Functional Solution Document

Image gallery selector for devices with no display monitor systems

Version

1.0

Prepared for

App Pilot

Prepared by

Einar Lárusson

Contributors
Ingvar Örn Ingólfsson

1 SOLUTION DESIGN

1.1 Requirement

In scenarios where there are no display monitors in parks or other places where PYF will be used there is a need to make the app be available to display all images for a given ride/device in an image gallery.

For version 1.0 we will need to add a filter/search functionality to provide users the option to get images for a given time span.

1.2 Solution Visualisation

For rides/devices that have the field Device.HasMonitors = false the app needs to display an image gallery instead of fetching an image based on a display id that the user gets from monitors in the park for the ride the image is from.

1.3 Modifications

In the fragment for buying images we need to do a check on the field Device.HasMonitor. If the value of the field is false then the gallery should be displayed. The user is taken to a new screen that displays the gallery.

The gallery should just be a standard image gallery for Android or iOS.

These fields have been added to the Device table

Field	Type	Comment
NumberOfColumns	Number	Control the number of columns
NumberOfMinutes	Number	How many minutes back in time should be loaded in a bulk load to populate the gallery for the device
ImageSold	Boolean	Used to control if images for the device are sold or free of charge e.g. included in the ticket price

When the user has selected an image based on the settings for the device in DynamoDB the user clicks the Done button in the top right corner. If the user needs to pay for the image the user is navigated to a PayPal screen to handle the payment. If the image is free of charge then the user is navigated to share to social media (FB, Goggle and Twitter) screen.

Fetching images for the gallery

Use the **DeviceId-DateTime-index** GSI on the the ImageListQuery table in DynamoDB. A filter needs to be done to filter on the DateTime field on the ImageListQuery. The DateTime field holds a milliseconds representation of the date and time the image is uploaded to the database. We use this number to filter on and do a range for images. This means that the Android and the iOS apps need to convert the date time (dateNow) to milliseconds and use that as basis for the DateTime range.

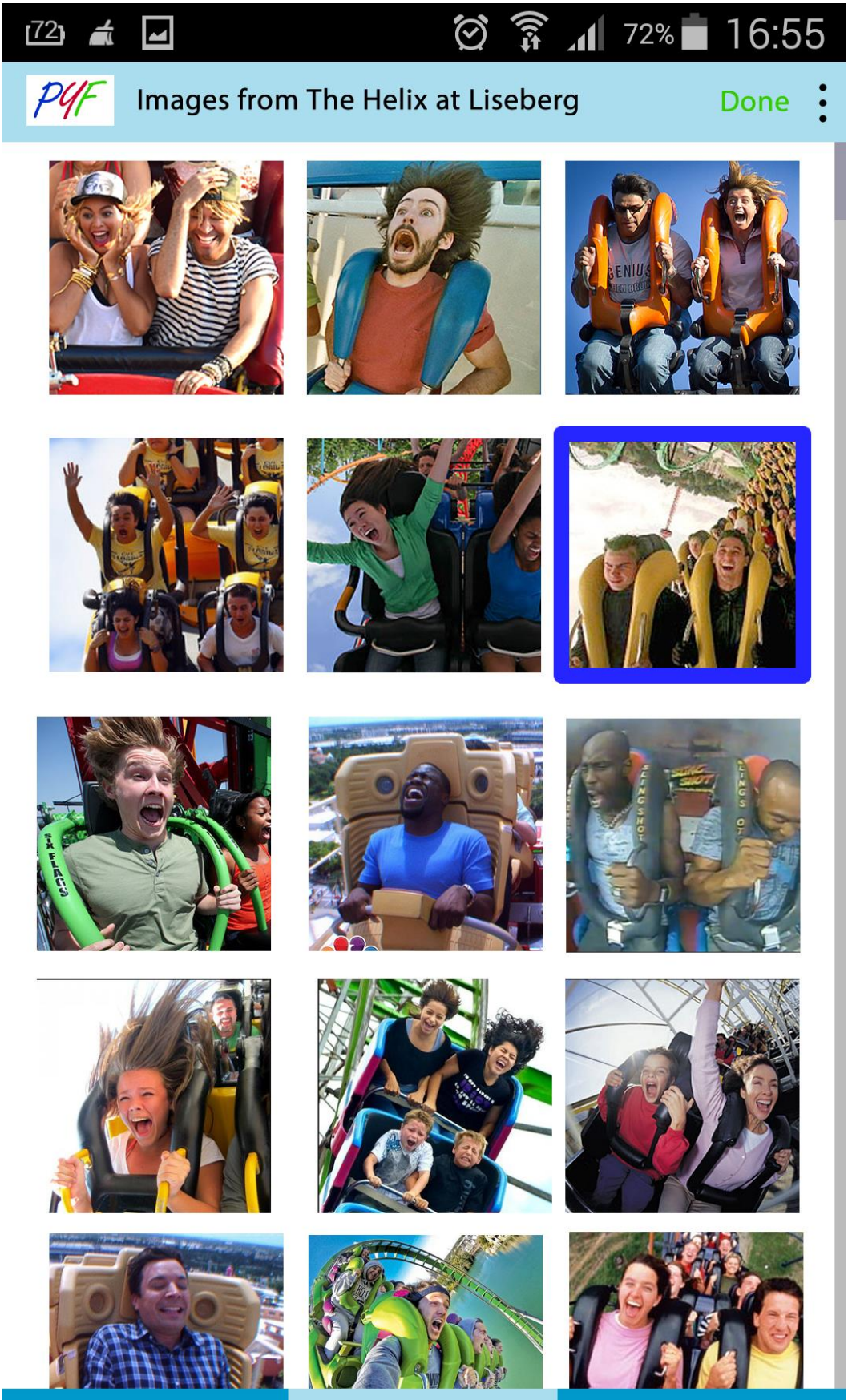
From time
int FromTime = DateTime.Now().Milliseconds()

To time
int ToTime (DateTime.Now() – Device.NumberOfMinutes).ToMilliSeconds()

When iterating and fetching more images, next batch the to time is calculated as follows, after doing the initial load for the gallery:

From time
To avoid fetching the last image twice or any other images for that matter is to take the DateTime for the last image that was fetched in the previous load sequence. This will give a new from time, new millisecond value to use for the filtering.
int FromTime = ImageListQuery.DateTime + 1

To time
int ToTime (DateTime.Now() – (Iteration * Device.NumberOfMinutes).ToMilliSeconds())


72
72%
16:55
Images from The Helix at Liseberg
Done
RATE
BUY
VIEW