Introduction

The Universal Windows Platform (UWP) will provide a guaranteed core API across devices. New adaptive controls and layout panels help to create UIs across a broad range of device screen resolutions and sizes, and respond to diverse inputs. The app could run on various Windows 10 devices such as PC, Tablet, Xbox, HoloLens, Surface Hub, and Internet of Things (IoT) devices.

All users will be able to visit the app store, then download and install it on a device. Once the app is launched, UI elements respond to the size and Dot Per Inch (DPI) of the screen the application is running on by adjusting their layout and scale. Depends on which type of the device, users could interact with multiple types of input such as keyboard, mouse, touch, pen, and Xbox controllers.

This is a critical concept of the UWP applications that delivers a common app platform on every device that runs Windows 10. Because UWP apps can run on a wide variety of devices with different form factors and types of input, UWP makes it possible to unlock the unique capabilities of each device with extension SDKs allow users to exploit the capabilities of specific devices.

System Requirements

The app can run on devices that have different forms of input, screen resolutions, and other unique characteristics. Here are main requirements that a device must have, so that users will be able to fully experience the functionalities.

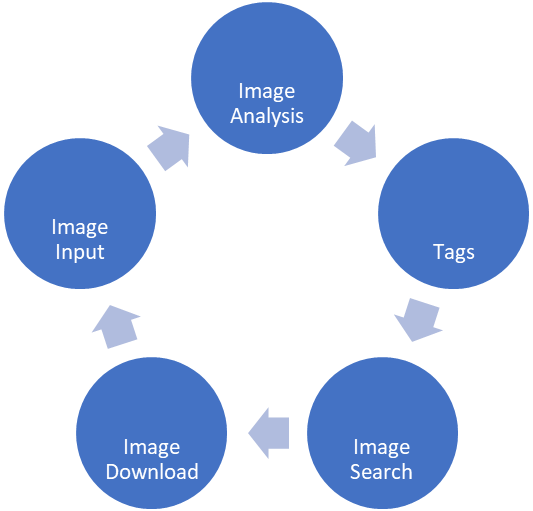
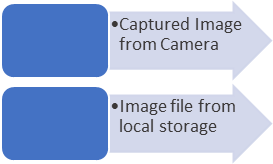
A device that runs Windows 10 to accommodate the app.

Any types of input that triggers events such as touch, mouse, or pen.

Camera and microphone devices to capture the media – Users will be asked permission to use webcam and microphone on a device.

File and folder system or Picture library – System IO provides the local storage to store and pick an image source.

Internet connection – 3G/4G Cellular data or Broadband network connection – The app needs to be connected to use Microsoft Cognitive services.



UI structure

For the app design to feel natural, it needs to be responsive. UWP provides panels that arrange content in grids, stack, and flows.

Command elements are the interactive UI elements that enable users to perform actions, such as sending an email, saving a file, or submitting form.

* Buttons in this app support the interactions
* Opening an image file from the local storage
* Saving image/s in the local storage
* Capturing an image from the camera preview stream
* Selecting items form the view panel/container
* Performing Microsoft Azure Cognitive Services

A screenshot of a cell phone

Description generated with very high confidence

Landing page will display a default UI with some panels

* Command Bar – The buttons will be placed, and users should be able to interact with the app by using multiple buttons.
* Image Preview Panel – Camera preview is displayed, and users can see the video stream from the camera.
* Analysis Result Panel – Two sub-panels are displayed. The top of this panel will show the result of image analysis that the app sends a request and receives a response from the Computer Vision Analysis service, and the bottom panel will display the selections what the user chooses from the top panel. Each TextBlock elements in the collection that will be dynamically changed by the result of image analysis.
* Search Result Panel – Multiple result of image search, depending on what selection of analysis result is chosen by users, it will be also changed dynamically.

A screenshot of a social media post

Description generated with very high confidence

Collections allow users to browse groups of contents or data.

Collection of Result – List view contains multiple text items. This categorized item control allows users to choose items, and the chosen ones will be displayed on the bottom panel.

Collection of Selection – List view that shows a list of selected items from the panel above. By clicking or tapping an item from the collection, users can unselect items.

Collection of Search Result – Grid view, like List view, however, this is primarily used to display multiple items which each item is a combination of image and text. This enables interactions with each item, users can see the details of an item, and then download it in the local storage.



Buttons allow users to interact with the app

Open File – Opening an image file from the local storage. This image source is the one of input methods that will be used to be analysed by Azure Computer Vision service.

Save – A primary saving method, that stores multiple images into the local storage, the collection of images is currently displayed on the Search Result Panel.

Tag – First it will check if there is an image loaded by opening an image file from the local storage, otherwise a captured image will be used as a source image, and then sends an Image Analysis request and receives a response. The result will be display on the top sub-panel of the Analysis Result Panel as a collection of tags.

Browse Photo – Based on the collection of selection, it sends a request of Image search service. The result will be distributed on the Grid view as a form of an image and lines of text. If no tags are chosen, it will make a collection of all tags, then sends a request.

Camera – Clean up the current image source, then starts a new preview video stream.

Save from an individual item – Users can also save an image from the selected item. It will open a save picker, so the user can choose a specific location and an image file format.

Features of the Implementation

Media capture

Camera initialisation – The app starts with the initialisation, this function will call the cleanup function. When the image source is flushed, it will start previewing.

Camera preview – After the initialisation, the app requests previewing, media capture properties and the size must be setup. Unauthorised camera device access will be denied by the system.

Cleanup image source – It will stop previewing immediately if the app is displaying the preview. Image source whether the media capture object or the software bitmap, will be disposed.

Capture image – Capture the current video frame then create the software bitmap, it will call a helper function to set this in the UI.

Load image – An image file picked by the user, is read as the file stream and the decoder creates the software bitmap. And a helper function will set this in the UI.

Save multiple images – The destination of the files, is pre-defined with PNG file format. Background downloader will start downloading each image from the URI and save it in the local storage.

Save single image – Unlike saving multiple images, it downloads an image asynchronously and saves it into the local storage where the user picked.

Helper

Set image view source – From the captured image or loaded image file, this will create the writeable bitmap and set as the image source in the UI.

Convert image source – To send a request of the image analysis or image search service, the image source must be converted as the array of byte, this helper function will read the image source as memory stream and read as the array buffer.

Request image analysis and image search – Http request header contains access key to use the image analysis or image search service, the client will send a request with the content and header. Response will be read as string so that can be deserialised.

Process result – Json string result will be converted into the view model that is used to bind as a collection of something in the List view or Grid view.

Update and clean list view content – By the triggered event, list view must be updated as its designed. The top panel of the Analysis Result Panel will be updating the list of tags. The bottom panel will be unselected which means removed from the List view.

Event trigger

All implementation of the events functions explained as how the buttons will interact and behave.

Current limitations and Future references