

ELABORATION SPECIFICATION

KY HBPA IT Project



# System Requirements

## Narrative

The following lists cover the functional and non-functional requirements that must be satisfied by the deliverable. The functional requirements are specific to the application / website itself. They cover issues related to user transactions, tools, and data maintenance. The non-functional requirements cover issues outside the main application such as security and backups.

## Functional Requirements

1. The website will process transactions electronically.
2. The website will be able keep track of members through a back-end database.
3. The website will be able to associate documentation with users in the database.
4. The website will allow users to contact the KHBPA through email.
5. The website will contact users when they need to be contacted
6. The website will be directly integrated with the KHBPA social media accounts.
7. The website will allow users to make donations to KHBPA
8. The system will keep track of and allow for management of a calendar of events
9. The system will support management of news items
10. The system will keep a database of links to outside pages
11. The system will support storing and managing submitted claims
12. The system will store photos and videos
13. The system will track admin-created polls and support embedding them
14. The system will store and allow downloading of medication information documentation
15. The system will store and allow downloading of a license form
16. The system will store and allow downloading of a bill of sale form
17. The system will support online benefits form submission and store related data
18. The system will allow users to provide feedback through a question form with a free text input
19. The system will store and report contact info for the board of directors

## Non-Functional Requirements

1. The website content will be easily editable by employees of KHBPA.
2. The website will be visually distinct from similar organizations.
3. The system will be secure from outside attack
4. System data will be backed up every 24 hours with the last 3 backups being stored in separate locations
5. Data transferred to and from the server will be encrypted via SSL
6. Sensitive user data stored on the server will be encrypted with PGP
7. Permissions to application data will only be modifiable by an administrator
8. System must maintain full traceability of transactions
9. The system will maintain relational integrity in the database
10. The system will maintain 99% uptime
11. The system will be accessible from any location and at any time

# Use Case Diagram

## Narrative

## Diagram

# Trace Matrix Narrative

The trace matrix below is designed to describe the relationship between each system requirement and each use case. More specifically it describes how each use case satisfies the full list of system requirements. An ‘X’ occurs at the intersection of the requirement and each use case involved in satisfying the requirement

# Trace Matrix

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Requirements** | **Creates account** | **Makes a Donation** | **Maintains Calendar of Events** | **Posts a News Item** | **Posts a Link** | **Submits a Claim** | **Uploads a Picture** | **Creates a Poll** | **Votes on a Poll** | **Requests Medication Info** |
| **The website will process transactions electronically.** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** |
| **The website will be able keep track of members through a back-end database.** | **X** |  |  |  |  |  |  |  |  |  |
| **The website will be able to associate documentation with users** | **X** |  |  |  |  | **X** |  |  |  |  |
| **The website will allow users to make donations to KHBPA** |  | **X** |  |  |  |  |  |  |  |  |
| **The system will keep track of and allow for management of a calendar of events** |  |  | **X** |  |  |  |  |  |  |  |
| **The system will support management of news items** |  |  |  | **X** |  |  |  |  |  |  |
| **The system will keep a database of links to outside pages** |  |  |  |  | **X** |  |  |  |  |  |
| **The system will support storing and managing submitted claims** |  |  |  |  |  | **X** |  |  |  |  |
| **The system will store photos and videos** |  |  |  |  |  |  | **X** |  |  |  |
| **The system will track admin-created polls and support embedding them** |  |  |  |  |  |  |  | **X** | **X** |  |
| **The system will store and allow downloading of medication information documentation** |  |  |  |  |  |  |  |  |  | **X** |

# Trace Matrix (cont..)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Requirements** | **Requests Bill of Sale** | **Signs In** | **Requests Profile Info** | **Edits Profile Info** | **Requests Calendar of Events** | **Requests Photo Gallery** | **Submits Benevolence Form** | **Provides Feedback** | **Requests Contact Info** | **Contact KY HBPA** |
| **The website will process transactions electronically.** | X | X | X | X | X | X | X | X | X | X |
| **The website will be able keep track of members through a back-end database.** |  | X | X | X |  |  |  |  |  |  |
| **The website will be able to associate documentation with users** |  | X | X | X |  |  | X |  |  |  |
| **The website will allow users to contact the KHBPA through email.** |  |  |  |  |  |  |  |  | X | X |
| **The system will keep track of and allow for management of a calendar of events** |  |  |  |  | X |  |  |  |  |  |
| **The system will store photos and videos** |  |  |  |  |  | X |  |  |  |  |
| **The system will store and allow downloading of a bill of sale form** | X |  |  |  |  |  |  |  |  |  |
| **The system will support online benefits form submission and store related data** |  |  |  |  |  |  | X |  |  |  |
| **The system will allow users to provide feedback** |  |  |  |  |  |  |  | X |  | X |
| **The system will store and report contact info for the board of directors** |  |  |  |  |  |  |  |  | X | X |

|  |  |  |
| --- | --- | --- |
| **Requirements** | **Search News Archive** | **Send Newsletter** |
| **The website will process transactions electronically.** | X | X |
| **The website will contact users when they need to be contacted** |  | X |

# Use Cases

## **Use Case Specification: Creates Account**

### Brief Description

A user creates an account by providing personal information, an email address, and a password. By doing so this user becomes a member of the KHBPA and gains the benefits of membership (access to members-only features).

### Preconditions:

None

### Main Flow

1. User selects “Join KY HBPA” action
2. System serves create account screen
3. User enters First Name
4. User enters Last Name
5. User enters Stable, Corp., or Farm Name
6. User enters Managing Partner
7. User enters Address
8. User enters City
9. User selects State
10. User enters Zip Code
11. User enters Primary Phone Number
12. User enters E-mail address
13. User enters Password
14. User selects best description of themselves from: “Owner”, “Trainer”, or “Owner & Trainer”
15. User enters KRS License #
16. User digitally signs form
17. User agrees to terms
18. User clicks “Join Now” button
19. System validates user inputs
20. System stores form values in database

### Postconditions

1. A new account has been created

### Alternative Flows

1. InvalidEmailAddress
2. FormNotDigitallySigned
3. TermsNotAgreedTo

### **Alternative Flow: InvalidEmailAddress**

#### Brief Description

The system marks the email address field as invalid

#### Primary Actor

User

#### Preconditions

1. The user has entered an invalid email address

#### Alternative Flow

1. The alternative flow begins after step 19 of the main flow
2. The system informs the user that they have entered an invalid email address by adding a red mark to the email address field.

#### Postconditions

None

### **Alternative Flow: FormNotDigitallySigned**

#### Brief Description

The system marks the digital signature field as invalid

#### Primary Actor

User

#### Preconditions

1. The user has failed to digitally sign the form

#### Alternative Flow

1. The alternative flow begins after step 19 of the main flow
2. The system informs the user that they have failed to digitally sign the form by adding a red mark to the digital signature field.

#### Postconditions

None

### **Alternative Flow: TermsNotAgreedTo**

#### Brief Description

The system marks the terms agreement field as invalid

#### Primary Actor

User

#### Preconditions

1. The user has failed to agree to the terms and conditions

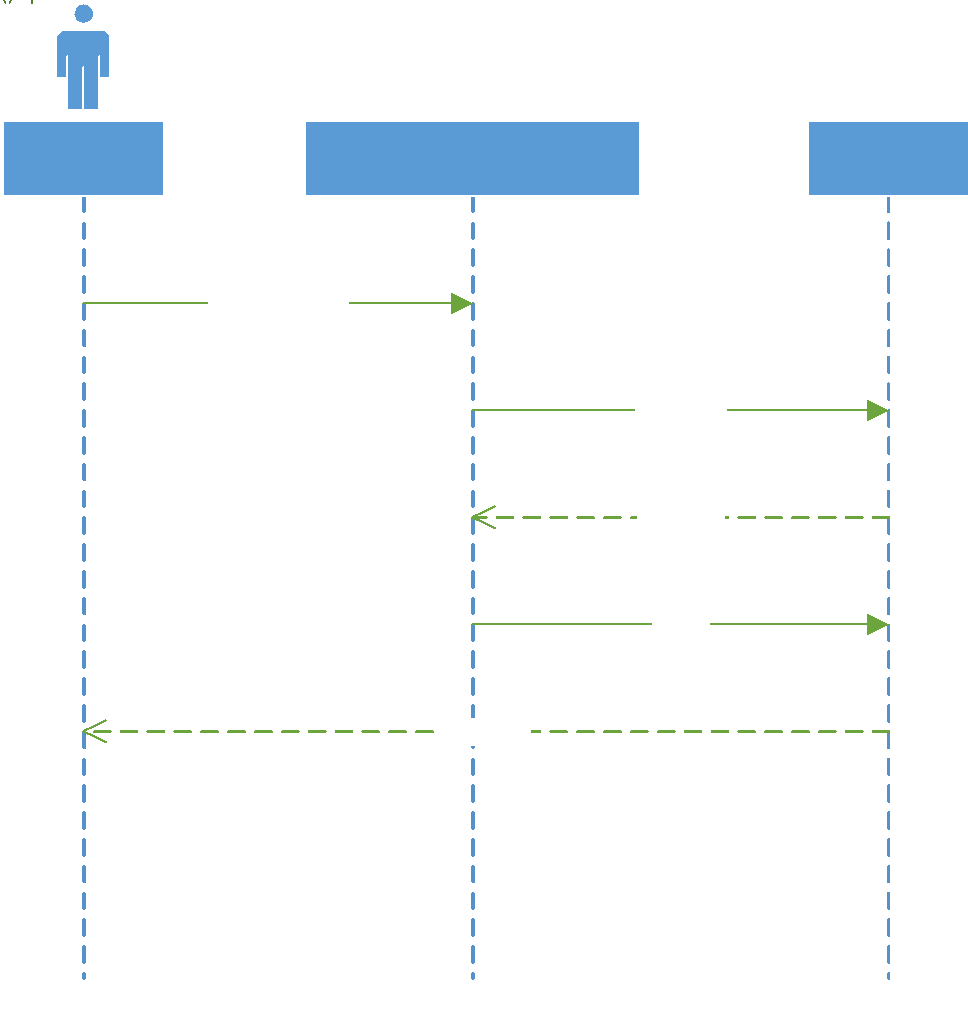
#### Alternative Flow

1. The alternative flow begins after step 19 of the main flow
2. The system informs the user that they have failed to agree to the terms and conditions by adding a red mark to the terms agreement field

#### Postconditions

None

### Sequence Diagram



## Use Case Specification: Makes a Donation

### Brief Description

A user has decided that they would like to contribute financially to the KHBPA, so they will make a one-time or recurring donation. To do so they will enter PayPal or credit card information along with the amount they would like to donate and then confirm the donation

### Primary Actors

User

### Preconditions

None

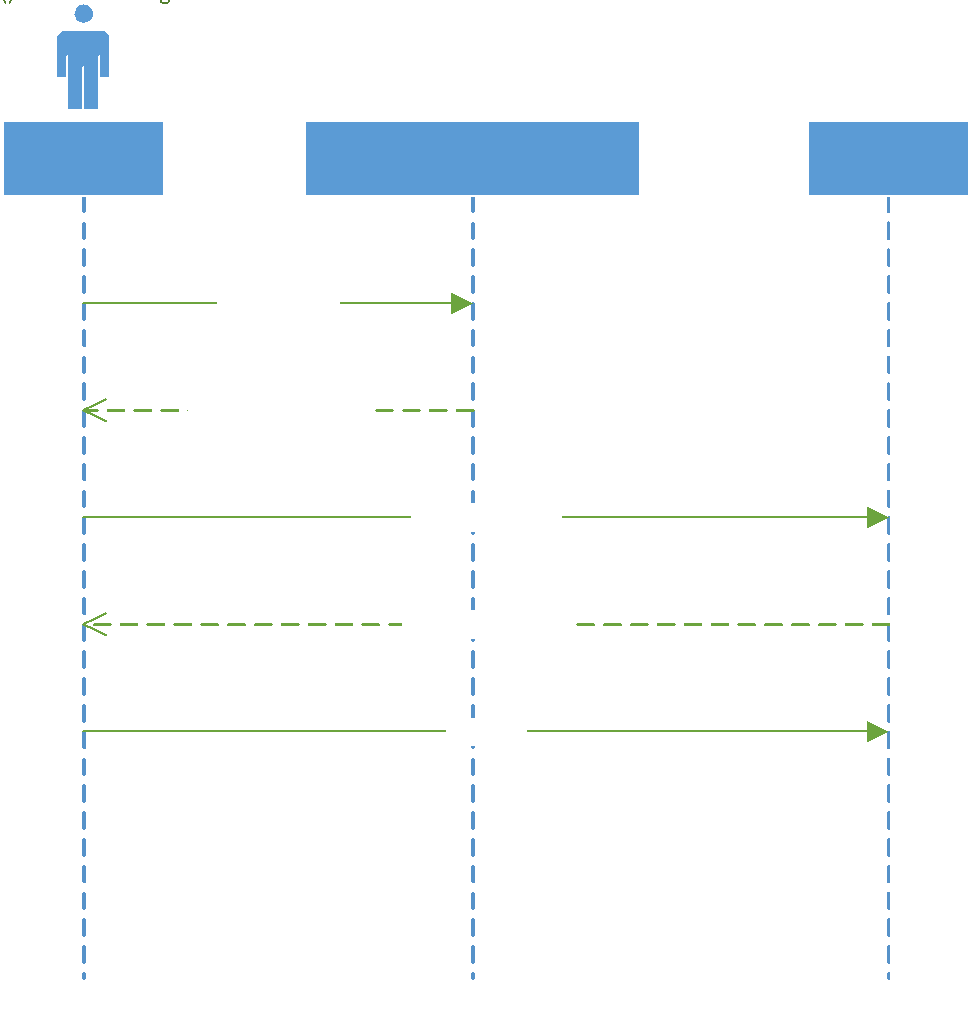
### Main Flow

1. User clicks “Donate” button
2. System serves donate screen
3. User selects donation type
4. User selects payment method
5. User clicks “Next”
6. System redirects user to payment portal
7. User enters payment information
8. User submits payment
9. Payment provider accepts payment
10. Payment provider redirects user to “Thank you” page in KY HBPA domain

### Postconditions

A donation has been processed to KY HBPA

### Sequence Diagram



## Use Case Specification: Posts a News Item

### Brief Description

An administrator would like to add a news item to the feed so they choose a title and text. Optionally they may also include a link, if the news item is posted on another page. Once this information is submitted, the news item is created in the feed with a date time stamp available for all to see.

### Primary Actors

Administrator

### Preconditions

Administrator is logged in

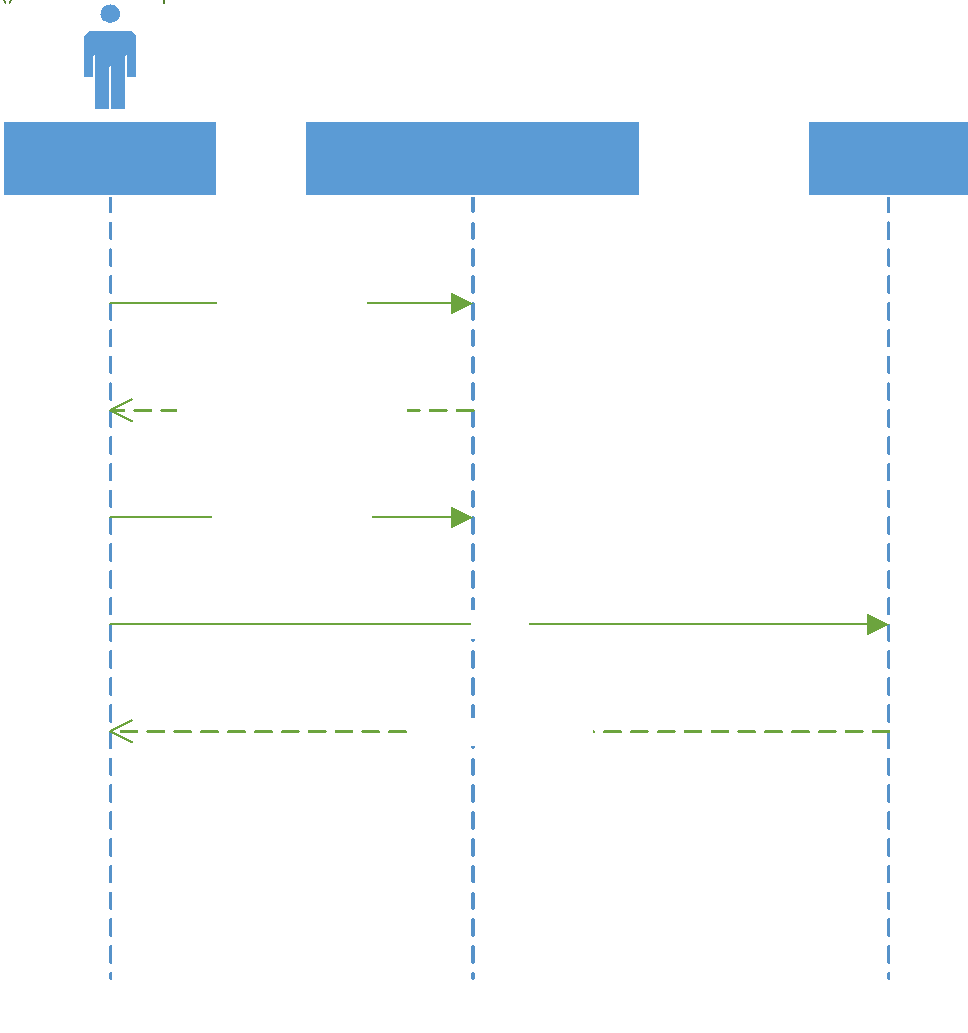
### Main Flow

1. Administrator clicks News link
2. System serves news page
3. User clicks “Create Post” button
4. User enters title
5. User enters link address, if any
6. User uploads related picture
7. User enters text description
8. User clicks “submit”
9. System validates user inputs
10. System adds news item to database
11. System returns success message

### Postconditions

1. A news item has been created

### Sequence Diagram



## Use Case Specification: Submits a Claim

### Brief Description

A member would like to submit a claim for a horse that has been injured or euthanized on the track. After clicking a submit claim button they are directed to a claim form to fill out. Once filled out, they submit the form to KHBPA where it waits for approval.

### Primary Actors

Member

### Preconditions

Member is logged in

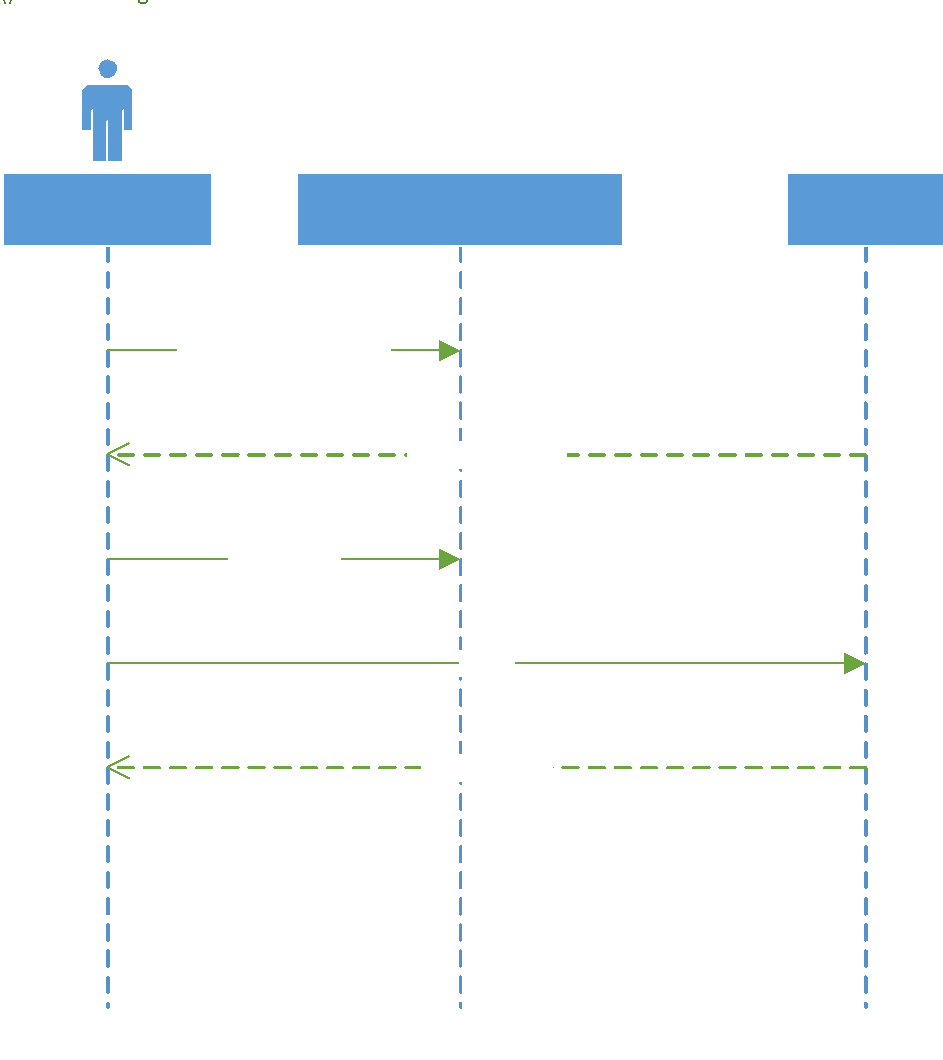
### Main Flow

1. Member clicks "Submit Claim" link
2. System serves claim page
3. Member enters owner's name
4. Member enters trainer's name
5. Member uploads veterinarian-signed statement of euthanasia
6. Member submits form
7. System stores claim in database

### Postconditions

A claim has been created

### Sequence Diagram



## Use Case Specification: Signs In

### Brief Description

A member signs in to the website using their e-mail address and password chosen when they created their account / signed up for the HBPA. This gives them access to all members-only sections and features of the website.

### Primary Actors

Member

### Preconditions

None

### Main Flow

1. Member clicks "Sign In" button
2. System serves sign in page
3. Member enters e-mail address
4. Member enters password
5. Member clicks "Sign In" button
6. System authenticates credentials
7. System updates user signed in status
8. System redirects user to previous page

### Postconditions

The user has been signed in

### Alternative Flows

1. InvalidEmailAddress
2. InvalidPassword

### **Alternative Flow: InvalidEmailAddress**

#### Brief Description

The system marks the email address field as invalid

#### Primary Actor

Member

#### Preconditions

1. No account exists for the specified e-mail address

#### Alternative Flow

1. The alternative flow begins after step 6 of the main flow
2. The system informs the user that they have entered an invalid email address by adding a red mark to the email address field.

#### Postconditions

None

### **Alternative Flow: InvalidPassword**

#### Brief Description

The system marks the password field as invalid

#### Primary Actor

Member

#### Preconditions

1. The specified password is incorrect

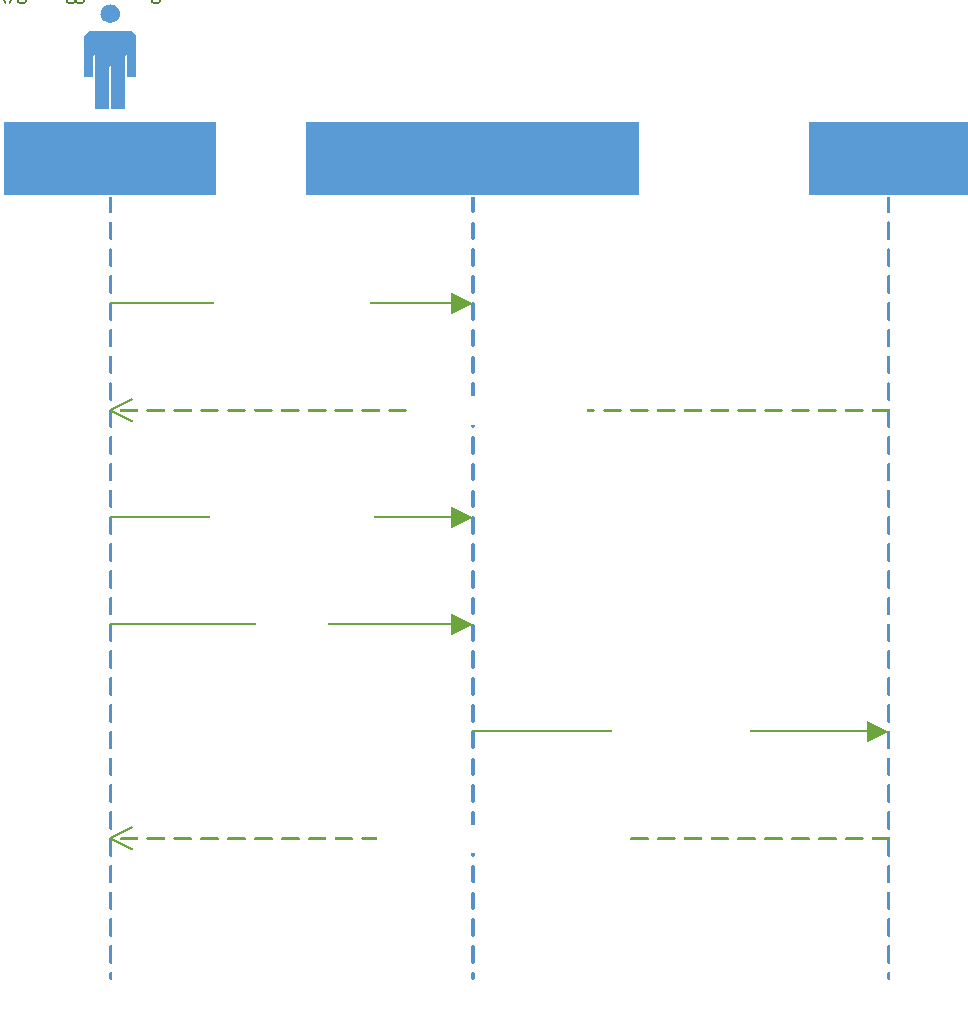
#### Alternative Flow

1. The alternative flow begins after step 6 of the main flow
2. The system informs the user that they have entered an invalid password by adding a red mark to the password field.

#### Postconditions

None

### Sequence Diagram



# Class Diagram

## Explanation

The class diagram is a visual representation of the classes that will be used to build the system. Below, we model the classes and their relationships to visually demonstrate the interconnectivity of this system.

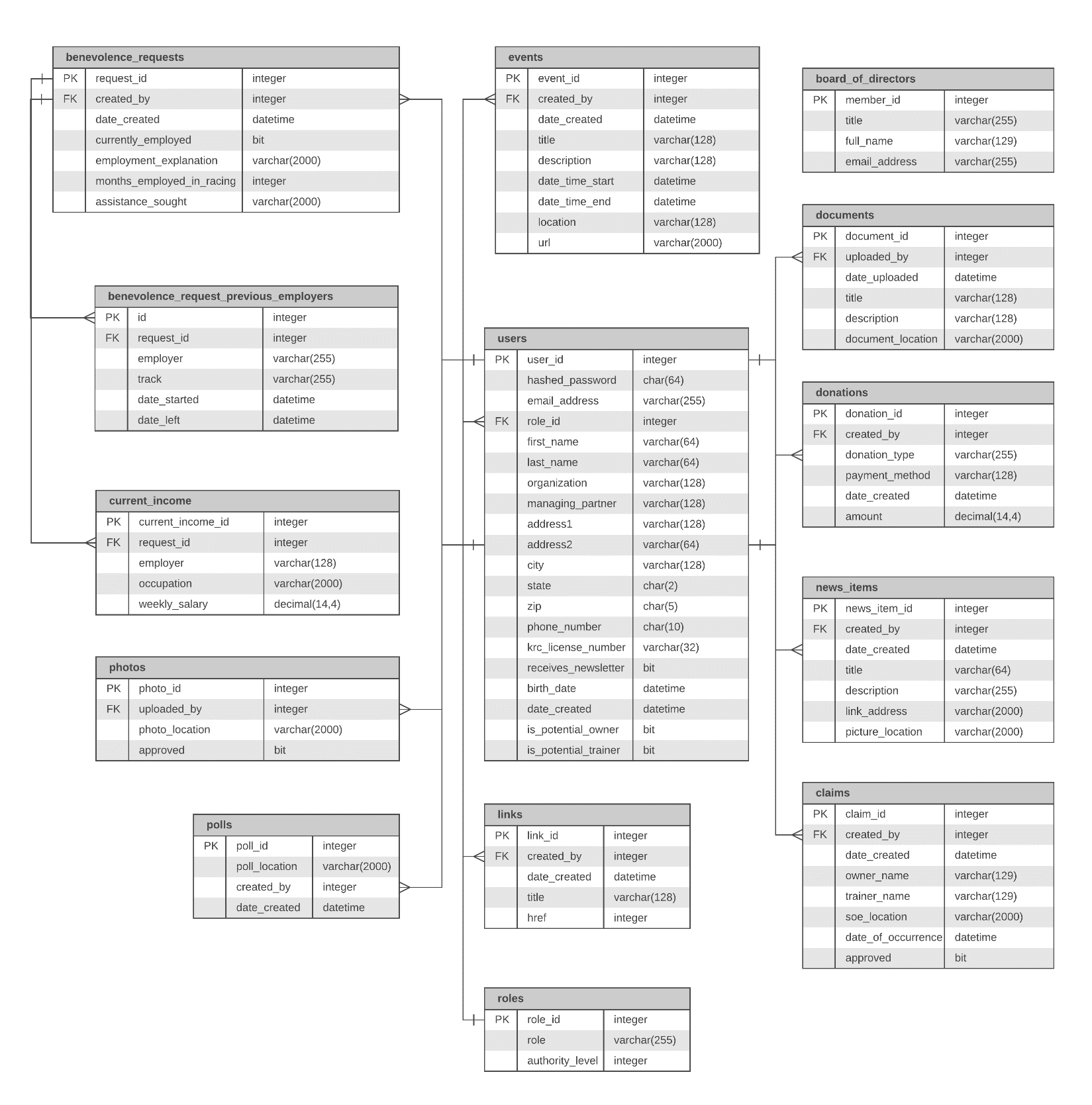
## Diagram

# Database Design and Data Definitions

## Explanation

The database design below is in the third normal form (abbreviated 3NF). In simple terms, this means that each field in each table depends directly on the primary key and only on the primary key. Attributes related to each atomic entity are stored in their own tables. For example all users have roles, and roles have authority levels. So you could say that all users have authority levels, but the authority level is dependent on the role, **not** on the user. For this reason *roles* is its own table, and each user has a role id. Doing this eliminates transitive dependence which is very important for ensuring relational integrity and data consistency. This entity relationship diagram was created from the class diagram through analysis of each classes’ attributes and methods. Each attribute that requires long term persistence has a corresponding field in the ERD. So each attribute is roughly related to each field and each class is roughly related to each table.

## Entity Relationship Diagram



## Data Dictionary

### benevolence\_request\_previous\_employers

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **KEY** | **ATTRIBUTE NAME** | **DESCRIPTION** | **DATA TYPE** | **SIZE** |
| PK | id | auto-numbered unique identifier for a benevolence\_request\_previous\_employers record | integer |  |
| FK | request\_id | unique identifier of the related benevolence request | integer |  |
|  | employer | name of previous employer | varchar | 255 |
|  | track | name of track where requestor was previously employed | varchar | 255 |
|  | date\_started | the date employment started with this employer | datetime |  |
|  | date\_left | the date employment ended with this employer | datetime |  |

### benevolence\_requests

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **KEY** | **ATTRIBUTE NAME** | **DESCRIPTION** | **DATA TYPE** | **SIZE** |
| PK | request\_id | auto-numbered unique identifier for a benevolence request | integer |  |
| FK | created\_by | user\_id of benevolence request creator | integer |  |
|  | date\_created | date and time of benevolence request creation | datetime |  |
|  | currently\_employed | true if the requestor is currently employed, false otherwise | bit |  |
|  | employment\_explanation | text explanation of the reason the requestor is not employed, if currently\_employed is false | varchar | 2000 |
|  | months\_employed\_in\_racing | total number of months the requestor has been employed in racing | integer |  |
|  | assistance\_sought | the type of assistance the requestor is seeking | varchar | 2000 |

### board\_of\_directors

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **KEY** | **ATTRIBUTE NAME** | **DESCRIPTION** | **DATA TYPE** | **SIZE** |
| PK | member\_id | unique identifier of a board member | integer |  |
|  | title | board member's title | varchar | 255 |
|  | full\_name | board member's full name | varchar | 129 |
|  | email\_address | board member's email address | varchar | 255 |

### claims

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **KEY** | **ATTRIBUTE NAME** | **DESCRIPTION** | **DATA TYPE** | **SIZE** |
| PK | claim\_id | unique identifier of a claim | integer |  |
| FK | created\_by | user\_id of the claim creator | integer |  |
|  | date\_created | date and time of claim creation | datetime |  |
|  | owner\_name | name of horse owner involved in claim | varchar | 129 |
|  | trainer\_name | name of horse trainer involved in claim | varchar | 129 |
|  | soe\_location | statement of euthenasia location in UNC format | varchar | 2000 |
|  | date\_of\_occurrence | date and time of claim-relevant event occurence | datetime |  |
|  | approved | true if claim is approved, false otherwise | bit |  |

### current\_income

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **KEY** | **ATTRIBUTE NAME** | **DESCRIPTION** | **DATA TYPE** | **SIZE** |
| PK | current\_income\_id | auto-numbered unique identifier for a current\_income record | integer |  |
| FK | request\_id | unique identifier of the related benevolence request | integer |  |
|  | employer | name of current employer | varchar | 255 |
|  | occupation | title of current occupation | varchar | 2000 |
|  | weekly\_salary | weekly salary in dollars | decimal | 14,4 |

### documents

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **KEY** | **ATTRIBUTE NAME** | **DESCRIPTION** | **DATA TYPE** | **SIZE** |
| PK | document\_id | unique identifier of a document | integer |  |
| FK | uploaded\_by | user\_id of document uploader | integer |  |
|  | date\_uploaded | date and time document was uploaded | datetime |  |
|  | title | title of document | varchar | 255 |
|  | description | description of document | varchar | 128 |
|  | document\_location | location of document in UNC format ex: \\Server\Volume\File | varchar | 128 |

### donations

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **KEY** | **ATTRIBUTE NAME** | **DESCRIPTION** | **DATA TYPE** | **SIZE** |
| PK | donation\_id | unique identifier of a donation record | integer |  |
| FK | created\_by | user\_id of donator | integer |  |
|  | donation\_type | type of donation (one time, recurring, etc..) | varchar | 255 |
|  | payment\_method | method of payment (credit card, paypal) | varchar | 255 |
|  | date\_created | date and time donation was made | datetime |  |
|  | amount | donation amount | decimal | 14,4 |

### polls

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **KEY** | **ATTRIBUTE NAME** | **DESCRIPTION** | **DATA TYPE** | **SIZE** |
| PK | poll\_id | unique identifier of a poll | integer |  |
| FK | created\_by | user\_id of poll creator | integer |  |
|  | poll\_location | url of poll | varchar | 2000 |
|  | date\_created | date and time donation was made | datetime |  |

### events

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **KEY** | **ATTRIBUTE NAME** | **DESCRIPTION** | **DATA TYPE** | **SIZE** |
| PK | event\_id | unique identifier of an event record | integer |  |
| FK | created\_by | user\_id of event creator | integer |  |
|  | date\_created | date and time of event creation | datetime |  |
|  | title | title of event | varchar | 64 |
|  | description | description of event | varchar | 128 |
|  | date\_time\_start | start date and time of event | datetime |  |
|  | date\_time\_end | end date and time of event | datetime |  |
|  | location | location (address) of event | varchar | 128 |
|  | url | url of event website, if relevant | varchar | 2000 |

### news\_items

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **KEY** | **ATTRIBUTE NAME** | **DESCRIPTION** | **DATA TYPE** | **SIZE** |
| PK | news\_item\_id | unique identifier of a news item | integer |  |
| FK | created\_by | user\_id of the news item poster | integer |  |
|  | date\_created | date and time the news item was posted | datetime |  |
|  | title | title of the news item | varchar | 128 |
|  | description | text description of the news item | varchar | 128 |
|  | link\_address | URL the news item points to where relevant | varchar | 2000 |
|  | picture\_location | location of news item related picture in UNC format (where relevant) | varchar | 2000 |

### photos

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **KEY** | **ATTRIBUTE NAME** | **DESCRIPTION** | **DATA TYPE** | **SIZE** |
| PK | photo\_id | unique identifier of the photo | integer |  |
| FK | uploaded\_by | user\_id of the photo uploader | integer |  |
|  | photo\_location | location of the uploaded photo in UNC format ex: \\Server\Volume\File | varchar | 2000 |
|  | approved | true or false indicator of current approval status for a photo | bit |  |

### roles

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **KEY** | **ATTRIBUTE NAME** | **DESCRIPTION** | **DATA TYPE** | **SIZE** |
| PK | role\_id | unique identifier of a role | integer |  |
|  | role | name of the role (administrator, member) | varchar | 255 |
|  | authority\_level | numeric indicator of relative power level of a given role | integer |  |

### links

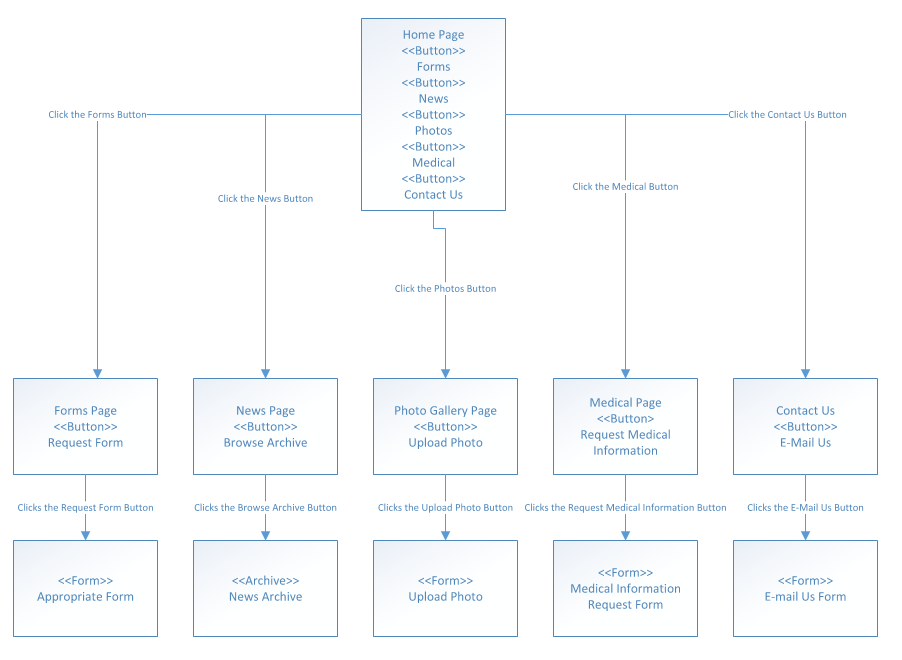
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **KEY** | **ATTRIBUTE NAME** | **DESCRIPTION** | **DATA TYPE** | **SIZE** |
| PK | link\_id | unique identifier of a link item | integer |  |
| FK | created\_by | user id of the link creator | integer |  |
|  | date\_created | date and time link was created | datetime |  |
|  | title | title of the link | varchar | 128 |
|  | href | URL reference the link points to | varchar | 2000 |

### users

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **KEY** | **ATTRIBUTE NAME** | **DESCRIPTION** | **DATA TYPE** | **SIZE** |
| PK | user\_id | auto-numbered unique identifier for a user | integer |  |
|  | hashed\_password | hashed password tied to user account | char | 64 |
|  | email\_address | email address of user | varchar | 255 |
|  | role\_id | user's role in system (member, administrator, etc…) | integer |  |
|  | first\_name | user's first name | varchar | 64 |
|  | last\_name | user's last name | varchar | 64 |
|  | organization | user's organization (stable, corporation, syndicate, farm) | varchar | 128 |
|  | managing\_partner | user's managing partner | varchar | 128 |
|  | address1 | user's street address | varchar | 128 |
|  | address2 | line 2 of user's address (apt / suite / bldg) | varchar | 64 |
|  | city | user's city | varchar | 128 |
|  | state | user's state | char | 2 |
|  | zip | user's zip code | char | 5 |
|  | phone\_number | user's phone number | char | 10 |
|  | krc\_license\_number | user's Kentucky racing commission license number | varchar | 32 |
|  | receives\_newsletter | true if the user has checked that they would like to receive the newsletter, false otherwise | bit |  |
|  | birth\_date | user's date of birth | datetime |  |
|  | date\_created | date user account was created | datetime |  |
|  | is\_potential\_owner | true if the user has indicated that they are a potential owner | bit |  |
|  | is\_potential\_trainer | true if the user has indicated that they are a potential trainer | bit |  |

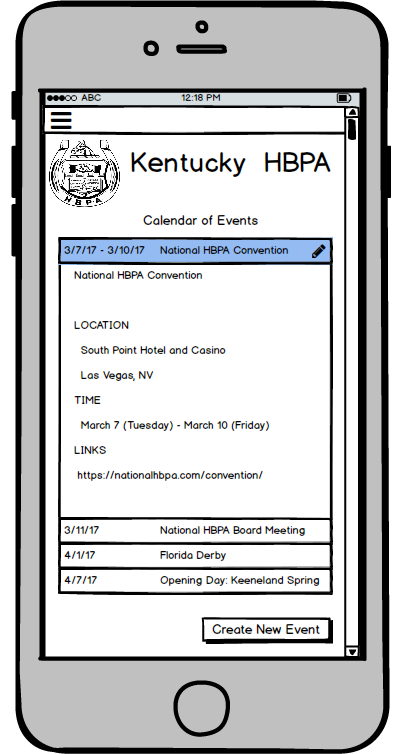
# User Interface

## Navigation Diagram

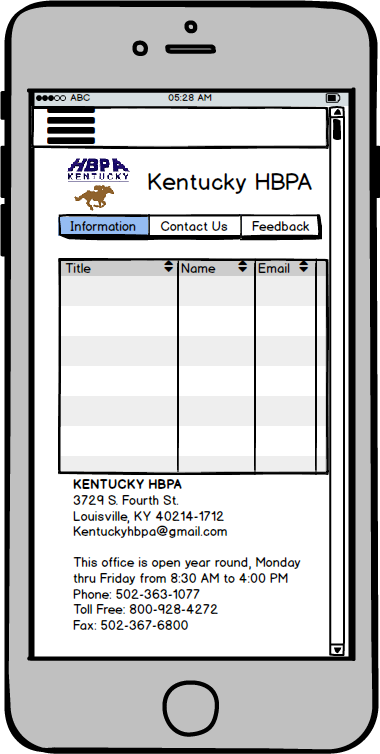


## Screen Layouts

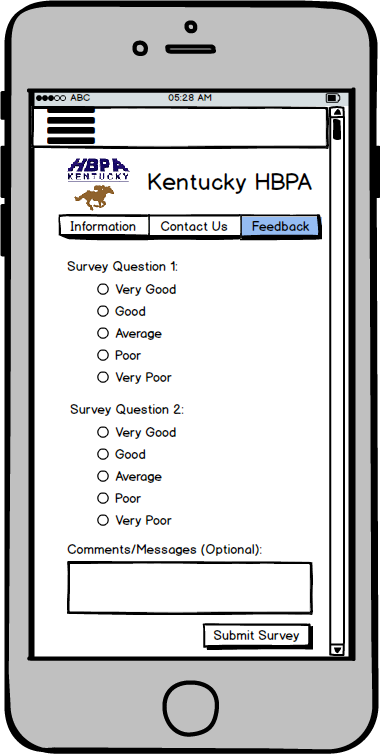
This following section will be featuring screen layouts, this is how we imagine the site to look when taking in and displaying information that either the user or admin gives. We divide these prototypes into two different kinds, reports which give information, and forms which need to be given information.



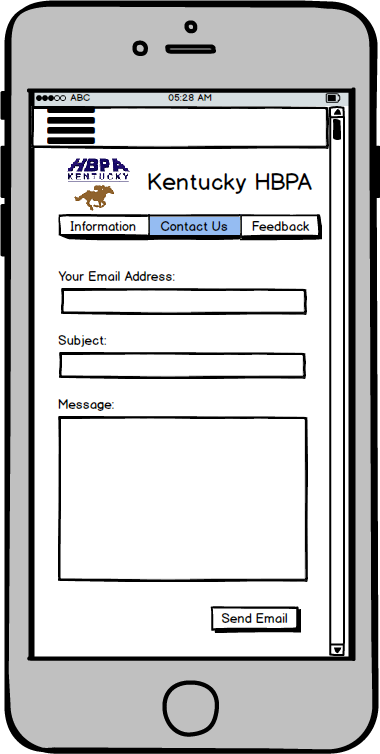
This is the layout we will use to display a calendar of events. In this prototype, we have already set up a fictitious event and filled in all needed areas. One of the big features that the Kentucky HBPA wanted was an easy to use calendar and we believe that this one will work just fine.



This will be what the contact us page will look like. This report will obviously be filled in with actual people from the Kentucky HBPA, but for now we have it as an unpopulated table. It will allow for easy lookup of Kentucky HBPA administrators.



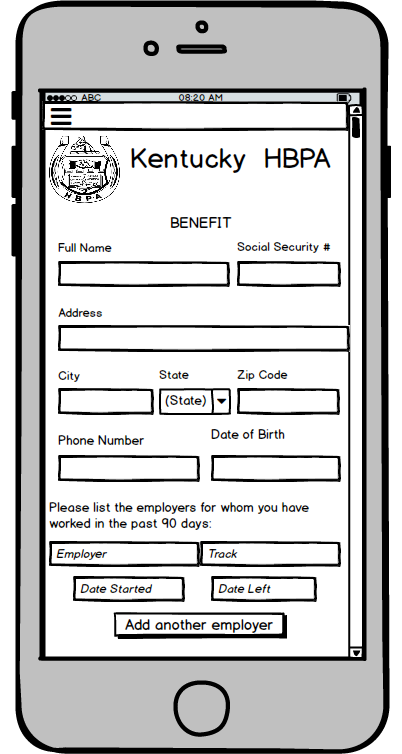
This is a form for giving feedback to the administrators about whatever it is you may be asking. This will be very similar to the form that will be used to poll your membership basis. Surveying the membership population was certainly one of the main points they stressed as well.



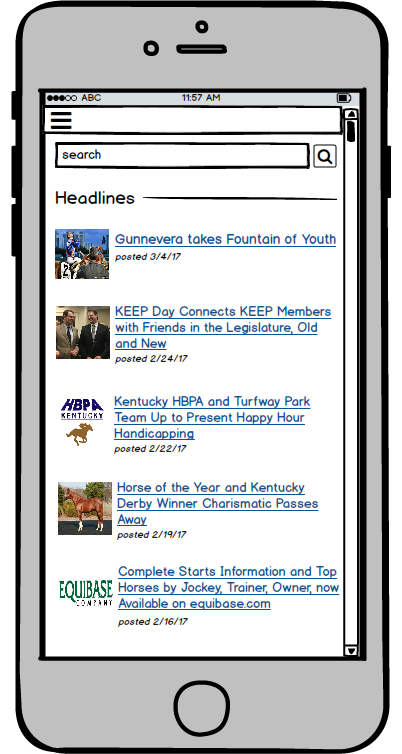
This will be another contact us, this time in a form instead of a report. This will allow a user to send an e-mail to the Kentucky HBPA and ask them any potential questions they may have.



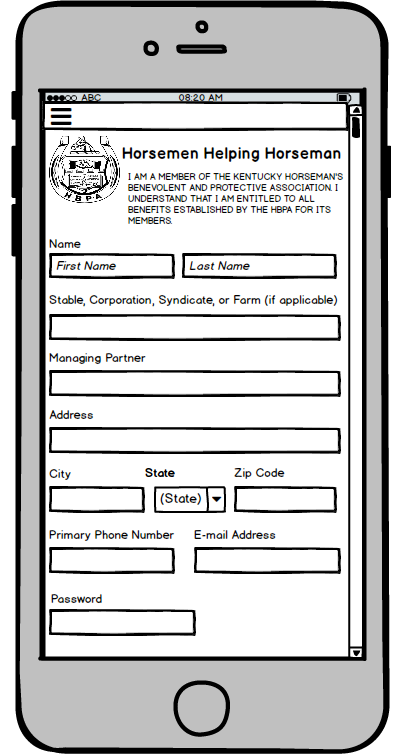
Here we have the create an event in form, unlike the previous prototype, this will be what we imagine the screen will look like when someone is creating an event. It will be simple enough, just pick dates and give it a title and description with a location.



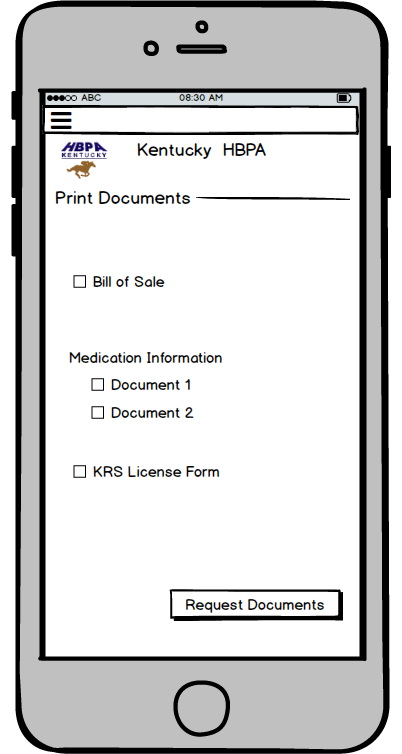
Here is the form layout for the submit benefits. In this view you can see what will be needed, such as your name, SSN, address and so on. We believe that with the ease of access and use, it will encourage more people to sign up with the Kentucky HBPA.



This will be our report view for when a user searches through the news and what we believe the screen should look like afterwards. This page allows for a clean view of whatever articles a person my need, as well as a sample picture to help discern what the article may be about.



This will be our form view for joining the Kentucky HBPA, once again requiring people to input the information in the following fields. This will allow for administrators to easily pull up all needed info once the form is completed and let the back-end database store the new account easily.

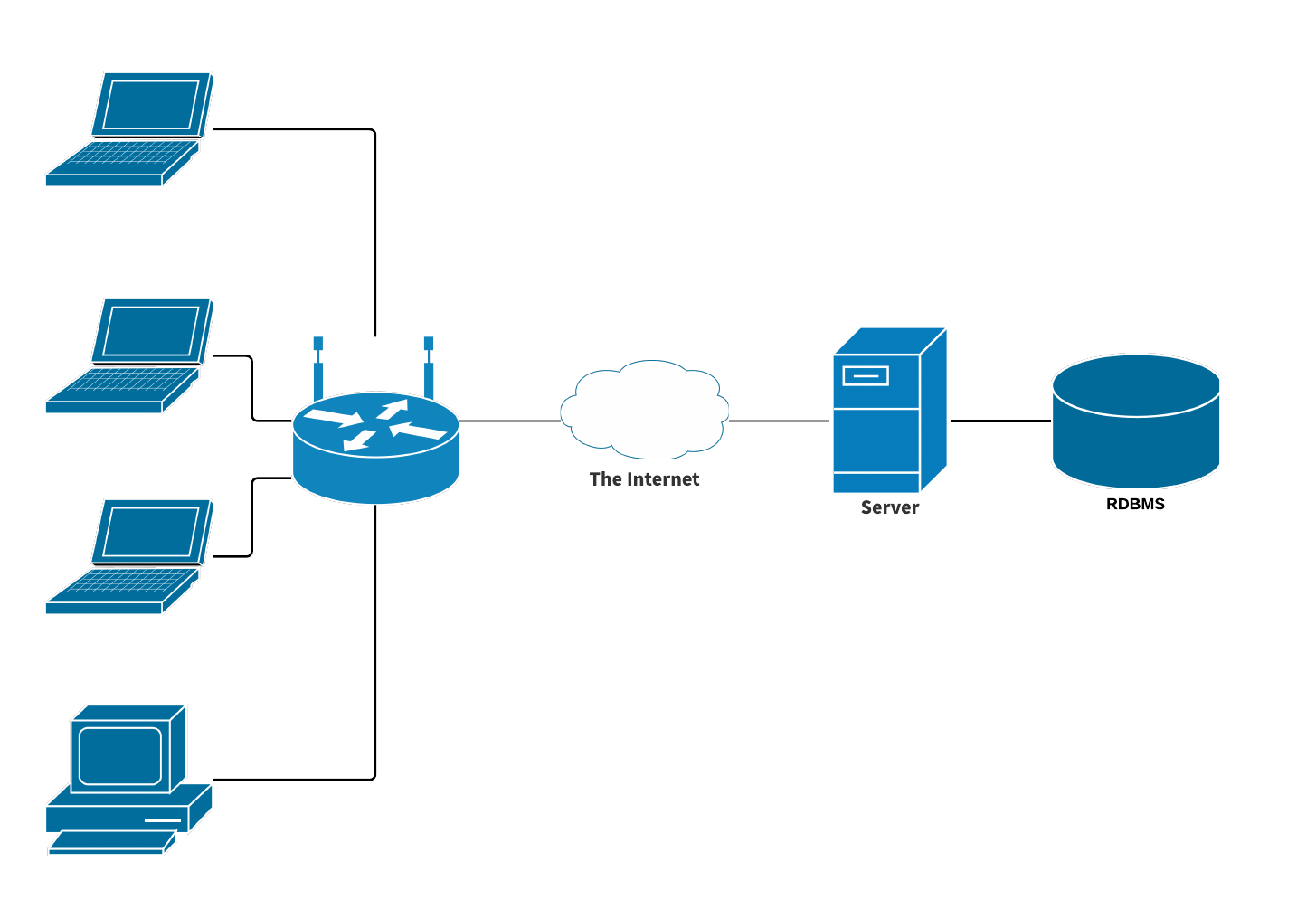


This will be our form view for when the members want to find some document or report and be able to print it out. Being able to select multiple ones to download lets the user just have one quick trip to the site where he or she can get all the documents they may need in on

# Physical Architecture Design

## Explanation

## Diagram



## System Software Specifications

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Specification** | **Standard Client** | **Standard Web Server** | **Standard Application Server** | **Standard Database Server** |
| **Operating System** | Windows | Linux | Linux | Linux |
|  | Microsoft Edge |  |  |  |
|  |  |  |  |  |
| **Special Software** | Office 365 | Apache | PHP | MySQL |
|  | MS OneDrive |  |  |  |
|  | Acrobat Reader |  |  |  |

# Procedures for Non-functional Requirements

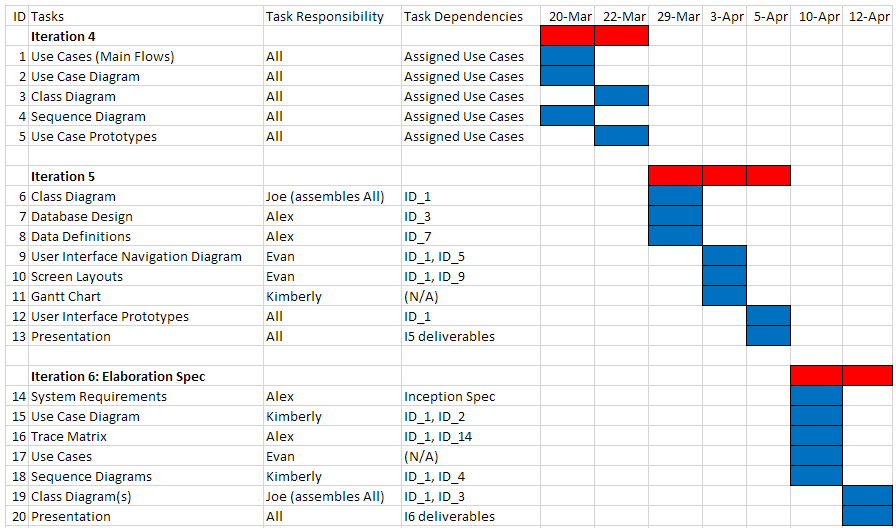
## List and Explanation

# Gantt Chart

## Narrative

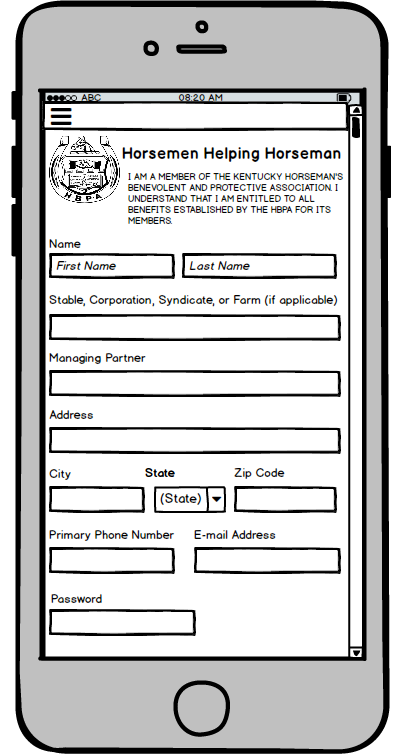
This Gantt Chart has been organized in chronological order of task assignments for each class deadline and the times we began the tasks and had them done by. Everyone in the group Serious Business was given their fair share of use cases and other deliverables to make this portion of the project successful, and organizing meeting times and spare time to work on these certain tasks was not a problem during this phase. We hope that we can keep our desired schedule the way it is leading up to the presentation date, and hope that our new ideas are worth implementing to the Kentucky HBPA.

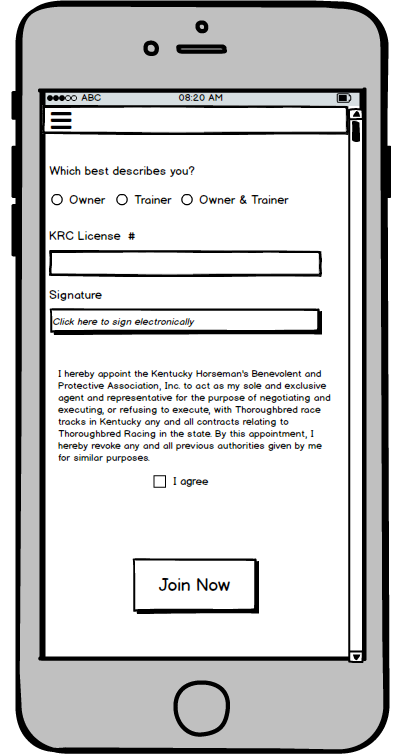
## Chart



# Prototypes

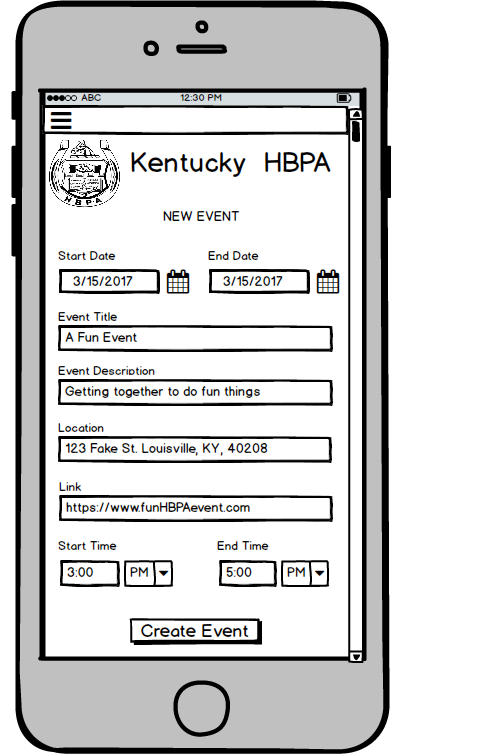
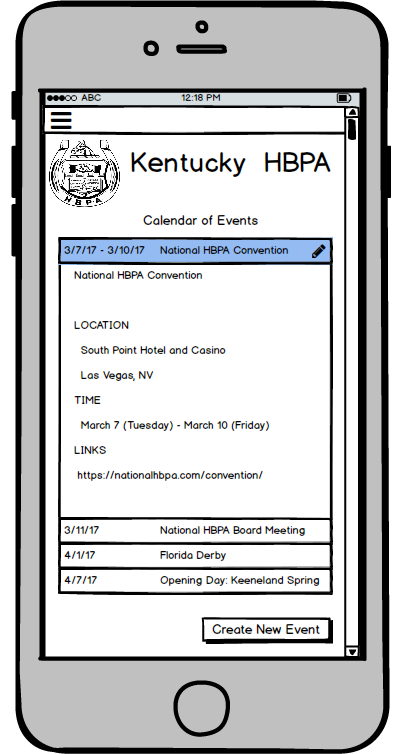
## Prototype: Create Account





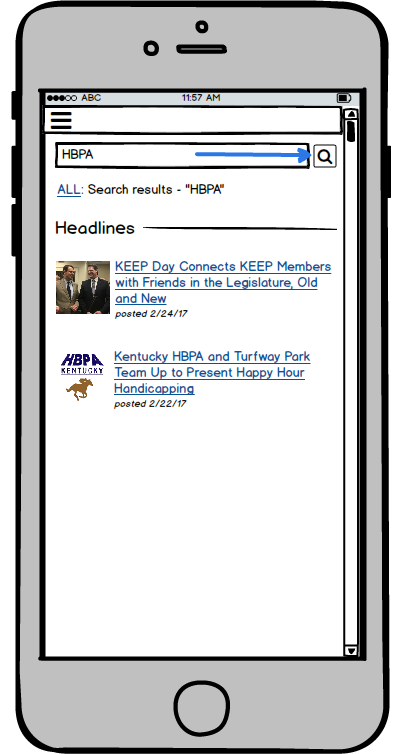


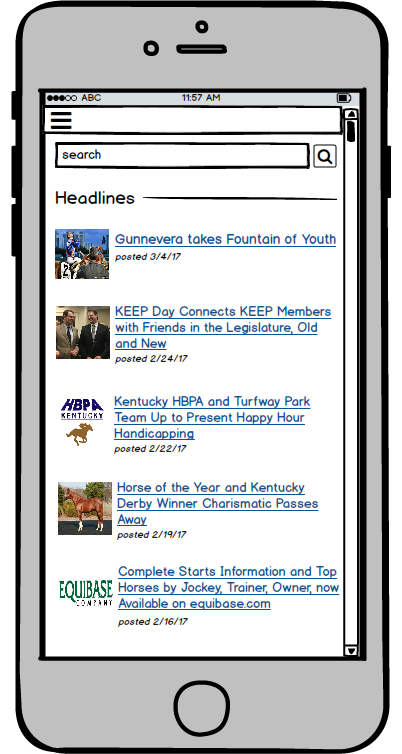
## Prototype: Create an Event





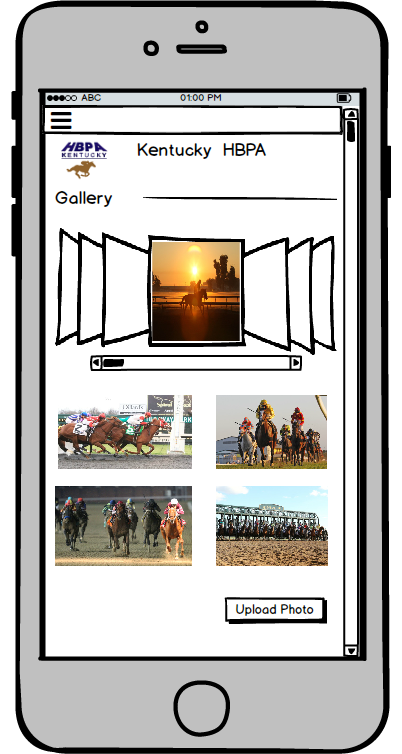
## Prototype: Search News Archives

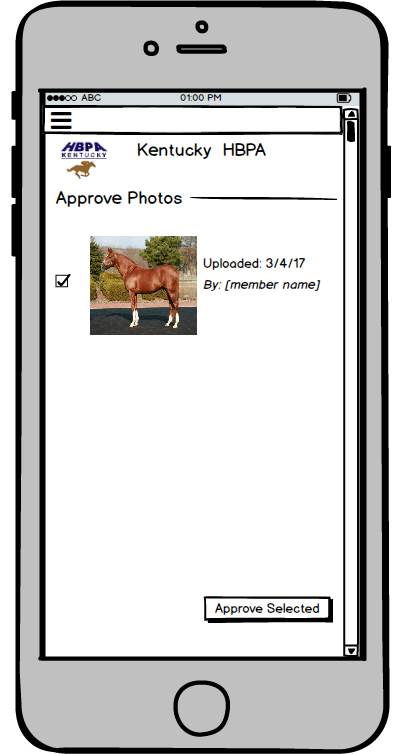
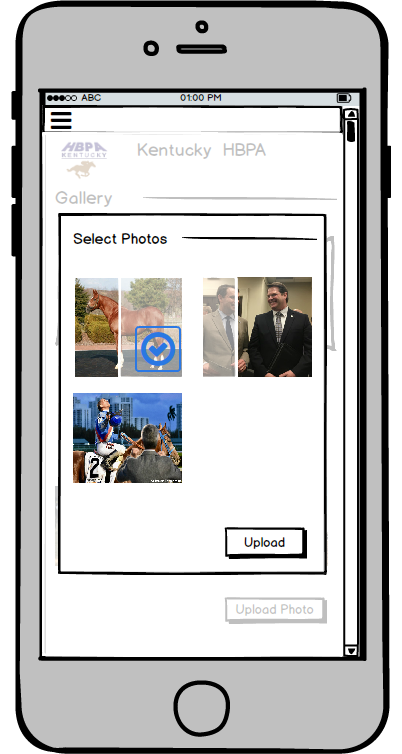






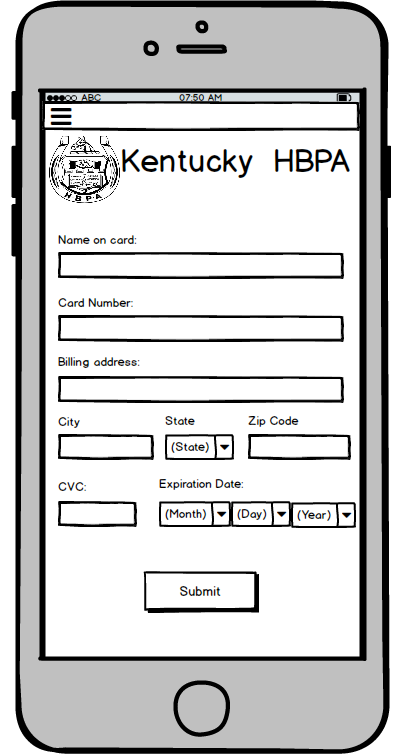
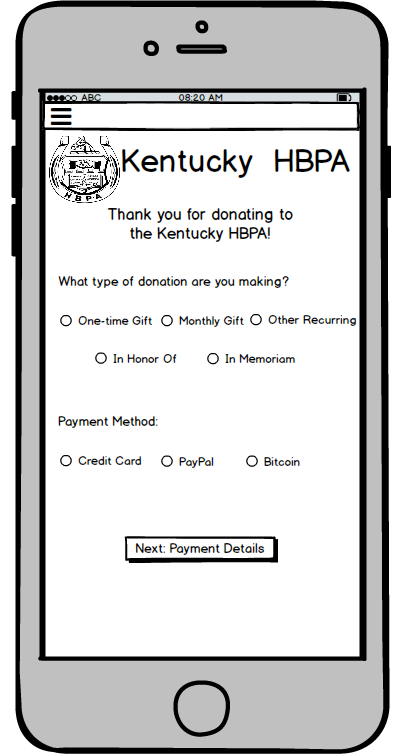
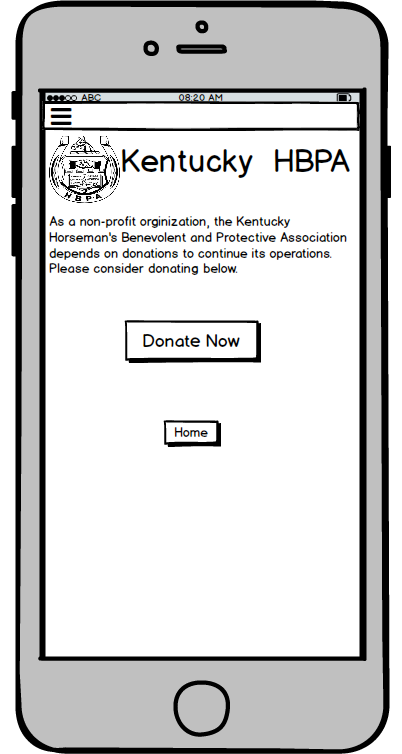
## Prototype: Upload a Photo



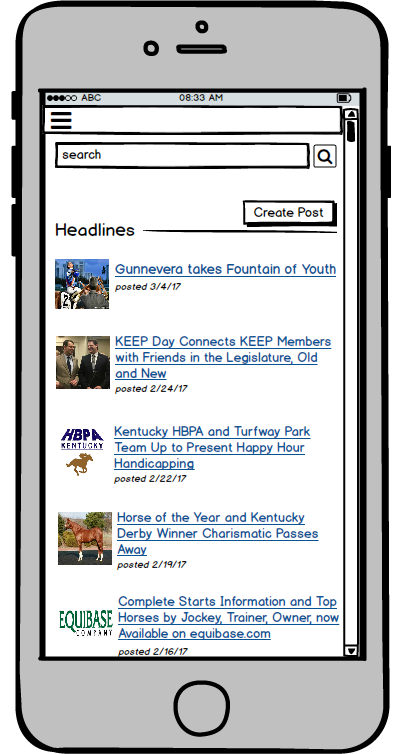




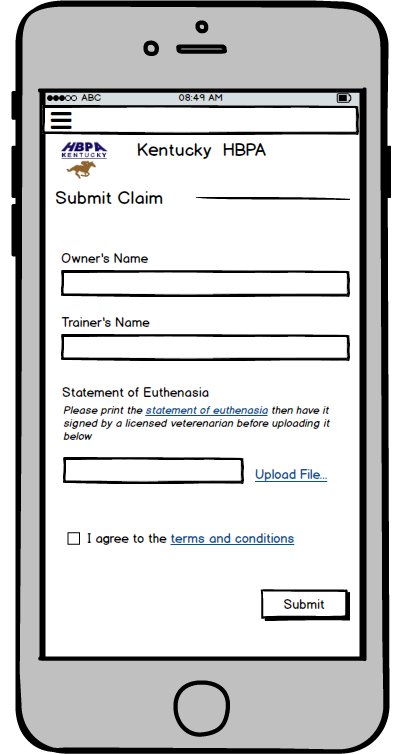
## Prototype: Makes a Donation



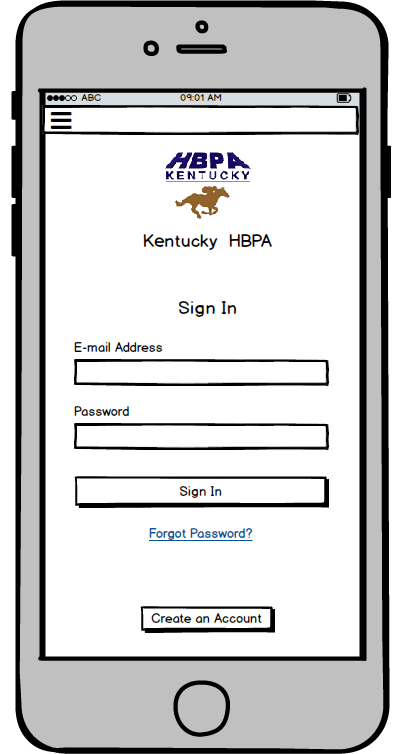
## Prototype: Posts a News Item



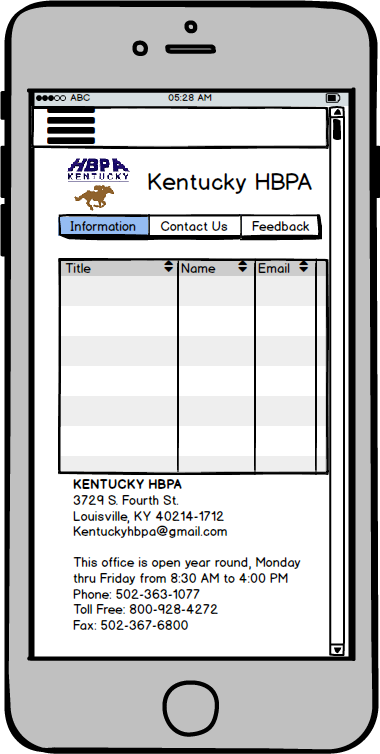
## Prototype: Submits a Claim

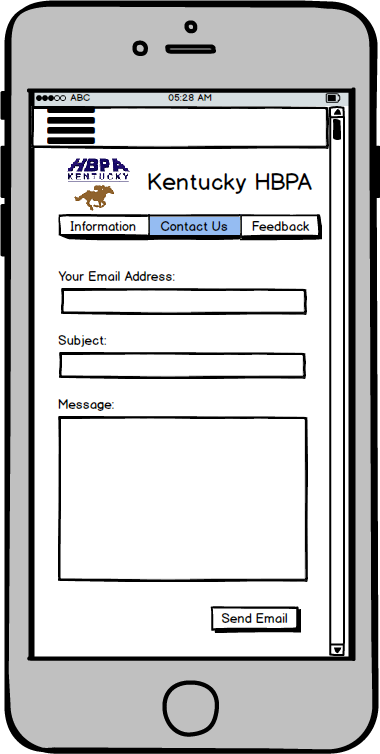


## Prototype: Signs In

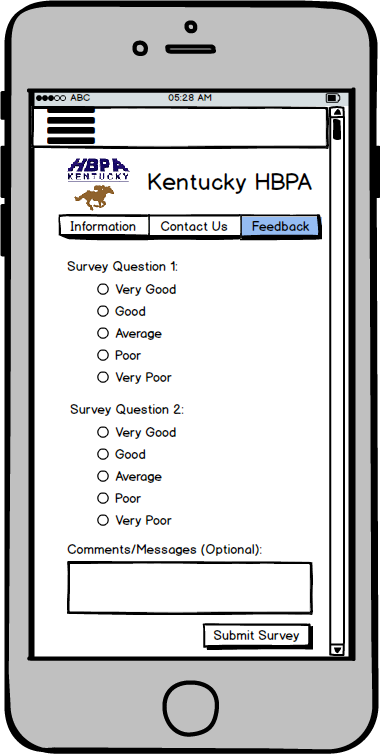
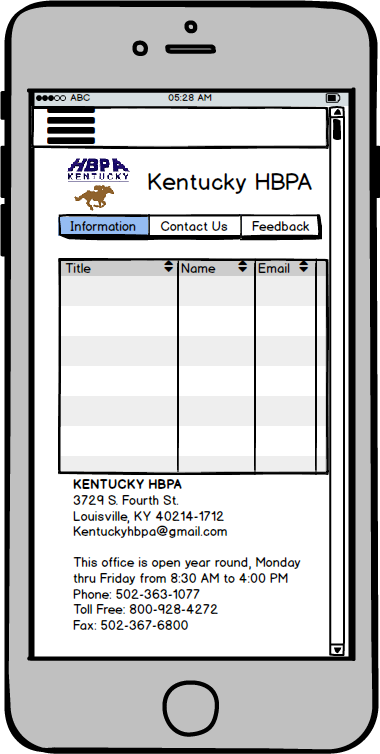


## Prototype: Contact KY HBPA

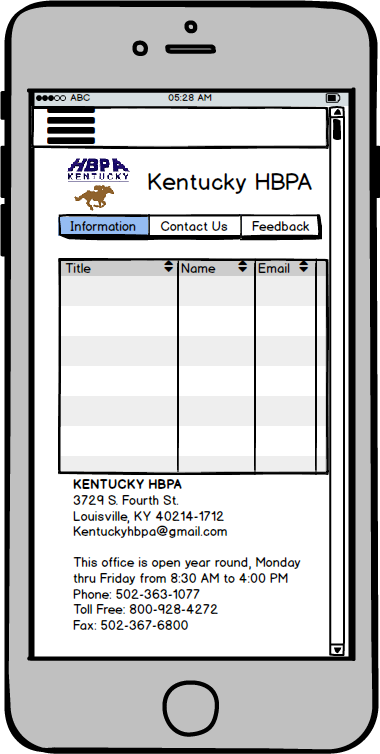




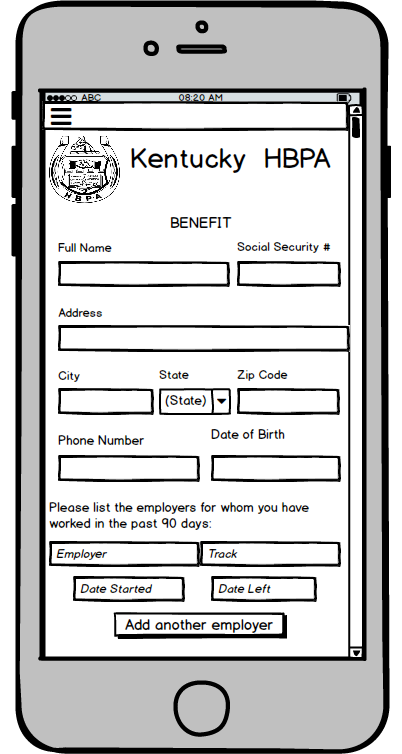
## Prototype: Provide Feedback

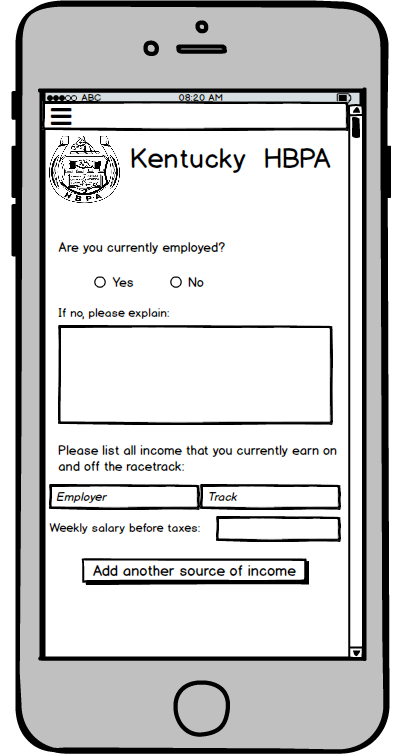
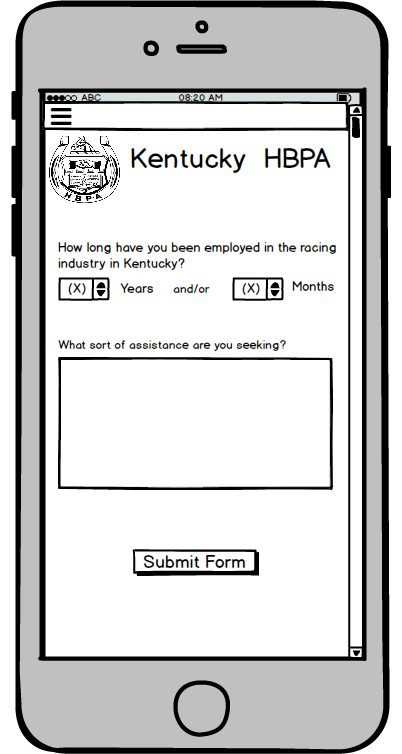


## Prototype: Request Contact Info

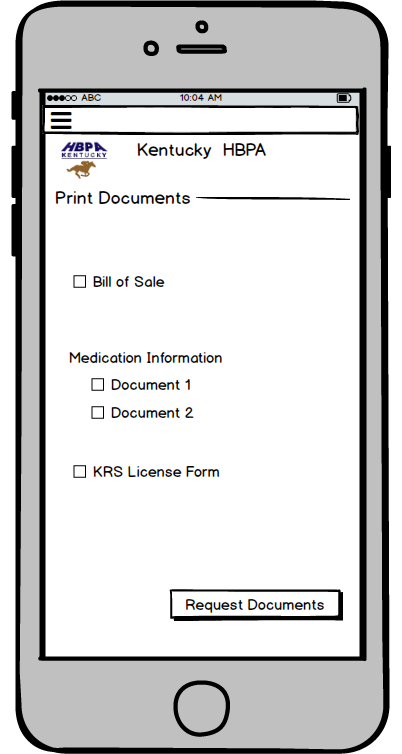


## Prototype: Submit Benefits Form

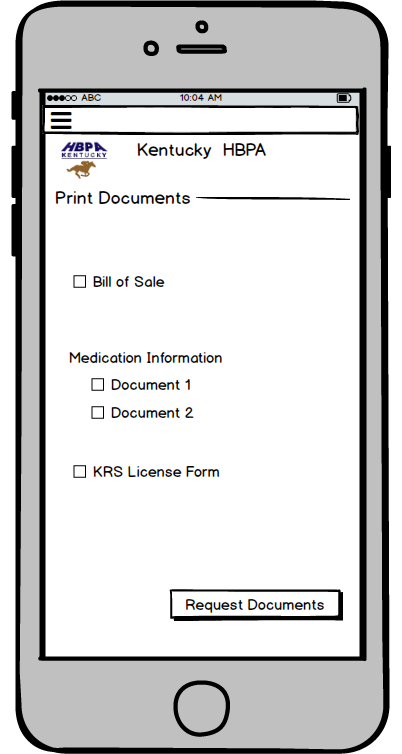




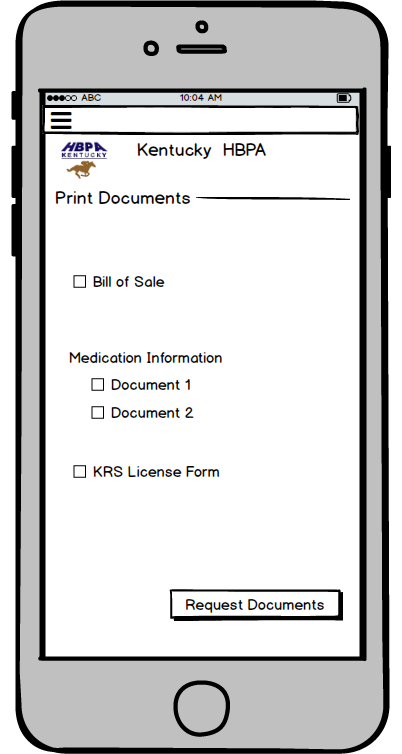
## Prototype: Request Medication Information



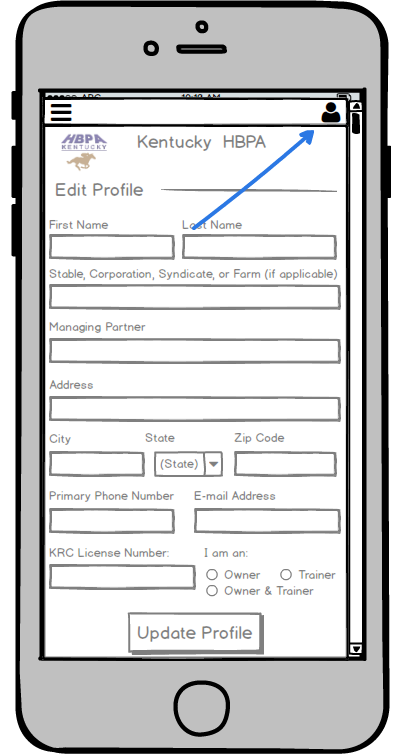
## Prototype: Request License Form



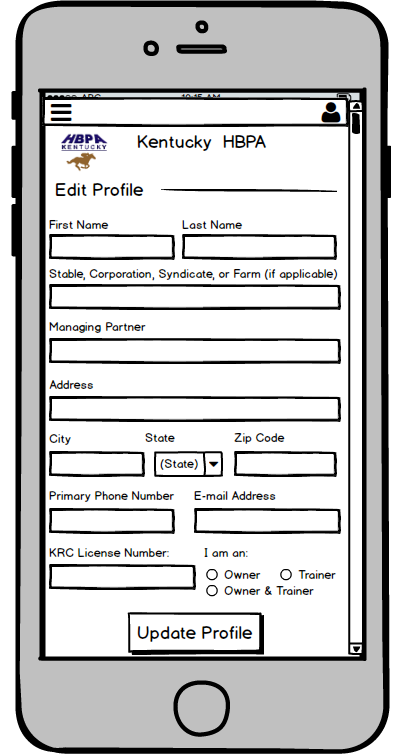
## Prototype: Request Bill of Sale



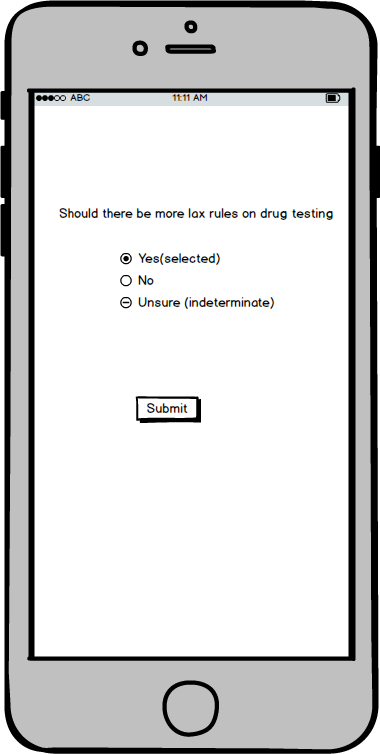
## Prototype: Request Profile Info



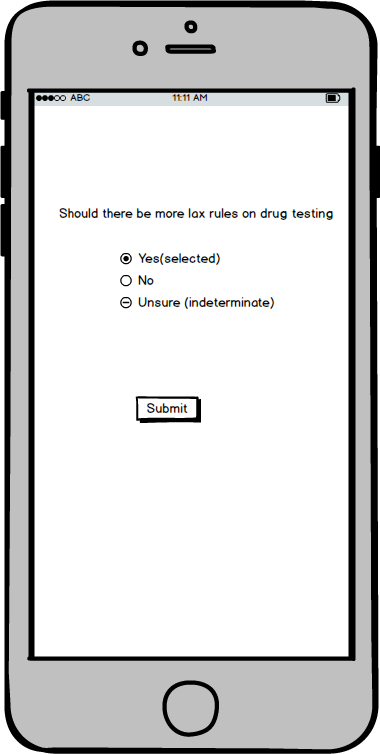
## Prototype: Edit Profile Info



## Prototype: Creates a Poll



## Prototype: Votes on a Poll



## Prototype: Posts a Link

