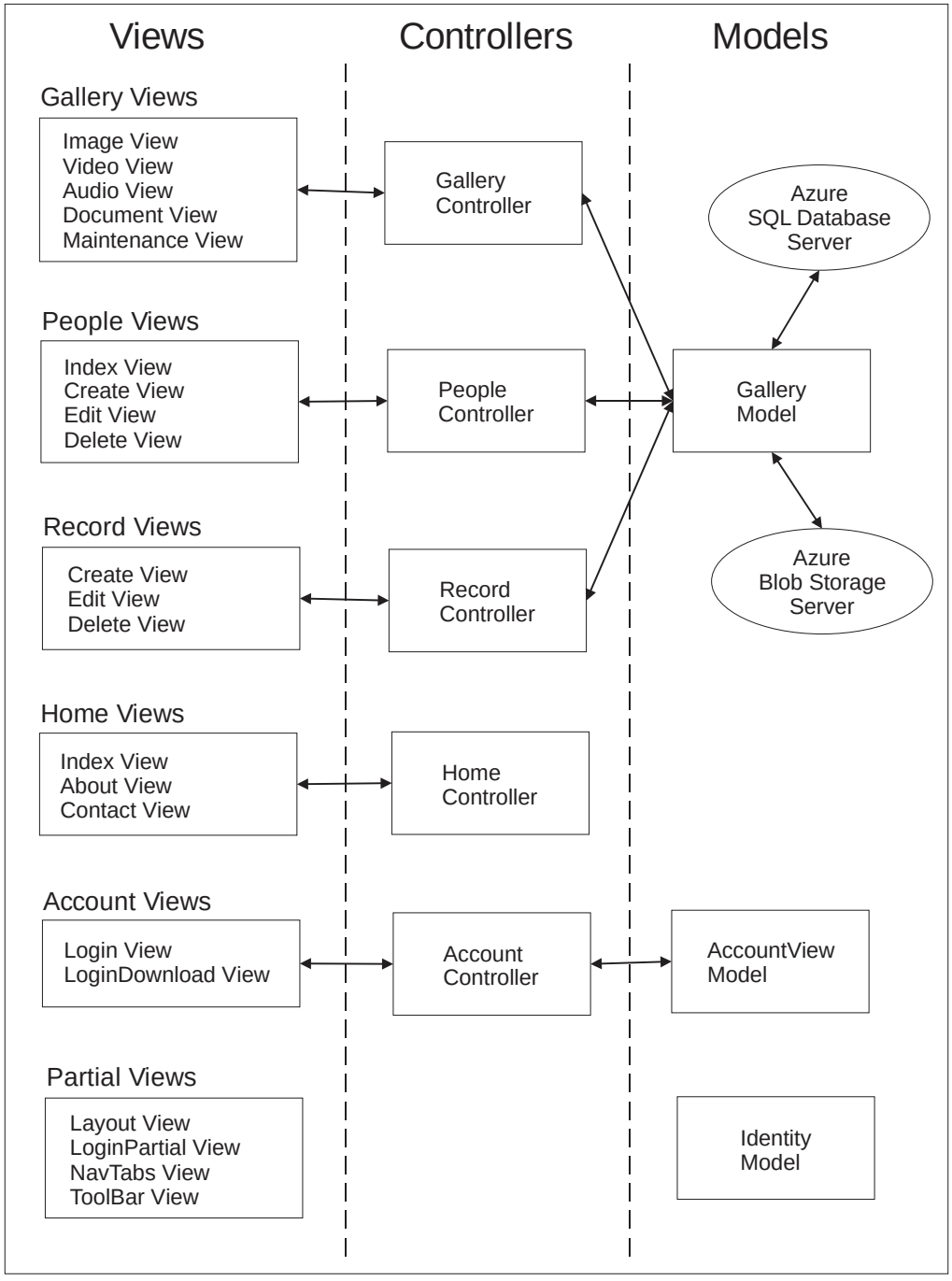
Stearns Family Archive Web Application Overview

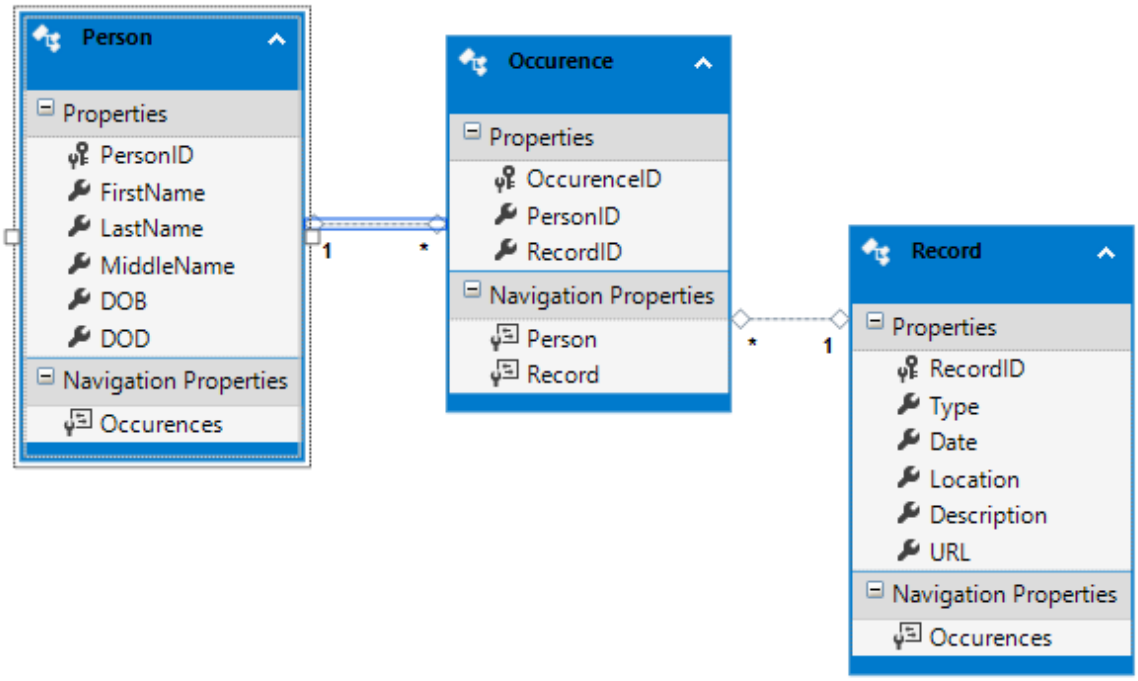
August 2019
D. Stearns

The web-based version of the Stearns Family Archive is a Visual Studio project named "StearnsFamilyWeb" and is developed using the Microsoft ASP.NET framework. This framework implements the Model-Controller-View pattern. The layout of the application is shown below.



The Views are web pages that are displayed in the browser of the client. These are written in html, CSS, and javascript. They can also contain server-side scripts written in C# that are prefaced with an "@" symbol. The Controllers and Models are running on the web server. These are written in C#. The role of the Controllers is to process requests from the client to change views. The Controller also sends and receives data from the Model that is required in the view. Their purpose of the Model is to define the structure of the data sent to a View and to interface with the data server and the storage server.

The archive database is stored on an Azure SQL server. The database consists of three tables: the Record, the Person, and the Occurrence. The organization of the database is shown below:

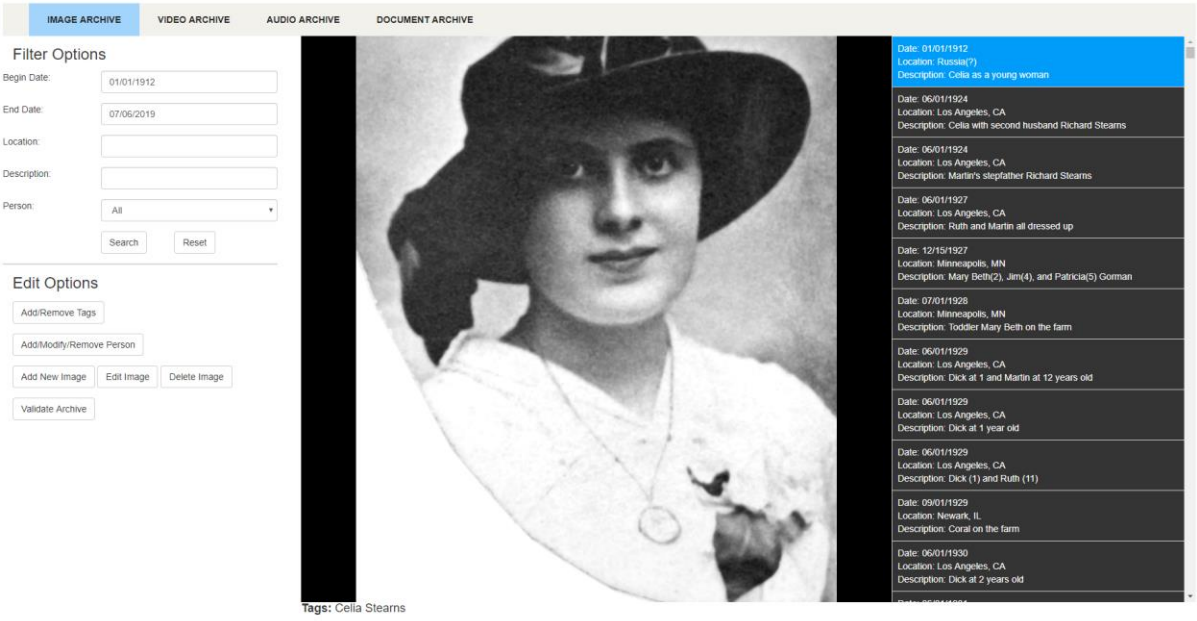


The Record is associated with content in the blob storage server. This is either an image, a video, an audio, or a document. It has the metadata for blob as well as the URL required to retrieve the blob. The Person is a member of a list of people that can be tagged to the content. The Occurrence is a tag that connects a Person to a Record.

The archive content is stored on the Azure blob storage server. It is divided into five containers: an image container, a video container, an audio container, a document container, and a thumbnail container. The thumbnail container has a low-resolution image for each blob in the archive. Each Record blob is assigned a unique RecordID and has a URL format of [blobname]_[RecordID].{ext]. The supported extensions are: jpg for images, mp4 and mov for video, mp3 for audio, and pdf for documents. The corresponding thumbnail has a URL format of [blobname]_[RecordID].jpg.

The Family Archive web application is accessed by navigating to the domain name “stearnsfamilyarchive.net”. The home page has basic information about the site and links to the archive applications. The “Login to Archive” link accesses the web application. The “Download Windows App” link installs a dedicated application called FamilyArchive that runs on the client workstation. The two applications have exactly the same functionality except for one difference: the web application has no capability to backup the archive on the client machine. Access to both applications requires the user to login with an email address and password. At this time there is no tracking of the email; any email with the standard structure will be accepted. However, access is password-protected.

The web application starts by opening the Image View. This takes up to 30 sec to load since there are currently more than 6000 images in the archive. The View is cached so that subsequent visits to the page load much faster, provided that no changes have been made to the database (e.g. new image uploaded). The layout of the View is shown below.



There are five basic elements in the view:

- The navigation bar on the top of the page.
- The directory of images on the right side of the page.
- The display area in the center of the page.
- The list of tags beneath the display area.
- The tools section on the left of the page.

The navigation bar is used to access the different types of content in the archive. These are the Image, Video, Audio, and Document Views. All of these have the same layout as shown above. The directory on the right side of the page is a one-column table showing the metadata for each image in the archive. This consists of the date, location, and description of the image. When the mouse hovers over a cell in the table the thumbnail image is displayed in the display area and the tags for the image are shown beneath the display area. Clicking on the cell causes the image to be loaded in the display area, and the cell color changes to blue to indicate that it is currently the displayed image. A mouse click on the loaded image converts the display to full screen. In this mode the images can be paged by clicking on the arrows located at the left and right edges of the page. Clicking on the image in full screen mode returns the display to the Image View.

The tools section on the left side of the page provides additional functionality. The Filter Options section is used to down-select the images of interest based on a date range, a keyword in the location text, a keyword in the description text, and a person in the tag list. The Edit Options section is used to make changes to and validate the archive. Specifically, the Add/Remove Tags button opens dropdown lists showing the current tags and the people that are available to be tagged. Clicking on the people list adds a tag; clicking on the tags list removes a tag. The Add/Modify/Remove Person button navigates to the Index View of the People Views group (see diagram above). This displays a list of all the people in the Persons database. A new person can be added to the list. An existing person can be modified or deleted from the list. If a person is deleted then all of their tags (in the Occurrence database) are also deleted.

The Add New Image button opens the Create View of the Record Views group. The input form is initially filled with the metadata of the currently selected image. The user selects an image located on the client machine to upload to the archive. A thumbnail image is created and displayed. Also, the EXIF header is examined to find the date and location of the image. If this metadata exists then the date is displayed in the Date textbox and a map of the location is displayed below the form. Clicking on the Submit button results in the image file being uploaded. This consists of a new record being created in the database, the file being uploaded as a new blob in the Image container of the storage server, and the thumbnail image being uploaded as a new blob in the Thumbnail container. The file is uploaded in 2 MB chunks and the status of the upload is display in a progress bar. When the upload is completed the display returns to the Image View, and the new image is automatically selected to be the current image in the directory.

The Edit Image button opens the Edit View of the Record Views group. The input form is initially filled with the metadata of the currently selected image. The date, location, and description fields can be changed. Clicking on the Submit button causes the Record in the database to be modified and the display returns to the Image View.

The Delete Image button opens the Delete View of the Record Views group. The delete operation is password-protected with a unique administrative password. Clicking the Confirm Delete button results in the blob being removed from the Image container of the storage server, the thumbnail being removed from the Thumbnail container, the Record being deleted from the database, and all of the tags (Occurrences) being deleted. The display returns to the Image view and the next image in the directory becomes the current selected image.

Finally, the Validate Archive button opens the Maintenance View of the Gallery Views group. This view is used to perform housecleaning on the archive. Occasionally the required one-to-one correspondence between Records in the database and blobs in the storage server is broken. This results in either orphan records that have no blob or orphan blobs that have no records. Clicking the Start Scan button performs an operation that scans through all records and blobs to find orphans. The orphans are listed and, if any exists, an option to remove the orphans is displayed.

The Video, Audio, and Document Views are functionally similar to the Image View. The navigation bar and tools section are identical. These are, in fact, the NavTabs View and the ToolBar view of the Partial Views group. They are embedded as partial views in the Gallery Views (Image, Video, Audio, Document). The main difference in the Gallery Views is the way the content is displayed. All of the views have a element (id = myMediaContainer) that contains a display element (id = myRecord). In the case of the Image View the display element is an <image> element. In the case of the Video view it is a <video> element. In the case of the Audio View it is an <audio> element. In the case of the Document View it is an <iframe> element. Using these HTML5 elements provides special functionality. For example, the <video> element creates a video player that has the standard toolset for playing a video, including changing to full screen mode. The thumbnail image for the Record is loaded in the background of the element. To display the thumbnail (mouse hovering over the directory) the visibility of the display element is set to “hidden”. In this way a simple CSS call in javascript can be used to toggle between the thumbnail and content displays.

We conclude with a brief description of the code behind. In the MVC pattern the data is sent to a view using a model. For example, the Gallery Views are sent a model called ArchiveData of this structure:

```
public class ArchiveData
{
    public List<Gallery> RecordData { get; set; }
    public List<Person> PersonData { get; set; }
    public string TabName { get; set; }
    public string CurrentRecordID { get; set; }
}
```

The RecordData consists of the metadata for the records in the view, including the URLs for the content and thumbnail blobs. The PersonData consists of the metadata for the Person database. The TabName is the current view (Image, Video, Audio, Document) and the CurrentRecordID is the Record in the directory to be loaded when the view is initially displayed. All of the data in the model is accessible to the client by converting the model properties to javascript arrays (JSON) and variables. During the initialization of the view we store the record metadata in the "data" attribute of each directory element. This includes the RecordID, the Date, the Location, the Description, the blob URL, the thumbnail URL, and the list of tags. The PersonData in the model is used to populate the dropdown list in the person filter of the Filter Options.

Most of the functionality on the client side is performed using CSS and javascript. In particular, loading records and thumbnails for display, filtering the directory, and changing to full screen mode are all done on the client and hence are fast operations. The calls to the controller(s) results in operations performed on the server side. These include changing the view, uploading/modifying/deleting records and people, and adding and removing tags.