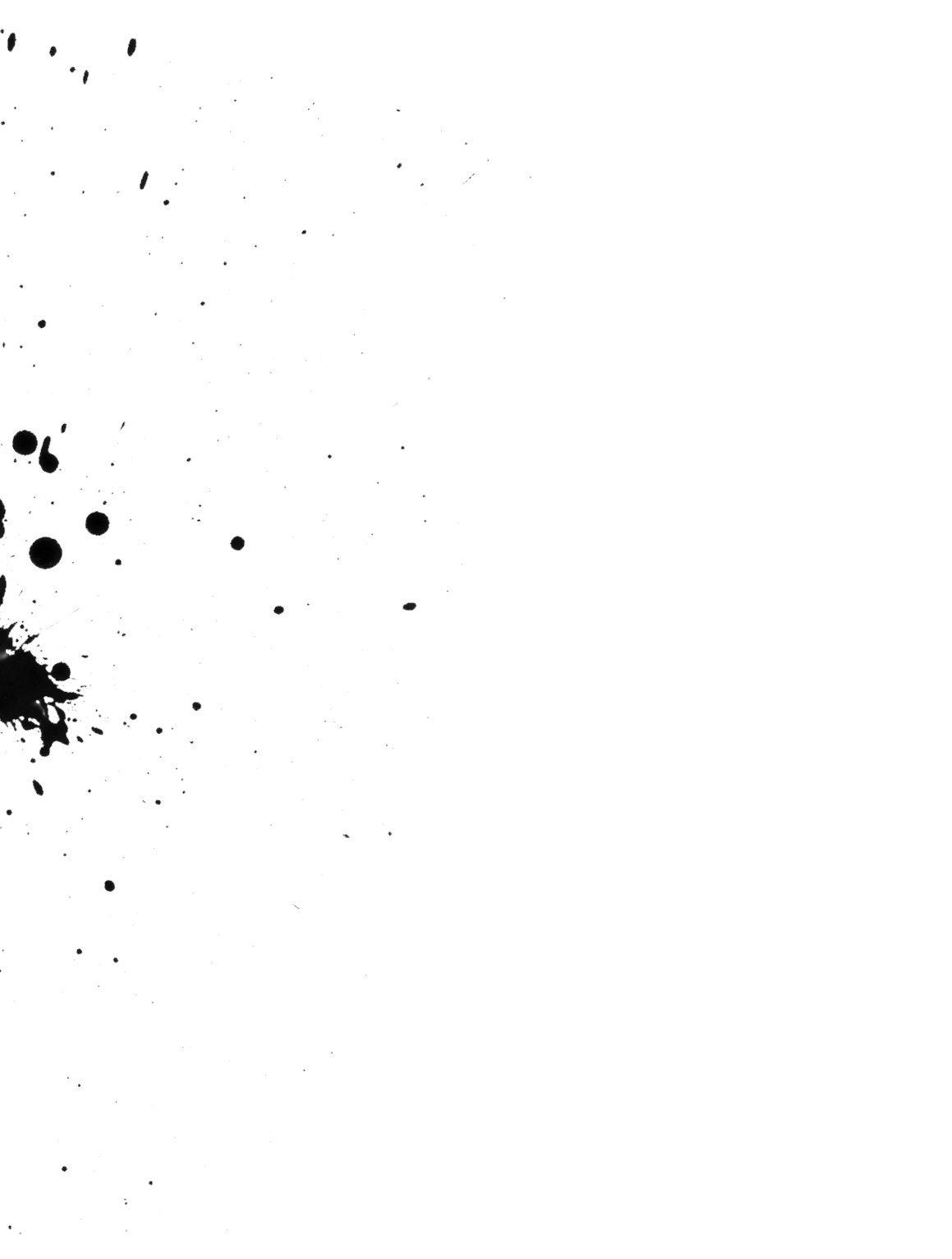
**Final Project Report**

**April 20, 2016**

**Groups for Project 14: Ilana Wasserman, Walker Mayercheck**

2016

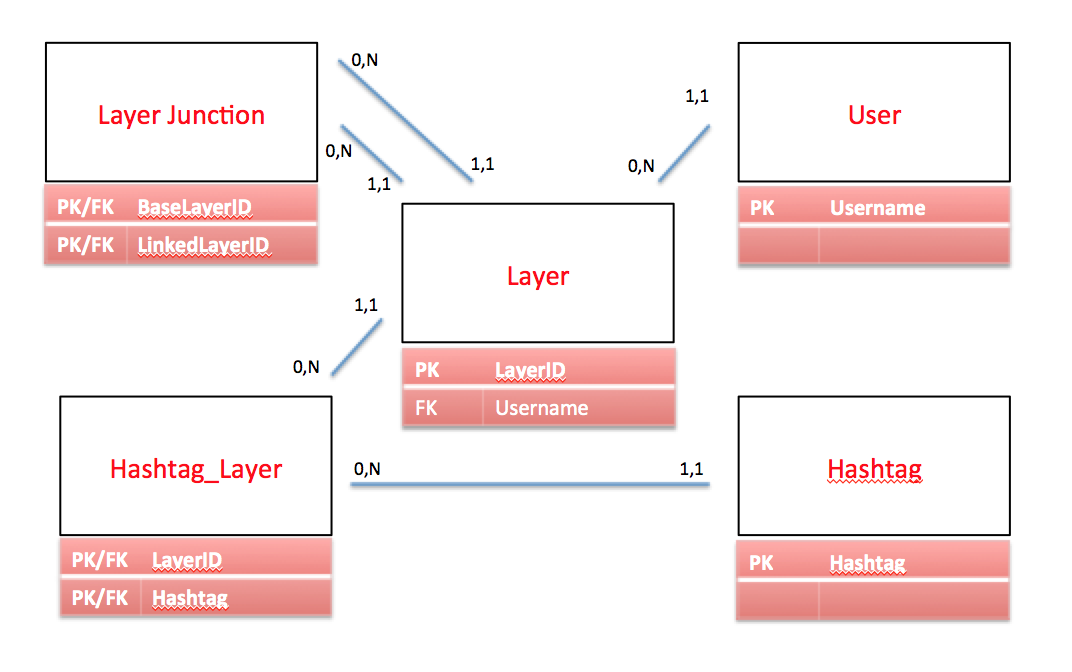
**Table of Contents**

|  |  |
| --- | --- |
| Table of Contents | 1 |
| README | 2 |
| ERD | 3-5 |
| * Simple ERD | 3 |
| * Detailed ERD | 4 |
| * EER Diagram | 5 |
| User Flow | 6-8 |
| Lessons Learned | 9 |
| Future Work | 10 |

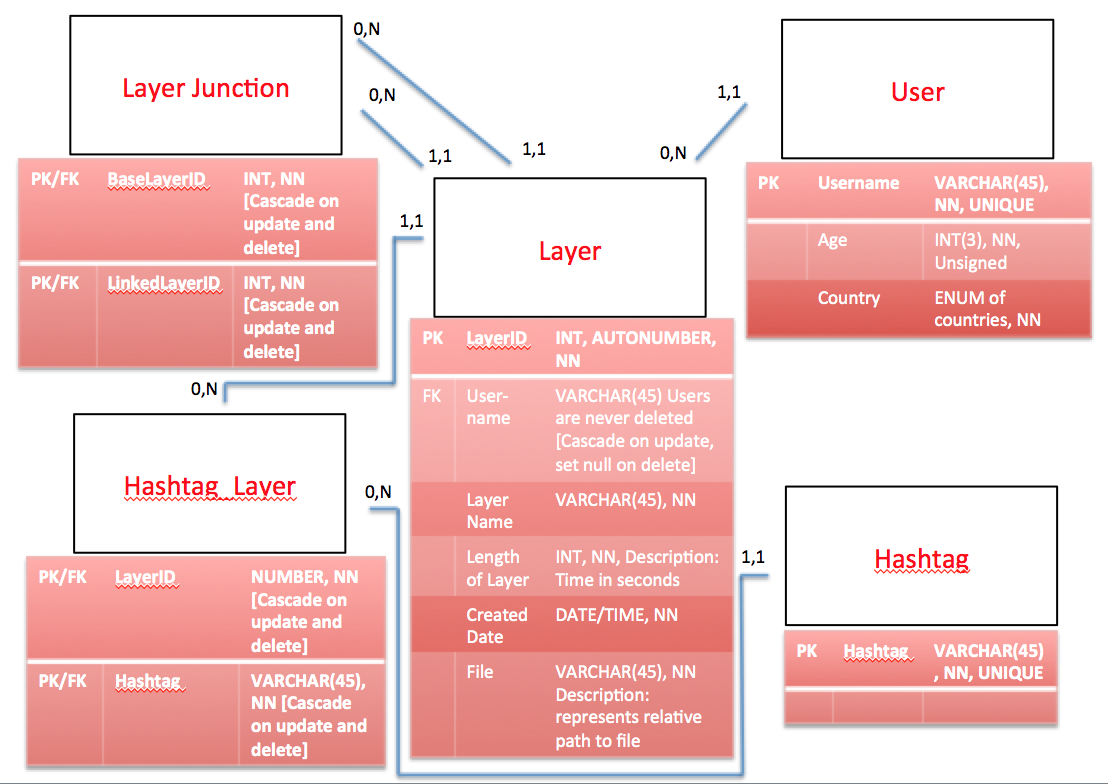
**README**

**ERD Diagrams**

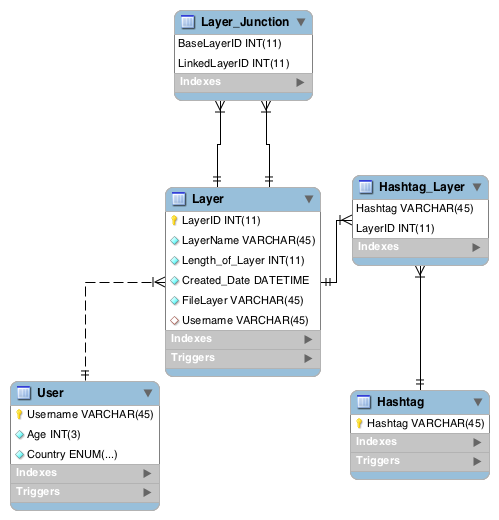
**Simple ERD**

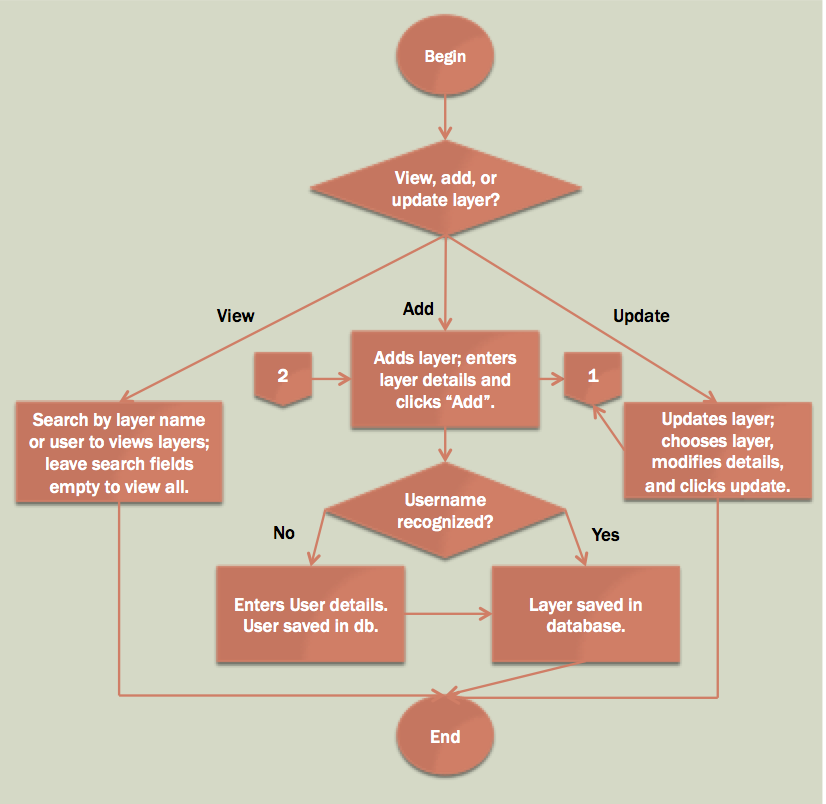
****

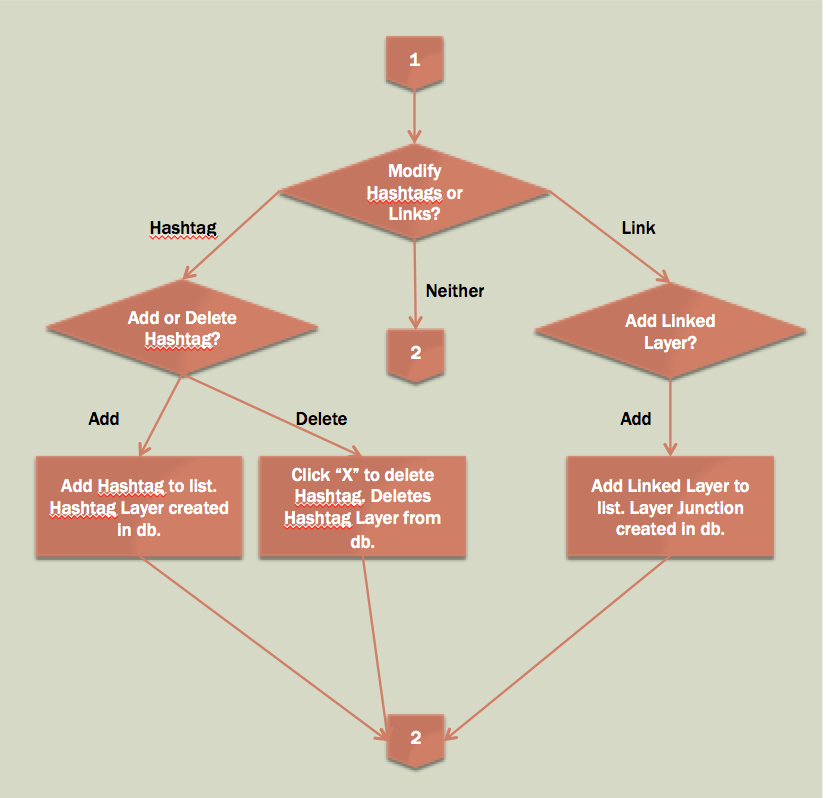
**Detailed ERD**



**EER Diagram**

**User Flow**





**Upload a Layer (Create)**

* Creates new Layer record
  + Autonumber is the primary key
  + User inputs the layer name, username, length of layer in seconds, and puts in a relative path to the file as a string
  + User creates hashtags
    - Need to create hashtag in Hashtag table AFTER search through table to make sure does not exist
      * If hashtag already exists, then DO NOT create new record in Hashtag table, move on to next step below
    - Need to create hashtag/layer record in Hashtag/Layer table, relating the hashtag to the layer
  + Today’s date is automatically stored as Created Date
  + User can attach the layer that they are uploading to multiple layers already in the database by entering the existing layer ids
    - BaseLayerID = currently uploading layer
    - LinkedLayerID = existing layer that current layer is being attached to
    - Need both and need new records in Layer\_Junction table for each layer that link current layer to
    - User cannot modify this relationship after it is created due to the fact that is it now part of the family tree for that layer

**Edit a Layer (Update)**

* Change Layer name (title)
* Delete/add hashtag
  + Add
    - Insert new record in the Hashtag table
    - Insert new record in the Hashtag/Layer junction table
  + Delete
    - Delete record in the Hashtag/Layer junction table
    - DO NOT delete records in the Hashtag table

**Delete a Layer (Delete)**

* Cannot delete a layer because other layers refer to this layer
  + Need to be able to see all of the layers in the family of layers to understand “family tree” and history behind the layer
* Since layers cannot be deleted, User’s are able to instead delete hashtags on a layer
  + Deleting the “Hashtag/Layer” record in the database

**View a Layer (Read)**

* Viewing all of the layers you add or search for
  + After upload a layer, can view the information that you inputted

**Create a User**

* When the username inputted to create a layer is not recognized, User enters information into the provided boxes and clicks Enter
  + Provided information:
    - Username, Age, Country
  + A new user record is created in the User table in the database with all of the provided information

**Views**

* User can see searched layers
* User can see links to the linked layers for a particular layer

**Lessons Learned**

**Future Work**

**User Table**

The focus of our project is how users create and interact with Layers and the functionality around the Layers (i.e. Hashtags). As such, the User table is not as robust as it would need to be in practice if we were designing a database to be used on a website with full functionality.

The User table would in practice need to have more complexity, such as an Active/Inactive field, to identify which users are currently interacting with the website and the database. In addition, the User table would need to have a Created Date field, to track when each User is created and therefore gain an understanding of how marketing efforts for the website are panning out and find trends based on seasons or specific events that trigger more users to join.

The User table would also have more functionality, and Users would be able to update their provided information, and delete a User account [create an inactive user rather than actually deleting the user from the database].

**Search Features**

Users would be able to search for specific layers and specific hashtags from the website.

**Statistics**

Users would be able to see which layers are the most popular (forked the most), and see the hierarchy of the layers as they are forked and then reuploaded to the database.