Development Testing

Strategy and Approach

Following the test-driven development outline, we will draw from our user stories, client comments and project requirements to create the tests. This will allow us to order the development of the different features of the app, based upon which parts depend on each other. For each of the app features the group has discussed in detail with the client what the feature should achieve and how it should take information from the user. All information should be displayed in a user-friendly format. When dealing with data being input by the user, for example when they add information to a new picture, we will use equivalence partitioning to test the robustness of the app. This ensures that the information we store is valid and that the app will not crash if the user enters unexpected values, for example an abnormally large string for their name field. Other outlier cases will be accounted for, including incorrect data types and empty fields.

Core Component

A core component of our system is the feature that allows the user to take a picture of their skin lesion/mole. They will have diary entries for moles that they have already begun tracking and will have the ability to add new diary entries for new moles. In this test description we will focus on the scenario where the user is adding new images to an existing diary entry. This feature is a key requirement specified by the client. In order to help the user track their moles they must be able to take periodic photos so that a doctor can look for changes.

Test	**Explanation**
The camera page displays a button at the bottom. Used to take a picture.	When this button is pressed a picture is taken and held temporarily. After pressing the button, the user is taken to the review page to check that image and add any additional information.
The camera page has an option to show a ghost photo.	This ghost photo is used to help keep consistency between the images, better allowing the doctor to spot changes in the mole. The ghost photo will be of the first image the user took. They can click the button to display and remove the ghost image. It shows up like a filter would.
The new picture is added to the correct diary entry.	There are two possible options on the review page, ‘save’ and ‘save and send’. Both options will store the image in the entry the user specifies.
The new picture has the attributes the user added in the review page.	These attributes include, whether the image is a near shot or a far shot, any additional comments about the photo from the user and the date the image was taken. The date will not be a user input; we will instead use a date & time module.
The user is encouraged to take pictures routinely. When a new picture is taken, the time until they need to take another set of images, is updated accordingly.	Each diary entry will have the date of the last entry and use this to remind the user of when they next should take a new photo. This date should be updated when a new entry is added.
The user is asked to take both a near shot and a far shot when adding a new entry.	Once the user has taken one of the two shots, they are taken back to the camera page so they can take the other shot.
After taking a picture the user is taken to the diary page.	The user can now see the diary entry has been updated with the two new images.
Back button on each page.	Both the review page and camera page will have a back button in the top left corner. This will bring the user to the previous page they were looking at. If the user does not save an image at the review page and presses the back button instead, the image will not be saved.
Helper button.	Both the review page and the camera page will have a helper button in the top right corner. Pressing this will display more detail about how to use the features of the page.