Foodograph

TEAM FOODAHOLICS

Rashmee Prakash – *rprakash@gmu.edu*

Shumaila Ahmad – *sahmad10@gmu.edu*

Elizabeth Guarasci – *eguarasc@ gmu.edu*

Michael Rosenberg - *mrosenb3@gmu.edu*

I. Abstract

Foodograph (combination of ‘food’ and ‘photograph’) is an application that is based on the ever-growing popularity of food photography. This app allows a user to create an account and log into the application. From there, the user has the choice to upload their own picture of food from either their device camera or from their device photo gallery, view and rate photos that have been submitted to the application, and see their stats. When viewing the images, the user has the option to either ‘like’ or ‘pass’ the photograph. The number of ‘likes’ are compiled on the stats page. Users also have the ability to share photos to social networks.

II. Completion Report

While the inception of the idea for Foodograph has been a relatively simple process, putting that idea into code was a different task altogether. The scope of this project was a lot more than Team Foodaholics had even remotely expected. Larger milestones were broken down because tasks that the team had assumed were going to be simple ended up being very complex. The first milestone that was completed was the creation of a successful user registration using NSUserDefaults. A flexible registration page was created in that, both registration and login could be handled via one view. If the user was a returning user who had closed the app, instead of seeing a registration page, they would be faced with a simple login form to enter the application. This way, the user is not prompted to register when they already have the downloaded app and have registered themselves.

Next, Team Foodaholics focused on view that would allow navigation into the other views and allow for extended functionality. While the original plan expected no more than four views, Team Foodaholics ran into an issue of only allowing a limited set of functions on one view. After struggling with attempting to put the majority of the functionality on one view, the team decided to make this view the navigation point to other views and assign each different view a specific functionality. The navigation view became the central point of the application, in that it allowed navigation to upload/take photos, view/rate/share photos, and view user stats. It also allowed a logout capability for the user to exit the application.

The next view that Team Foodaholics focused on was allowing the user to upload a picture of food to the application. While it only took the team a few hours to figure out how to upload photos from the gallery and take a photo using the camera feature, it was much more difficult to implement the saving of the photo to the application. This saving feature took several meetings and individual work to be fully completed and functional. Both a local and a global NSMutableArray was created for the saving of the user's uploaded photo. This array's functionality was imperative as it was how the user would view photos in the application as well. Functionality of the uploading of photos view became the second completed milestone.

The fourth view became much easier after the implementation of saving photos to the array. Team Foodaholics updated the image generator with sample photos as well as populating it with the user's uploaded images. This array was then outputted to the screen, one photo at a time. The user would be able to either 'pass' or 'like' the photo. Sharing functionality was also implemented and tested on Team Foodaholic's twitters. While the third view took some time as well, it was much more easier to complete as the bulk of the work had been done in the previous view. The fifth view was connected to the like button that the user is able to press, and calculated the number of photos the user has liked. Team Foodaholics considered adding the number of passed photos as well, but then felt that it would be unnecessary.

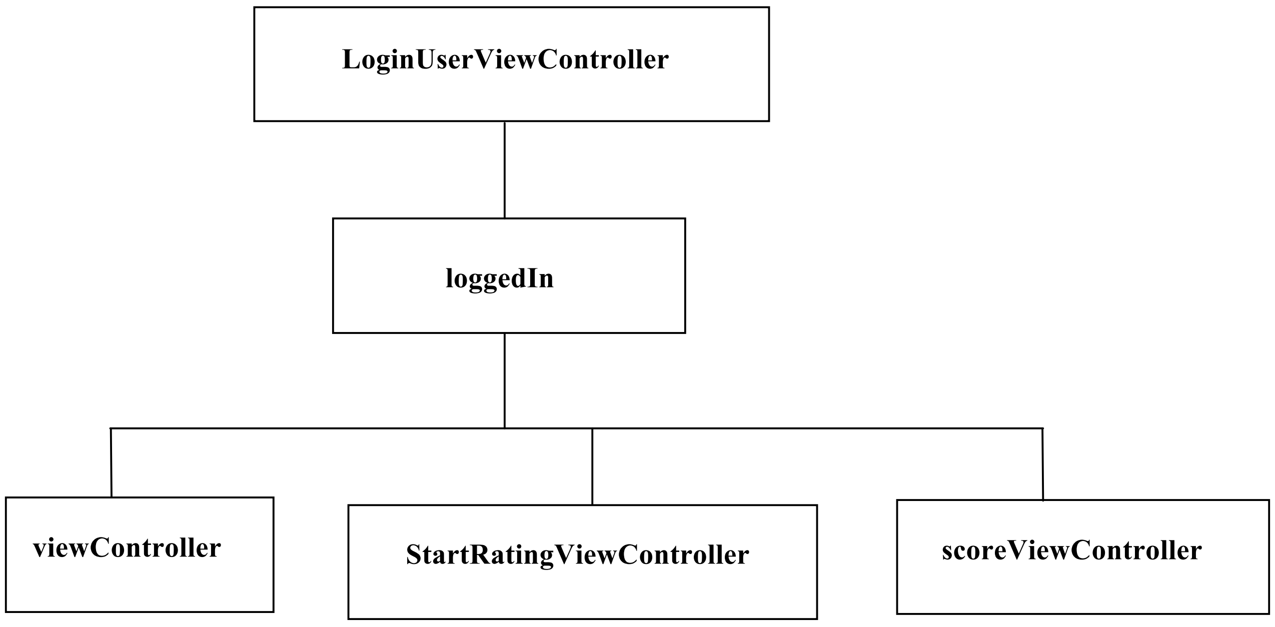
The last part of this application was final design and beta testing of all the features. While Teaam Foodaholics had been working on design and building simultaneously, they made specific design changes from their original plain, white design. A color scheme was referenced throughout Foodograph, and the app was made to look fun and inviting. Beta testing was done on each individual feature whether it be saving to the array (outputting the number of elements to see if the images were actually being saved to the array), or number of liked photos by the user (liking photos and seeing how many likes were presented on the stats view.) The team unanimously agreed to not create scoring based on how many of the user’s photos were liked. The team felt it was not a key feature and did not have enough time to implement it, even if they wanted to. Team Foodaholics was pleasantly surprised that they enabled the sharing to social networks functionality as that was not something they were expecting to be finished with. One of the primary issues the team ran into was scope creep, in that while they were working, they kept thinking of other ideas to make the application more functional and exciting. However, many of those ideas were not part of the original plan and took the focus away from the primary functionality that needed to be completed. Another problem that the team faced was an unequal contribution from the team members. One team member only attended one team meeting, and did not contribute anything to the building, testing, design, documentation, or even the creation of the proposal, beta testing, and the final presentation. This put a lot of stress on the rest of the team as they were going out of their way to meet several times to complete the application on time. All the milestones that the team had set for themselves during the last update have been completed, and the team was unsurprised that their plan changed over the course of the last few weeks. Attempting to build the application gave the team insight on what they were able to do and what capabilities and knowledge they had. The team realized they were not going to be able to implement a GPS to find pictures near their area, as they ran out of time and had to focus on the non-technical side of their application such as documentation and presentation.

III. Design

a. MVC Hierarchy

Foodograph has several classes which control the navigation of the application. Once the user registers an account or logs into their account, the view will transition modally to the primary navigation page that will describe the user's account and allow them to choose between three different pages: upload/take photos, view, rate, and share photos, and see their account stats. Sharing is enabled on the page that allows users to view and rate the photos. A back button is implemented on all views (except registration/login) to take the user back to the primary navigation page. (This navigation page will also allow the user to logout of their account.)

b. Class Decomposition



IV. Layout and Use Cases

Team Foodaholic's use cases have not changed much from their beta update. The user is able to interact with the system as stated before. The most successful scenario would be that the user has the ability to register and login. Their credentials are validated and move them to their navigation page. The second use case will be that a user can successfully upload their photos whether it is from their gallery or from the camera on their device. The next use case will be the other option the user is given and that is to continue on to view and rate photos. A successful scenario is that the user is able to view multiple photos and have the ability to 'pass' or 'like' them. The next successful scenario involves the user navigating to the view which allows them to see how many photos they have liked. The number should reflect how many times the user has liked a photo. The last scenario involves the user's ability to logout of the app, and navigate to different views in the app.

V. Implementation

a. ViewController.h/ViewController.m

The ViewController class allowed the user to either take a new photo from a camera or choose from existing photos in their gallery. Once the user had picked the photo, the user could save the photo into the local array which would copy into the global array. (A local array was needed because images were not being saved to the global array.) This view was connect through a push segue from the loggedin view.

b. LoginUserAppDelegate.h/LoginUserAppDelegate.m

The Global array was declared an initialized in the LoginUserAppDelegate class. This allowed it to be accessible from any other class.

c. startRatingViewController.h/startRatingViewController.m

The startRatingViewController class displayed the array of images both saved in the application and uploaded by the user. This class gave the user the option to either ‘pass’ or ‘like’ the photo. It also enabled sharing functionality to allow the user to share the photo on a social network such as Twitter. This view was connect through a push segue from the loggedin view.

d. Main.storyboard

The Main.storyboard was used to construct the visual display and connect the views with transitions and button.

e. scoreViewController.h/scoreViewController.m

The scoreViewController class allowed the user to see how many photos they had liked. This view was connect through a push segue from the loggedin view.

f. loggedIn.h/loggedIn.m

This class served as the navigation to all the other views. It presented the user with option to either upload/take a photo, view/rate/share photos, or view their user stats. It also allowed the user to log out of their account. This view was connect through a modal segue from the LoginUserViewController view.

g. LoginUserViewContoller.h/LoginUserViewController.m

The LoginUserViewController performed several authentication functions to produce a successful registration or login. Its methods included a check to see that the password and the confirm password field received the same input, and presented the user with an error message if they did not match. It also did now allow the user to login, if the user's username and password did not match. Account data was saved in NSUserDefaults. This class also hid the login button if the user was a new user who had just downloaded the application onto their device, and hid the registration field if the user was a returning user.

VI. Future Work

A great feature of Foodograph is that it is a very versatile app and can be used as-is, or can be expanded with several new features. If Team Foodaholics had the time, they would have liked to add identifying information to each photo that the user uploads. For example, they would have liked to let the user give the photo a name, a location, and a few tags that would identify it such as type of cuisine, or spice-level. They would have also liked to have expanded this idea and provide the user with a chart of frequently liked tags. For example, User1's likes could encompass 70% Thai, 20% Fast Food, and 10% restaurants. Another feature that Team Foodograph would have liked to implement is an optional feature that allows users to find food photos near them using the GPS feature. All in all, Foodograph is a versatile application that is both functional and expandable for the future.