**Image Recognition with IBM Cloud Visual Recognition**

**Project Objective:**

The objective of this project is to create an image recognition system using IBM Cloud Visual Recognition that not only accurately identifies objects and scenes within images but also enhances user engagement and storytelling by generating AI captions for the recognized content. The system should provide an intuitive user interface, incorporating the principles of design thinking to create a seamless user experience.

**Design Thinking Process:**

**Empathize:** Understand the user's needs and challenges when dealing with images and their desire for engaging storytelling.

**Define:** Define the specific problem and objectives of the project, such as accurate image recognition and enhancing user engagement.

**Ideate:** Brainstorm creative ideas for the user interface, AI caption generation, and integration with IBM Cloud Visual Recognition.

**Prototype:** Create a preliminary design and technical plan, including a wireframe of the user interface and the system's architecture.

**Test:** Gather feedback through user testing and refine the design and functionality based on user input.

**Develop:** Implement the system, focusing on the technical implementation of IBM Cloud Visual Recognition and AI-generated captions.

**Deliver:** Launch the final product, monitor its performance, and make any necessary improvements.

**Development Phases:**

**Data Collection:** Collect a diverse dataset of images for training and testing the IBM Cloud Visual Recognition model.

**Training the Model:** Utilize IBM Cloud Visual Recognition to train a custom model to recognize specific objects and scenes within the images.

**User Interface Design:** Design an intuitive and user-friendly interface that allows users to upload images and view the recognition results and AI-generated captions.

**Technical Implementation:** Integrate IBM Cloud Visual Recognition into the system using the relevant APIs and SDKs. Implement a server to handle image uploads and communication with the IBM Cloud service.

**AI Caption Generation:** Develop a machine learning model for generating captions based on the recognized content within images. This may involve utilizing Natural Language Processing (NLP) techniques.

**Integration:** Seamlessly integrate the AI caption generation module with the IBM Cloud Visual Recognition module for a unified user experience.

**User Engagement:** Implement features to enhance user engagement, such as allowing users to customize or edit generated captions, share images and captions on social media, or create visual stories.

**User Interface:**

The user interface should be intuitive and visually appealing. Users should be able to easily upload images, view recognition results, and read AI-generated captions. Additionally, it can include interactive features for users to engage with the recognized content, such as commenting and sharing.

**Technical Implementation Details:**

* Utilize IBM Cloud Visual Recognition API for image recognition.
* Develop a web-based user interface using HTML, CSS, and JavaScript.
* Implement a server using a web framework (e.g., Node.js or Django) to handle image uploads and communicate with IBM Cloud services.
* Create a machine learning model for AI caption generation, which can be integrated into the system.

**Integration of IBM Cloud Visual Recognition:**

Integrate IBM Cloud Visual Recognition by making API calls to analyze images. This can involve sending images to the service, receiving recognition results (labels, objects, scenes), and displaying them to the user.

**AI-Generated Captions for User Engagement and Storytelling:**

**Personalized Storytelling:** Captions can provide unique and creative descriptions of images, allowing users to tell their stories in a more compelling and personal manner.

**Accessibility:** Captions make visual content accessible to individuals with visual impairments, improving inclusivity.

**SEO Benefits:** Captions can improve search engine optimization (SEO) by providing textual content that can be indexed by search engines.

**User Interaction:** Users can interact with captions by editing, sharing, or using them as a starting point for their own narratives, fostering engagement.

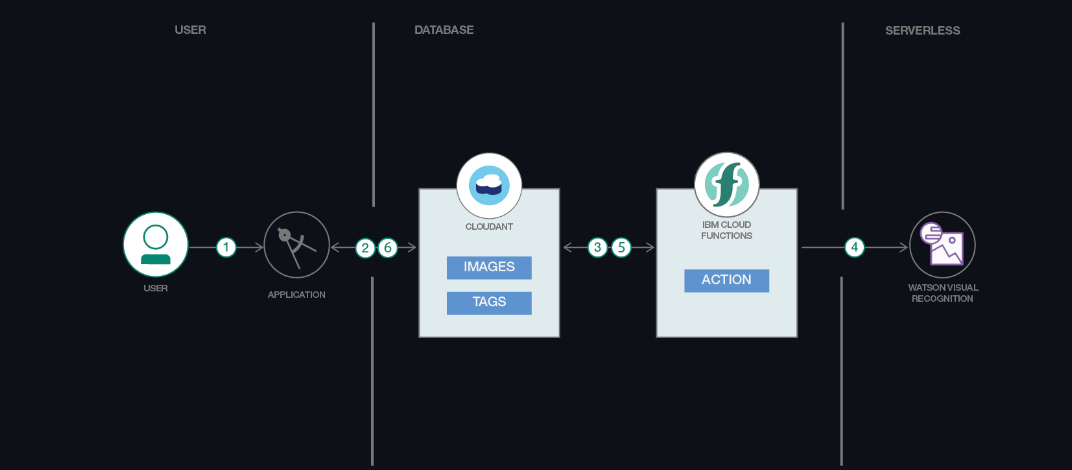
**Contextual Information:** Captions provide additional context, helping users better understand the content and fostering deeper connections.

The application demonstrates an IBM Cloud Functions (based on Apache OpenWhisk) that gets an image from the Cloudant database and classifies it through Watson Visual Recognition. The use case demonstrates how actions work with data services and execute logic in response to Cloudant events.

One function, or action, is triggered by changes (in this use case, an upload of a document) in a Cloudant database. These documents are piped to another action that submits the image to Watson Visual Recognition and uploads a new document in Cloudant with the classifiers produced by Watson.

When the reader has completed this Code Pattern, they will understand how to:

* Create and Deploy Cloud Functions
* Trigger Cloud Functions with Cloudant changes
* Use Watson Image Recognition with Cloud Functions



1. The user chooses a picture from the gallery.
2. The image is stored in the Cloudant database.
3. Cloud Function is triggered when there's a new image in the database.
4. Cloud Function gets the image and uses Watson's Visual Recognition to process the image.
5. Cloud Function stores the results (classes with scores) from Visual Recognition in the database.
6. The user can see the new tags or classes in the image they uploaded.

**Components included :**

**IBM Cloud Functions (powered by Apache OpenWhisk):** Execute code on demand in a highly scalable, serverless environment.

**Cloudant:** A fully managed data layer designed for modern web and mobile applications that leverages a flexible JSON schema.

**Watson Visual Recognition:** Visual Recognition understands the contents of images - visual concepts tag the image, find human faces, approximate age and gender, and find similar images in a collection.

**Steps :**

1. **Clone the repo**

Clone the serverless-image-recognition locally. In a terminal, run:

$ git clone <https://github.com/IBM/serverless-image-recognition>

1. **Create IBM Cloud Services**

Create a Cloudant instance and use legacy credentials and IAM for the Available authentication method option.

* Create credentials for this instance and copy the username and password in the local.env file in CLOUDANT\_USERNAME and CLOUDANT\_PASSWORD.
* Launch the Cloudant web console and create a database named images and tags. Create Cloudant credentials using the IBM Cloud dashboard and place them in the local.env file.

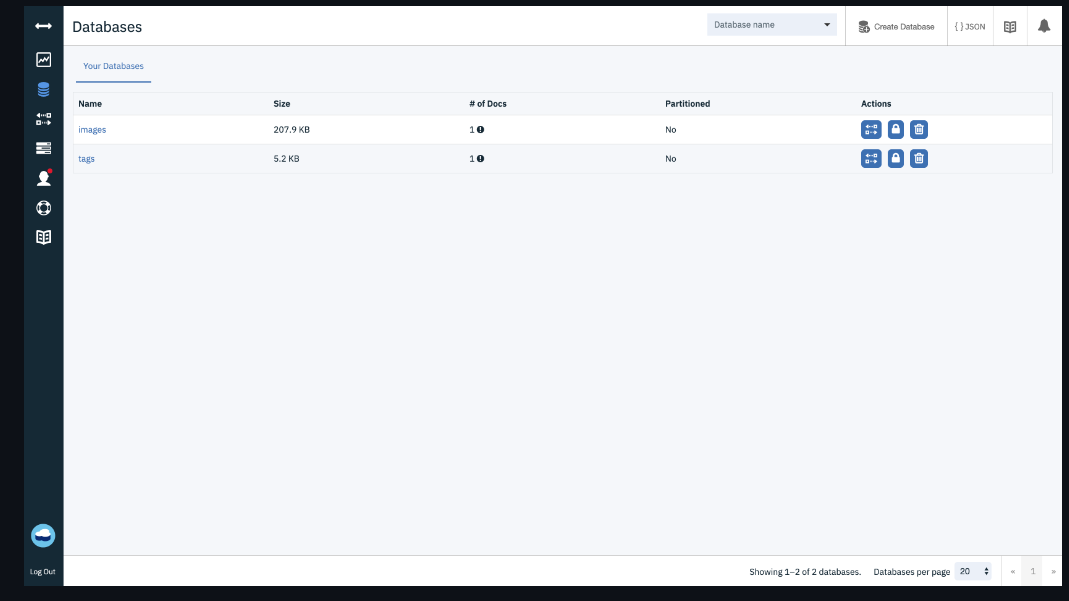
Create a Watson Visual Recognition instance.

* Copy the API Key in the Credentials section and paste it into the local.env file in the value of WATSON\_VISUAL\_APIKEY

1. **Deploy Cloud Functions**

Create 2 databases in Cloudant:

* images
* tags



1. **Deploy through the IBM Cloud Functions console user interface**

**Deploy using the wskdeploy command line tool**

This approach deploys the Cloud Functions with one command driven by the runtime-specific manifest file available in this repository.

Make sure you have the right environment variables in the local.env file. Export them in your terminal then deploy the Cloud Functions using wskdeploy. This uses the manifest.yaml file in this root directory.

* $ source local.env
* $ wskdeploy

1. **Launch Application**

**Configure web/scripts/upload.js. Modify the lines for your Cloudant credentials.**

let usernameCloudant = "YOUR\_CLOUDANT\_USERNAME"

let passwordCloudant = "YOUR\_CLOUDANT\_PASSWORD"

**Run the Electron app or open the html file.**

Electron:

$ npm install

$ npm start

**Sample Output :**

