment system is an internet application that allows a user to process transactions to a from other users. An online PS will allow donors to donate in a more convenient manner by donating directly on HOOF's WordPress website. This will allow for greater potential donor growth and more convenient giving options for the donors themselves.

A database management system is a software application that allows users to store data in one convenient and secure location. A DBMS will make storing and retrieving data effortless for HOOF. The DBMS will allow HOOF to store all their data in a single location. Consolidating data within a DBMS will make everyday operations simpler for HOOF, specifically when they need to do research for writing grants. Currently it takes HOOF 8-10 hours to write a single grant because of the time it takes to consolidate data that is scattered. A DBMS will dramatically reduce the amount of time it takes to write a grant which could result in more grants written and more grant money received.

A stronger social media presence on Facebook will increase HOOF's donor base and provide the community with a greater understanding of how they wish to help disadvantaged youth.

Realization Viewpoint

The best CMS option for HOOF is WordPress. We have selected this system because it is easy to use and upkeep, inexpensive, and produces good quality websites. Many content management systems are difficult to use and upkeep but WordPress is user friendly and does not require excess time to update the system on a day to day basis. It also includes many different tools at no extra cost to make HOOF's website unique. Some of these tools include Themes to assist in design and Media Management that includes a gallery for pictures and easily embedded media buttons.

The best PS for HOOF is PayPal, an easy to use system that is non-profit friendly and widely known to the average user. PayPal has one of the cheapest cost structures in the PS market but also offers a significant discount for 501(c)(3) charities. It also does not require a donor to have his or her own PayPal account in

order to donate, making it extremely easy to obtain new donors. HOOF can be confident in PayPal; it is a stable company founded in 1998 and one of the top online payment systems today, utilized by over 188 million users.

The most beneficial DBMS option for HOOF is Microsoft SQL Server. It is a non-expensive option that also provides quality service. A benefit of this particular piece of software is that it comes equipped with a large technical support community, being owned by Microsoft and having first launched in 1989.

Design Features for Security Concerns

Design Features for Security Concerns

This section will provide an overview of how different security aspects will be taken into consideration and embedded into our system. Addressing these issues is vital to HOOF and their new system. This system will allow HOOF to protect their company's private information as well as that of their donors and campers.

The main features that HOOF needs to implement include a virus control, encryption and authentication, accessibility limitations.

Requirement	Definition	Application
System Value Estimates	Estimated business value of the system and it's data.	HOOF's data will be well protected from loss by storing it on a cloud server. This will decrease loss and ensure HOOF will not need to spend large amounts recovering lost data.
Access Control Requirements	Limitations of who can access what data.	HOOF site members will have limited access to only their individual information whereas Admin (Board Member) approved access will be able to see other member's information as well as added features such as fundraiser financial information.
Encryption and Authentication Requirements	Defines what data will be encrypted as well as where and when authentication will be needed for user access.	Encryption from each user's computer will provide assurance of secure transactions. Logging onto HOOF's website will require authentication from the server.
Virus Control Requirements	Requirements to control the spread of viruses.	A virus detection software will add another layer of data protection to secure HOOF's information from potential virus threats.

Gantt Charts

Gantt Charts

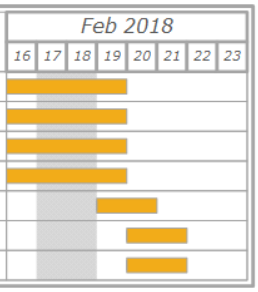
Each Gantt Chart for the iterations has a specific ID number for each task that was completed, along with the members who completed each task. Each task has a set start and finish date, including the duration of each individual task.

For Iteration 2, the list of use cases, the vision document, and the architecture considerations had to be completed in order to complete the risk analysis and the inception phase prototypes. The Gantt chart required that every other task is complete to record the start and finish dates of each task.

For Iteration 3, the use cases must be defined and explained in detail before creating the use case diagram and the use case prototype.

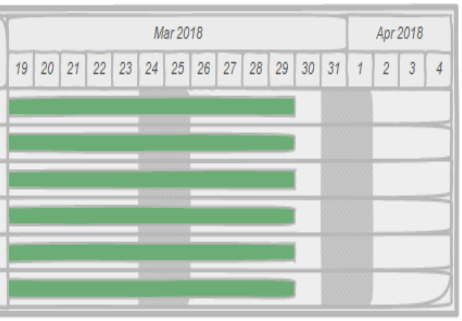
Iteration 2

ID	Task Name	Member Allocation	Start	Finish	Duration
1	System Requirements	Emily Green	2/16/2018	2/19/2018	2d
2	Feasibility Analysis	Logan Robinson	2/16/2018	2/19/2018	2d
3	Use Cases	James Jordan/Caleb DeSpain	2/16/2018	2/19/2018	2d
4	Initial Architecture Considerations	Caitlin Sullivan	2/16/2018	2/19/2018	2d
5	Risk Analysis	James Jordan	2/19/2018	2/20/2018	2d
6	Gantt Chart	Turner Barnett	2/20/2018	2/21/2018	2d
7	Inception Phase Prototype	Turner Barnett	2/20/2018	2/21/2018	2d



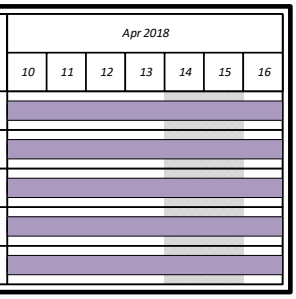
Iteration 3

ID	Task Name	Start	Finish	Duration
1	Use Cases 1-9 (Caitlin Sullivan)	3/12/2018	3/29/2018	14d
2	Use Cases 10-18 (Turner Barnett)	3/12/2018	3/29/2018	14d
3	Use Cases 19-27 (James Jordan)	3/12/2018	3/29/2018	14d
4	Use Cases 28-36 (Emily Green)	3/12/2018	3/29/2018	14d
5	Use Cases 37-46 (Logan Robinson)	3/12/2018	3/29/2018	14d
6	Use Cases 47-54 (Caleb DeSpain)	3/12/2018	3/29/2018	14d



Iteration 5

ID	Task Name	Start	Finish	Duration
1	ERD / Data Dictionary (Caleb, Jimmy)	4/10/2018	4/17/2018	6d
2	Windows Navigation Diagram (Logan, Emly)	4/10/2018	4/17/2018	6d
3	Prototypes (Turner)	4/10/2018	4/17/2018	6d
4	Gantt Chart (Caitlin)	4/10/2018	4/17/2018	6d
5	Deliverables / Power Point (Caitlin)	4/10/2018	4/17/2018	6d



Elaboration Phase Prototypes

Prototypes

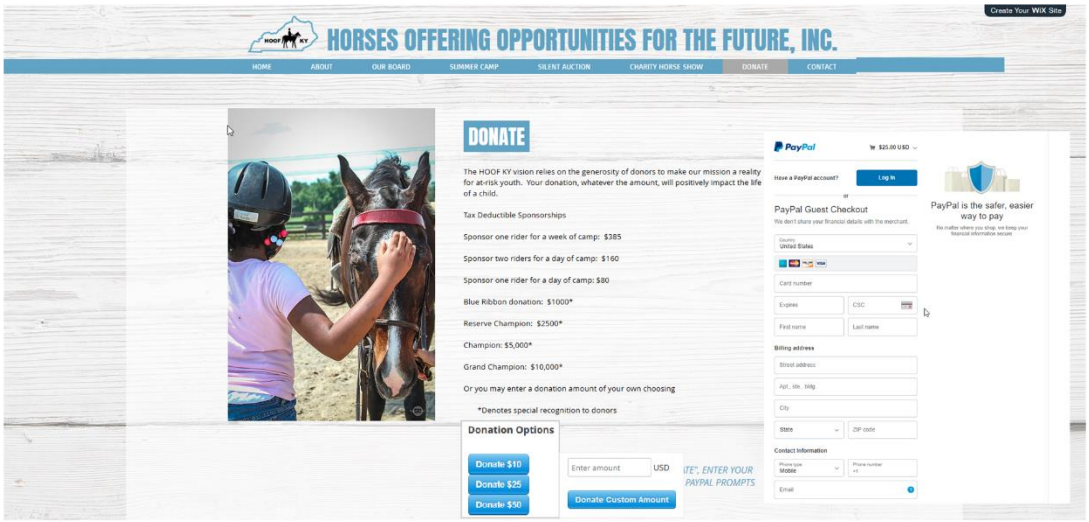
Currently, the HOOF KY website does not have options for people who visit the website to sign up for the HOOF KY events and volunteer work, working social media buttons for Twitter, Instagram, and Facebook, or a working donation button. We have created prototypes for each of these.

Looking at volunteer sign-up and event sign-up prototypes that have been created, people interested in volunteering for HOOF KY could visit the website and click on the “Volunteer Now!”, enter their personal information and available volunteer hours, and submit their volunteer application directly to HOOF KY. The event sign-up prototype is set up in the same way as the volunteer sign-up prototype, but instead of choosing volunteer hours, the people visiting the website will be selecting the event dates that they want to attend.

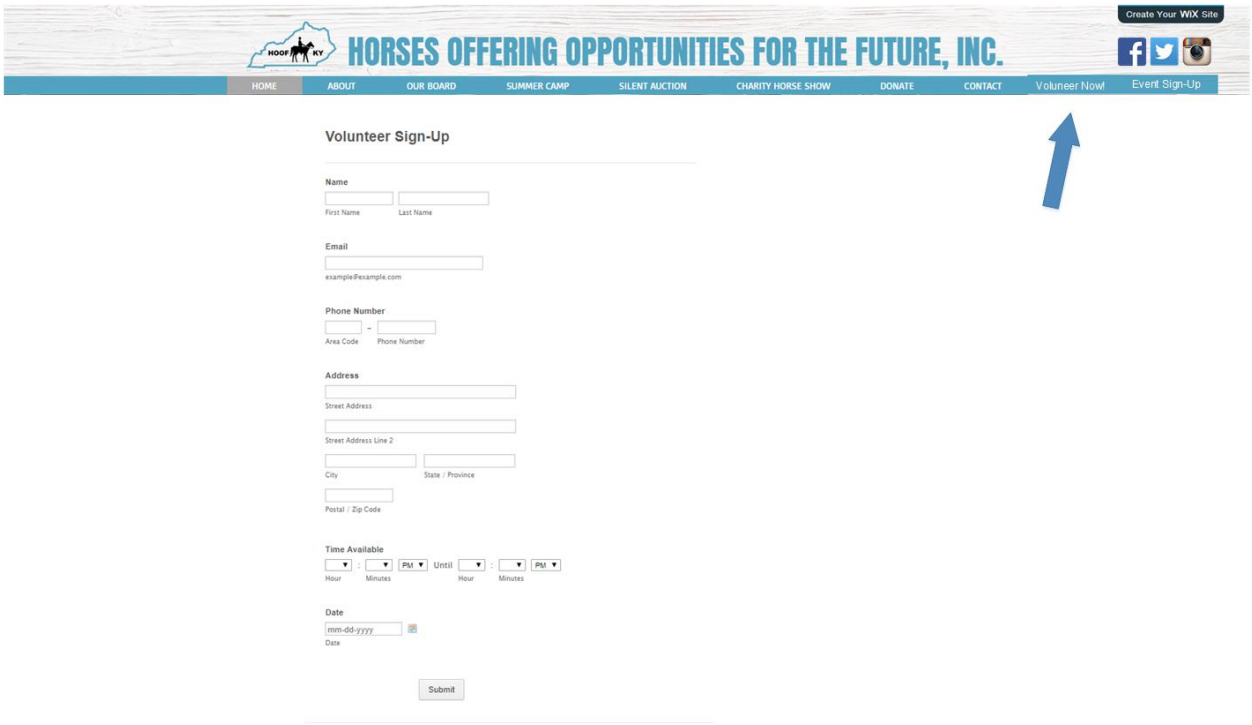
On the volunteer and event sign-up sheets, there is an added social media bar at the top right that can be used by people visiting the website to access HOOF KY’s Facebook, Twitter, and Instagram pages. This simple social media bar can be added using plugins for WordPress found on their website.

The donation page prototype uses donation buttons for standard donation amounts that can be altered, as well as a text box to type in a specific donation amount. The prototype can be created with simple plugins for WordPress and PayPal’s donation button that can be easily implemented through the PayPal website.

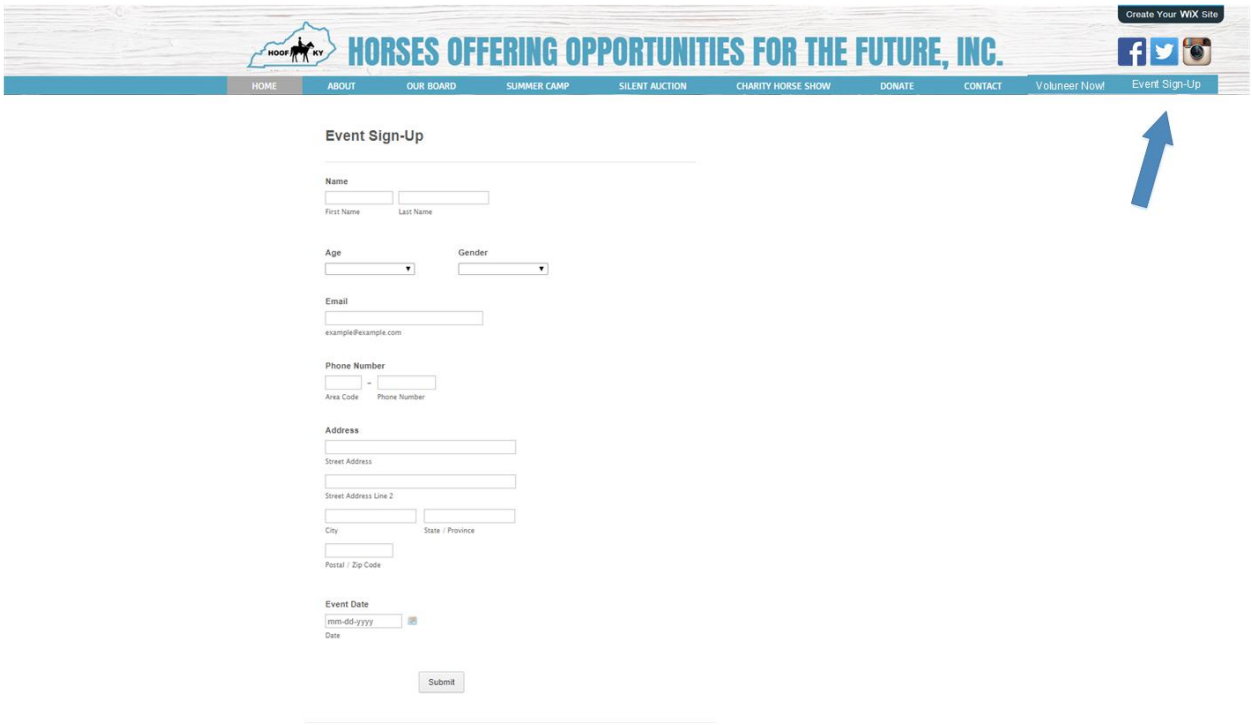
Donation Page



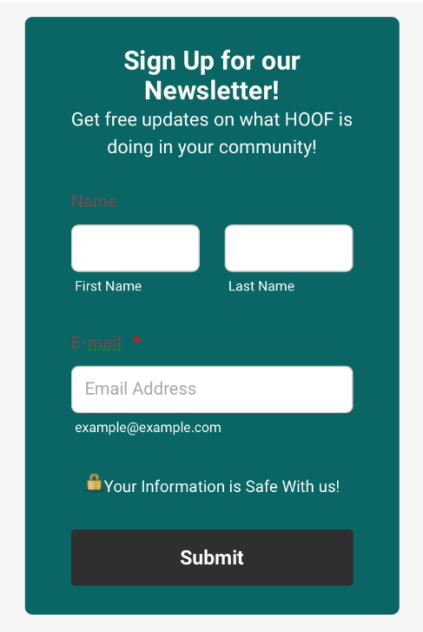
Volunteer Sign-Up Web Form



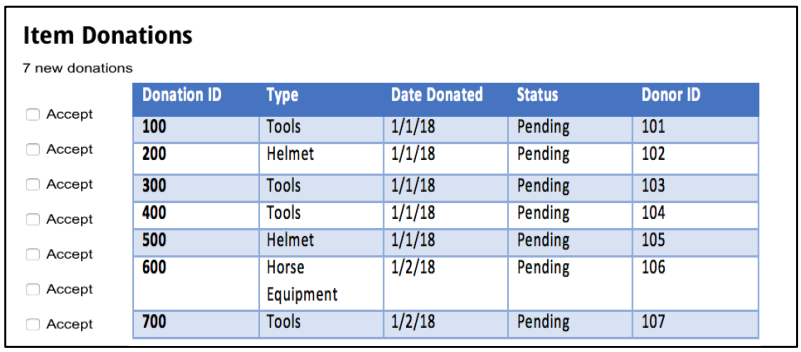
Event Sign-Up Web Form



UC1: Sign Up For Newsletter



UC2: Access Donation Information

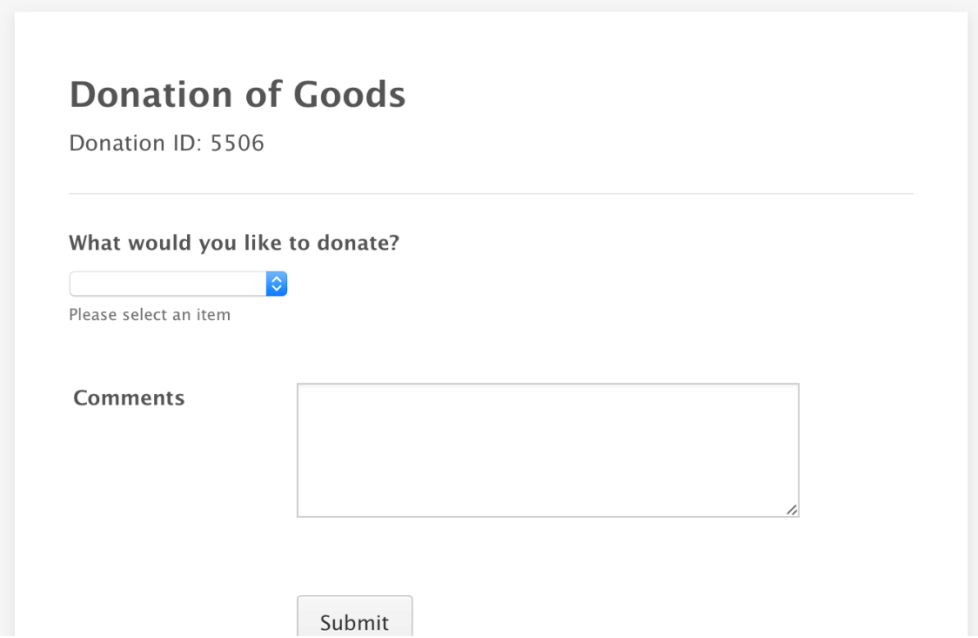


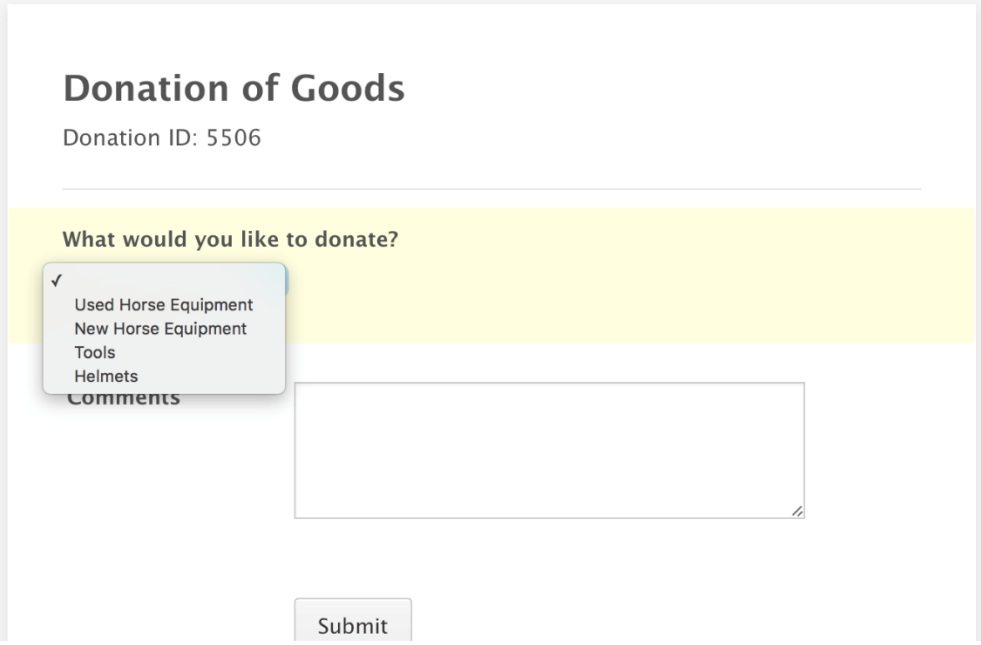
Item Donations

7 new donations

	Donation ID	Type	Date Donated	Status	Donor ID
Accept	100	Tools	1/1/18	Pending	101
Accept	200	Helmet	1/1/18	Pending	102
Accept	300	Tools	1/1/18	Pending	103
Accept	400	Tools	1/1/18	Pending	104
Accept	500	Helmet	1/1/18	Pending	105
Accept	600	Horse Equipment	1/2/18	Pending	106
Accept	700	Tools	1/2/18	Pending	107

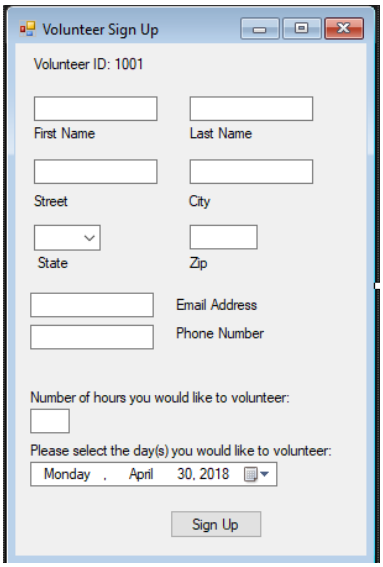
UC3: Sign Up To Donate Goods



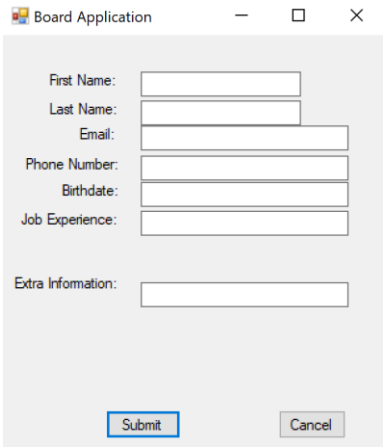


UC 4: Access Volunteer Work Information

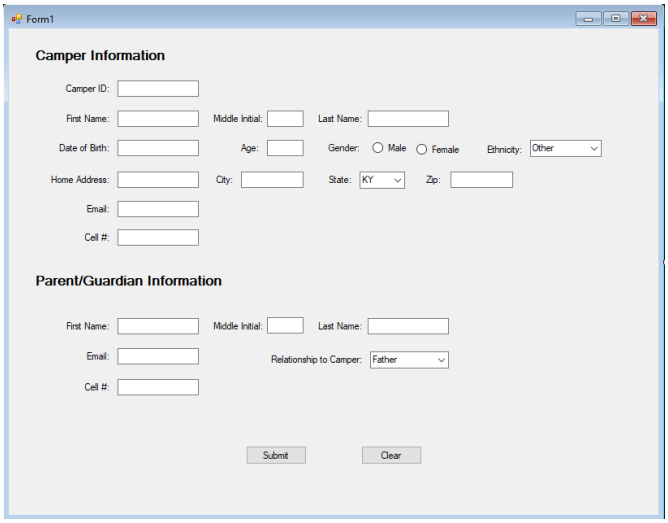
UC5: Sign Up For Volunteer Work

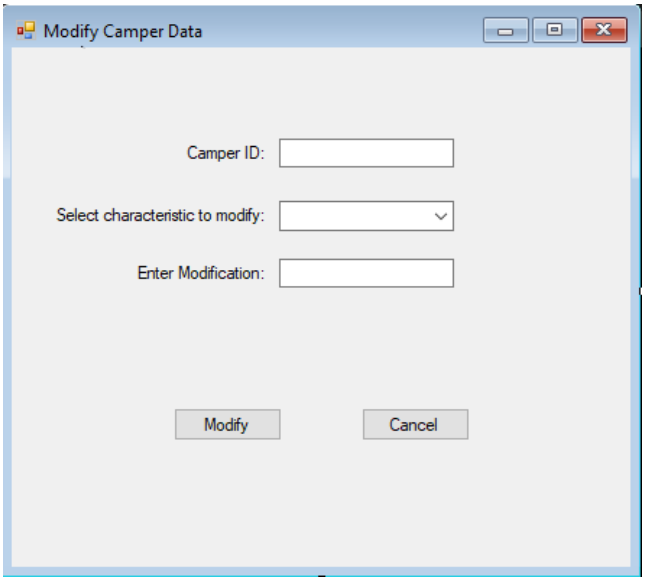


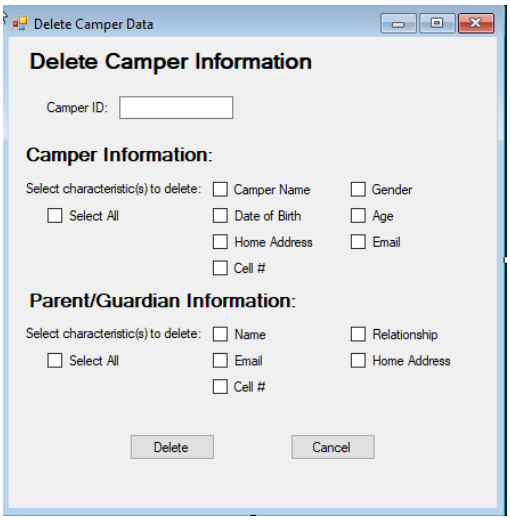
UC8: Apply For A Board Position



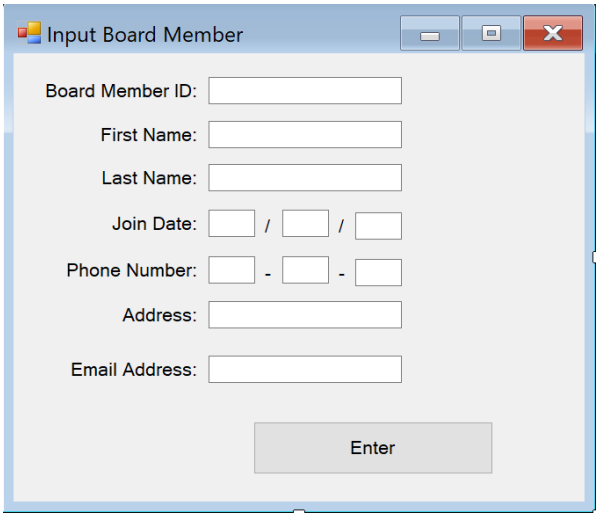
UC 12-14: Edit, Modify and Delete Camper Info

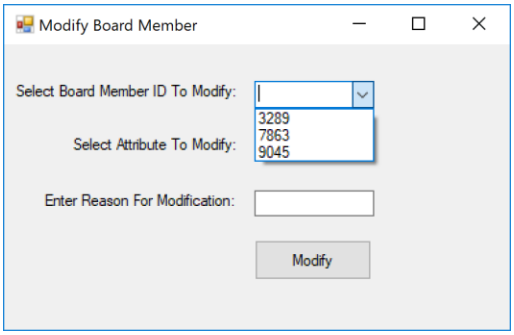


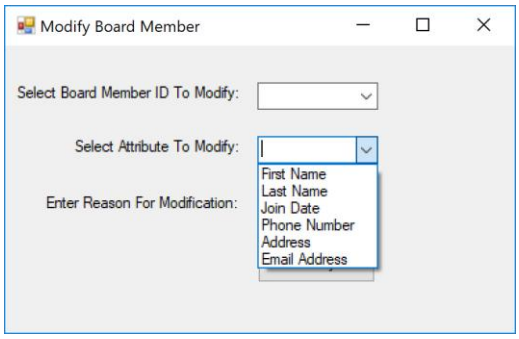


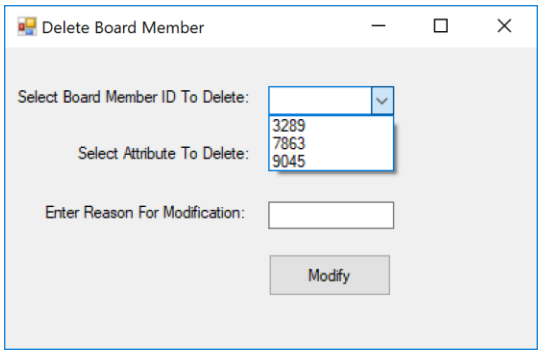


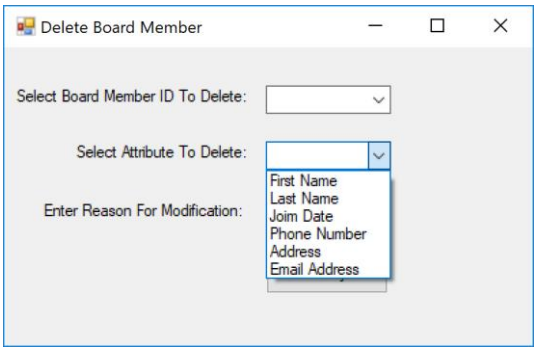
UC15-17 Edit, Modify and Delete Board Member Data

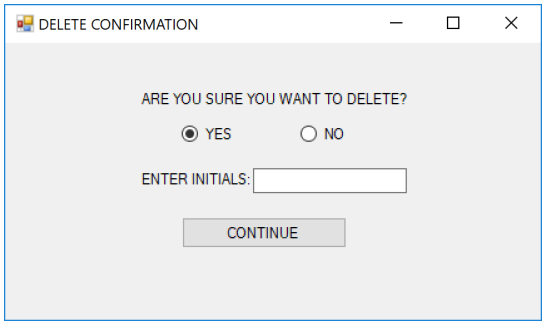




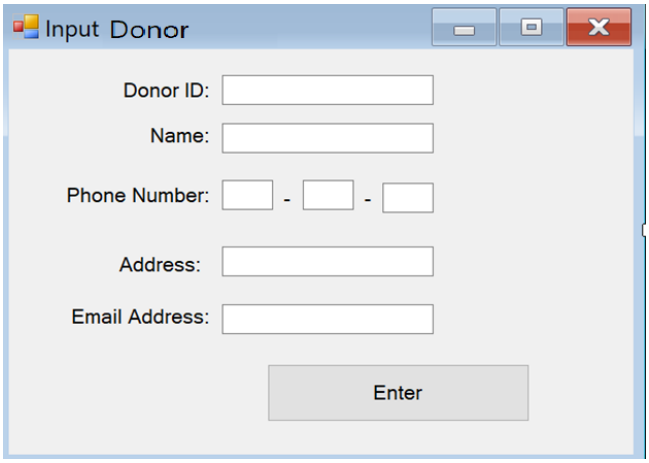


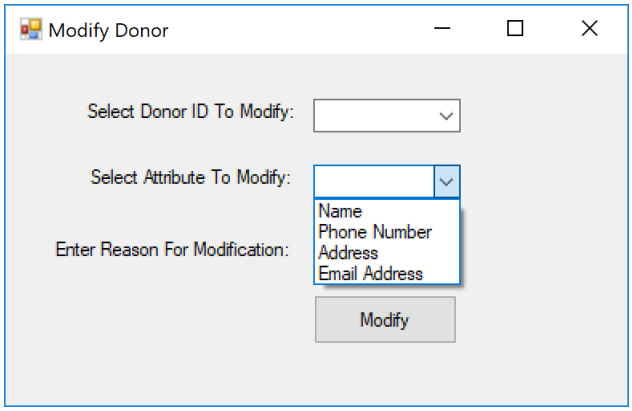


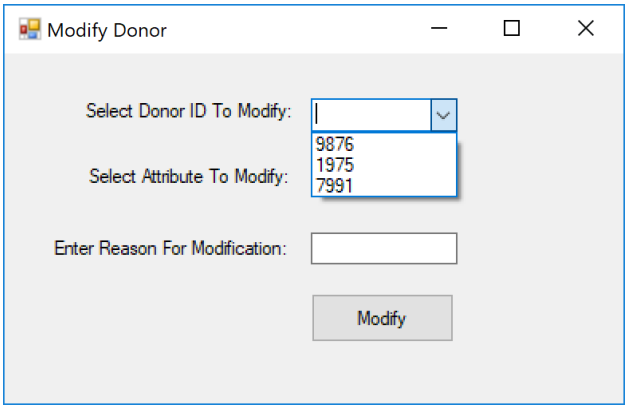


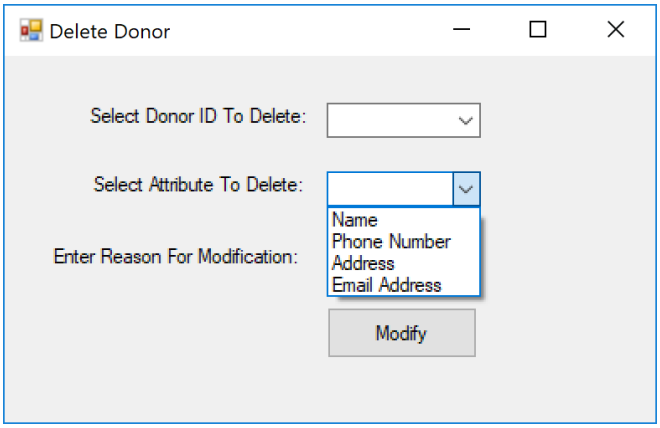


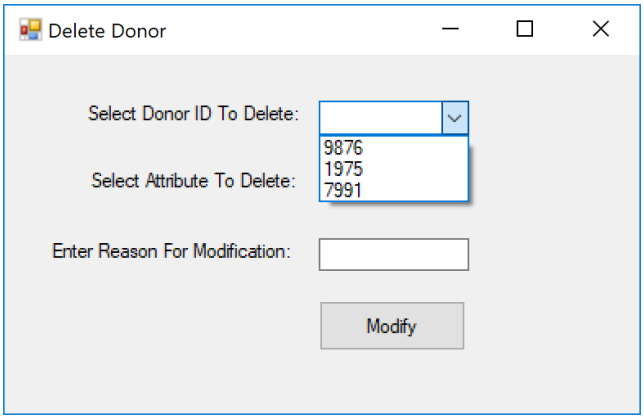
UC18-20 Edit, Modify and Delete Donor Data

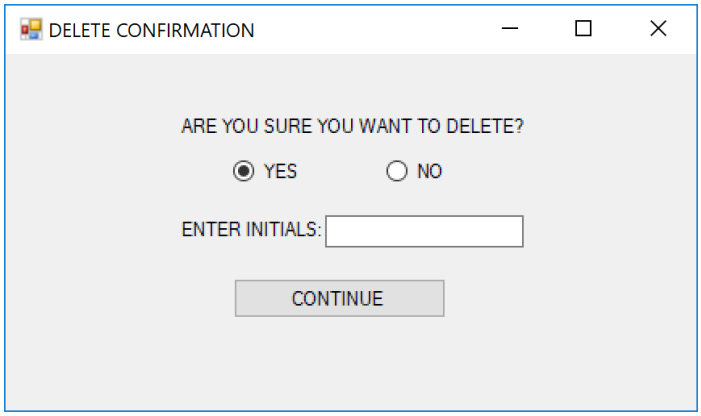




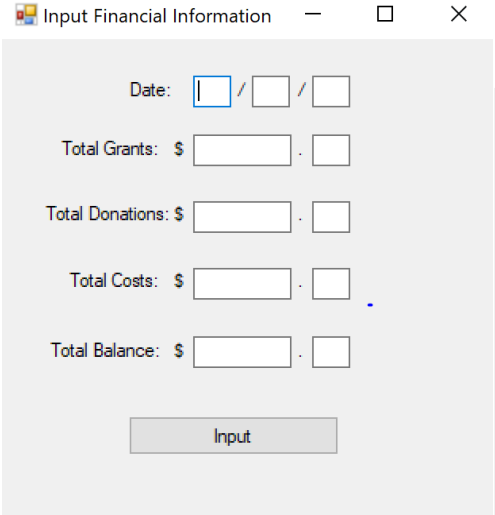


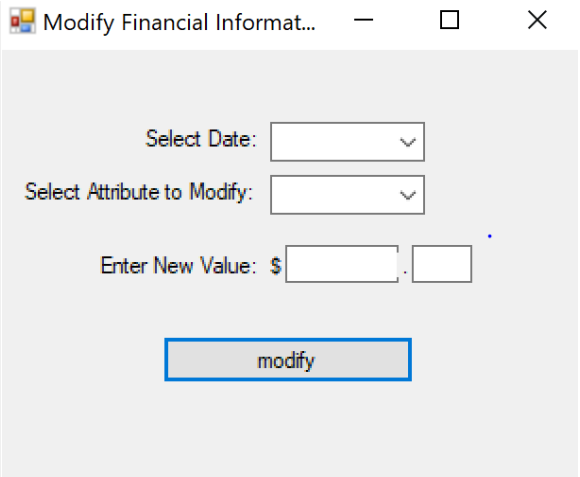


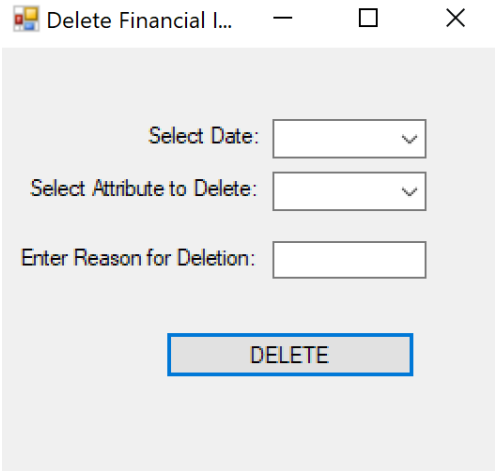


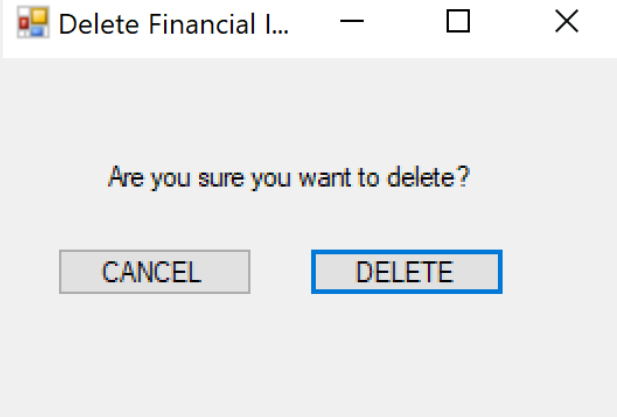


UC21-23 Input, Modify and Delete Financial data

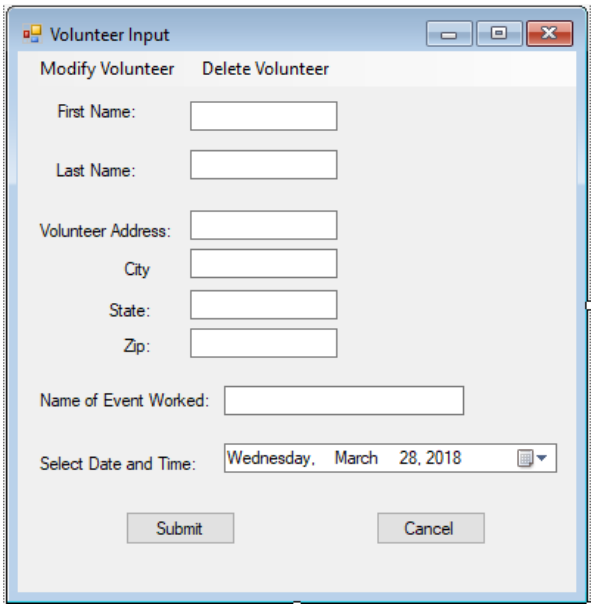


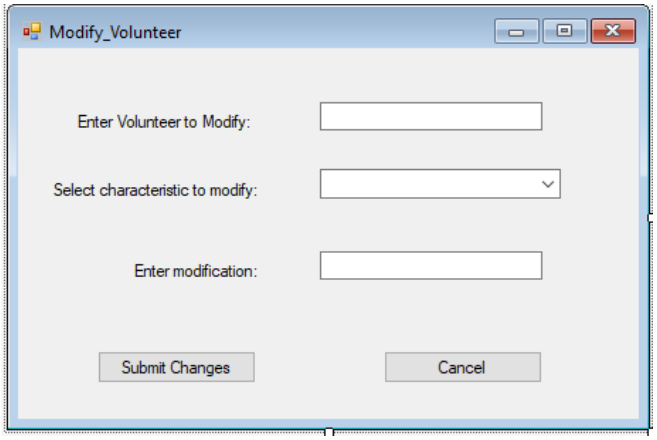


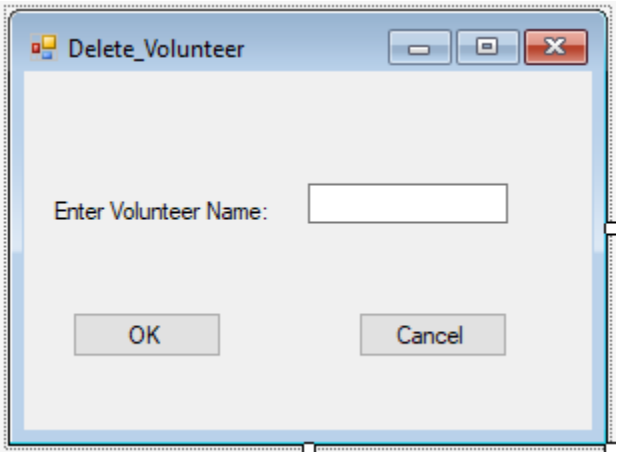




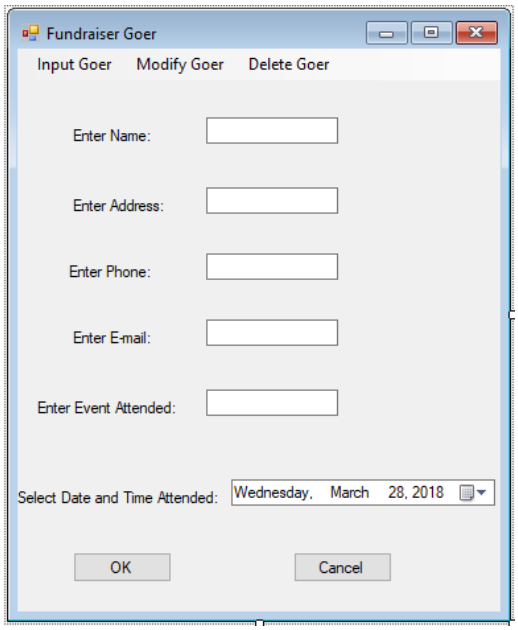
UC24-26Edit, Modify and Delete Volunteer Data

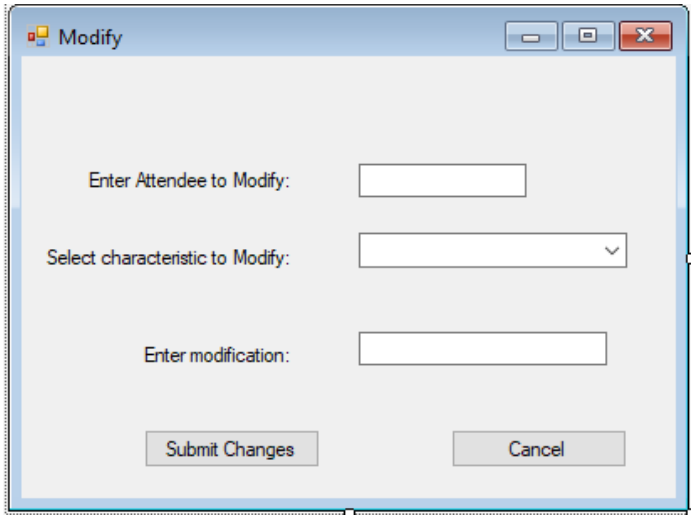


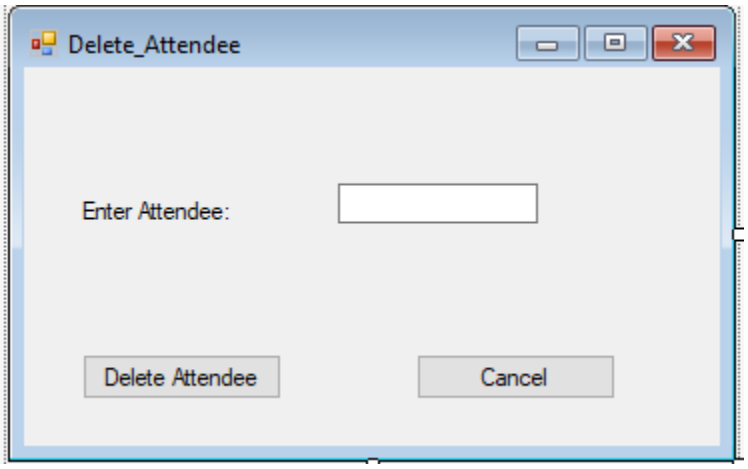




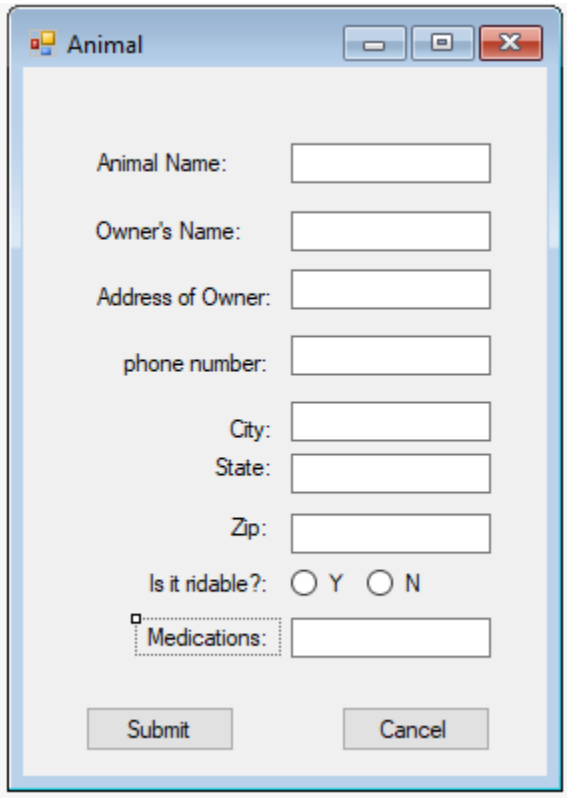
UC27-29: Input, Modify and Event-Goer Data

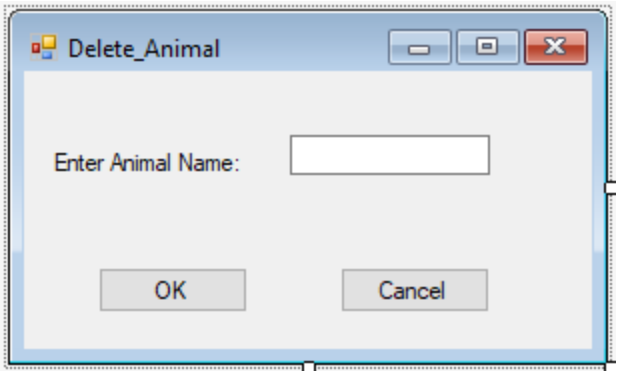


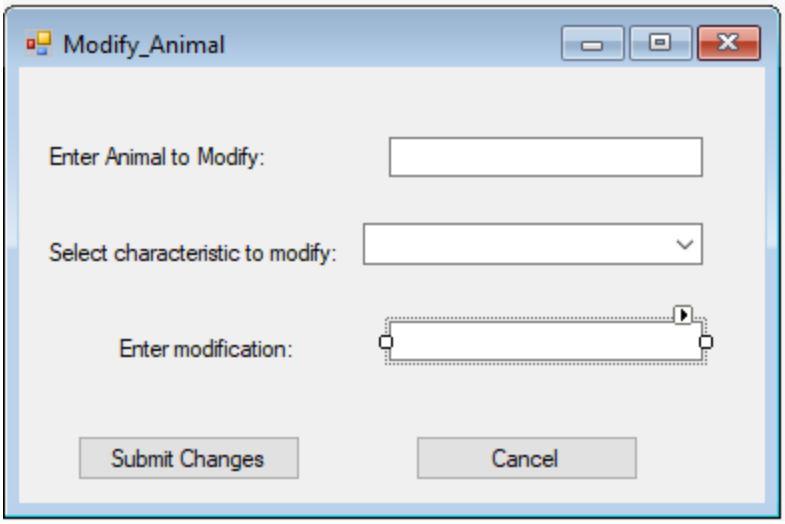




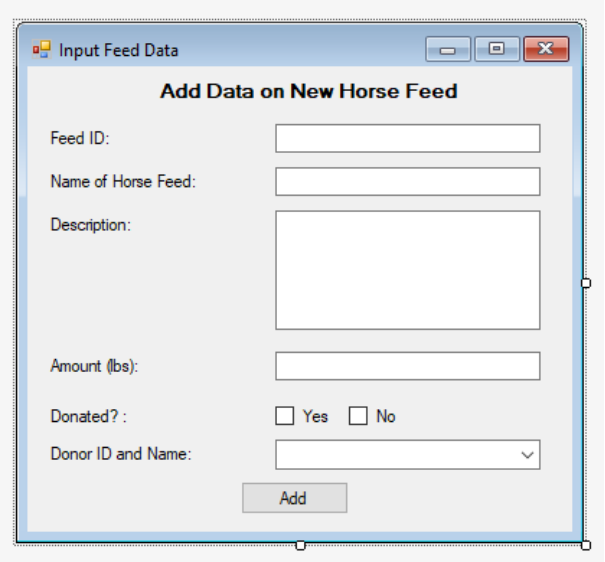
UC29-31: Input, Modify and Delete Animal data

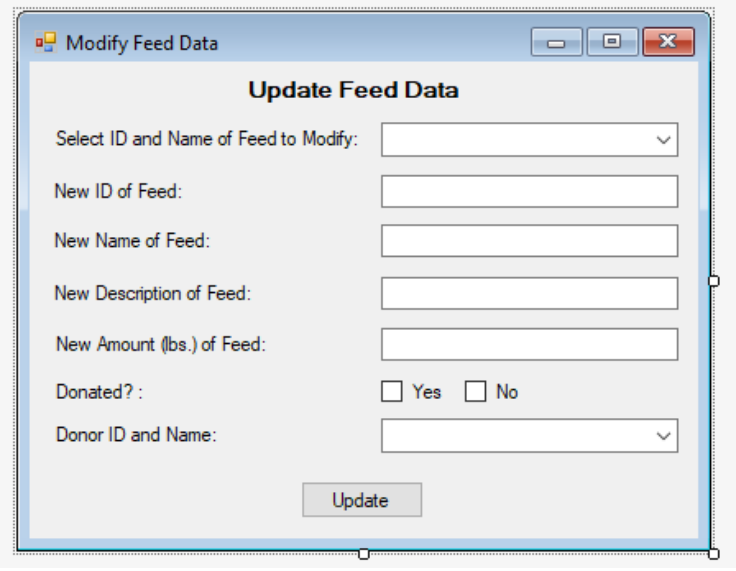


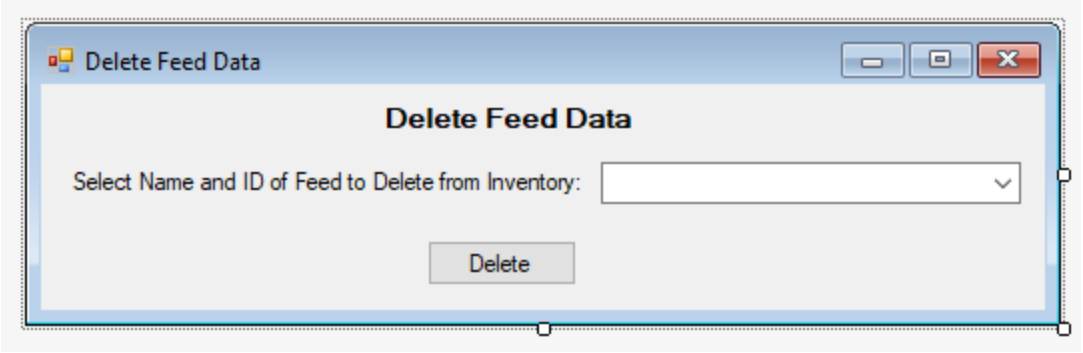




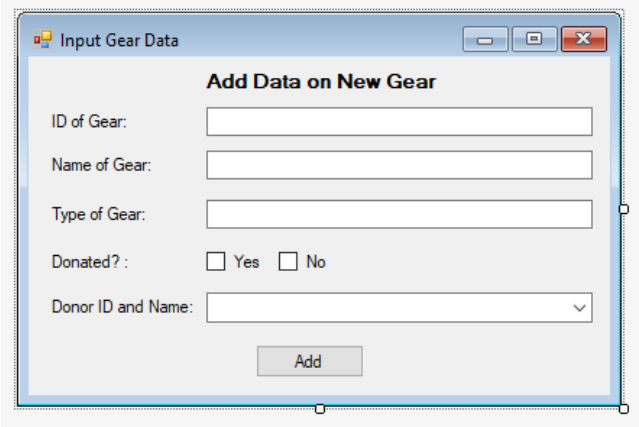
UC33-35: Input, Modify and Delete Feed data

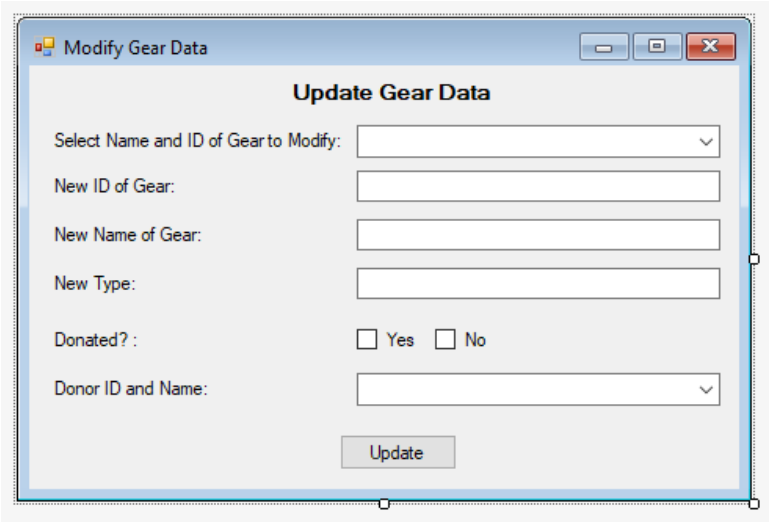


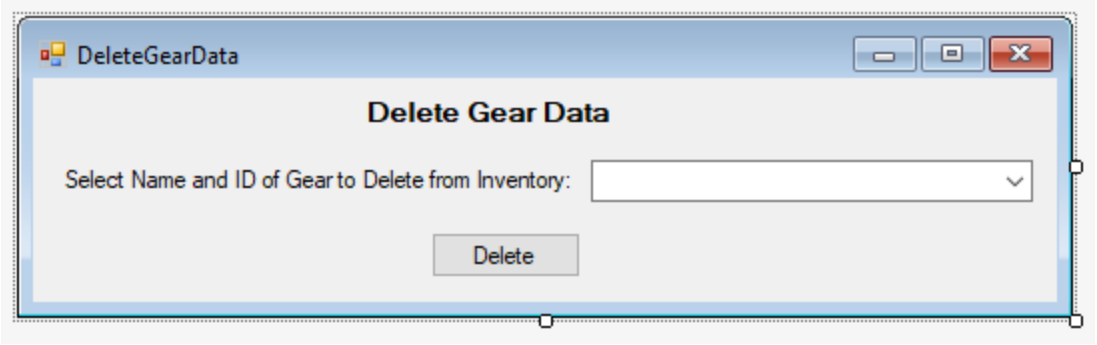




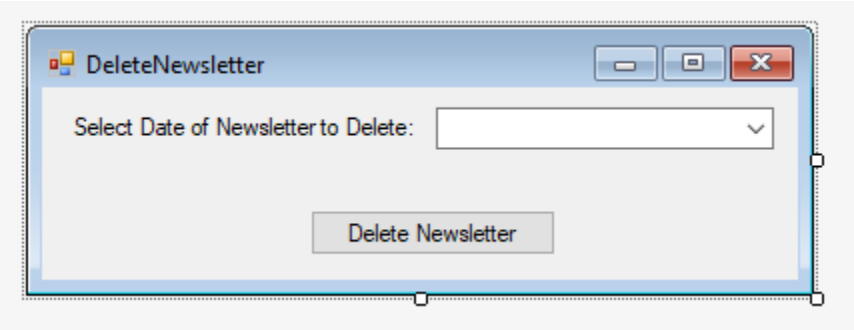
UC36-38: Input, Modify and Gear Feed data



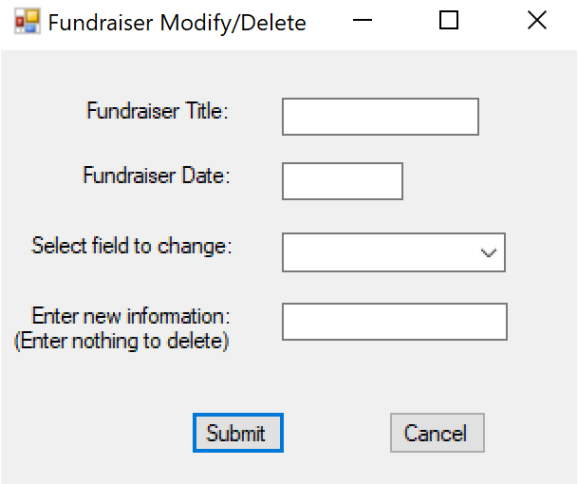


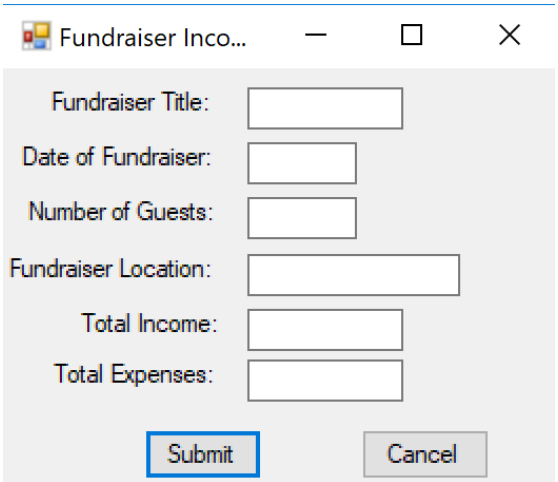


UC 39 -41 Post, Modify, Delete Newsletter

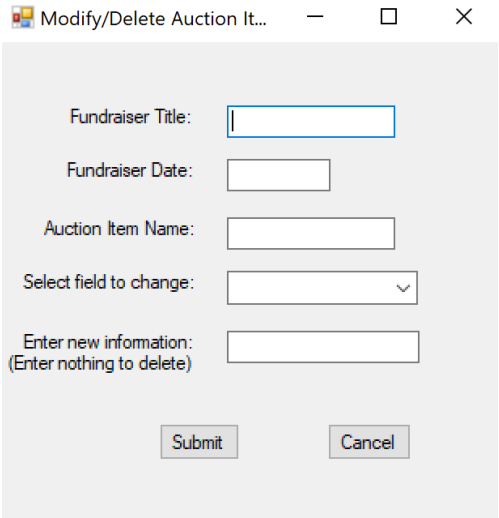


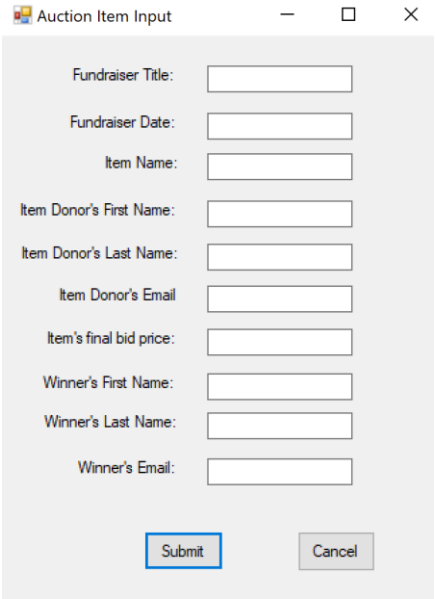
UC 42-44: Input, Modify and Delete Fundraiser Income



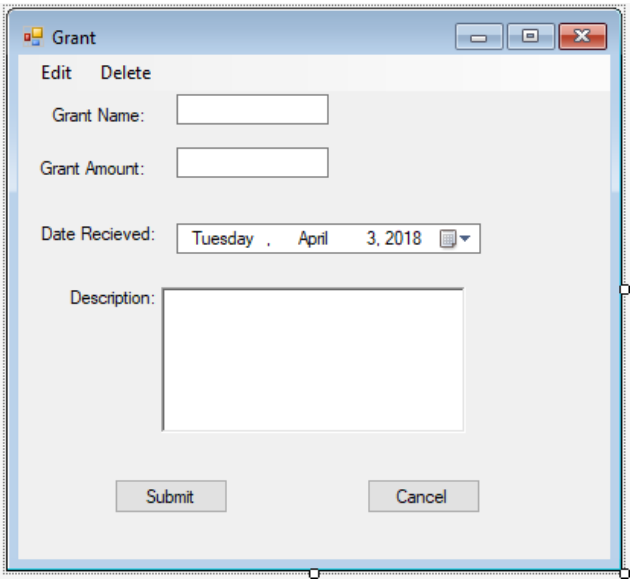


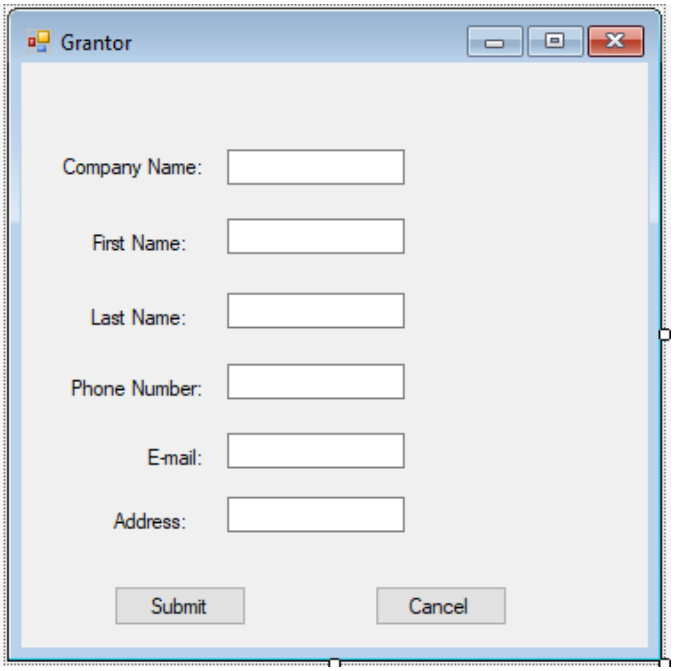
UC45-47: Input, Modify and Auction Items





UC48-53: Input, Modify and Delete Data on Grants and Grantors





UC 54-55: Member and Admin Login