

# Deep Learning Fundus Image Analysis For Early Detection of Diabetic Retinopathy

## 1 INTRODUCTION

### 1.1 Overview

#### **A brief description about your project**

The project "Deep Learning Fundus Image Analysis for Early Detection of Diabetic Retinopathy" focuses on leveraging deep learning techniques to analyze fundus images for the early detection of diabetic retinopathy (DR). Diabetic retinopathy is a diabetes-related eye condition that can lead to vision impairment or blindness if not diagnosed and treated in its early stages.

Fundus images capture the back of the eye, including the retina, blood vessels, and optic nerve head. Analyzing these images manually can be time-consuming, and early signs of diabetic retinopathy may be subtle. Deep learning, a subset of artificial intelligence, offers a promising solution by automating the image analysis process and providing efficient and accurate results.

The ultimate goal of the project is to contribute to the early detection and management of diabetic retinopathy, thereby preventing vision loss and improving patient outcomes. The use of deep learning in fundus image analysis holds significant potential to enhance the efficiency and accuracy of diabetic retinopathy screening programs.

### 1.2 Purpose

Using Deep Learning Fundus Image Analysis for Early Detection of Diabetic Retinopathy project involves the application of deep learning techniques to analyze fundus images with the goal of identifying signs of diabetic retinopathy (DR) at an early stage. Diabetic retinopathy is a common complication of diabetes that affects the blood vessels in the retina and can lead to vision loss if not detected and treated early.

#### **Early Detection of Diabetic Retinopathy:**

The primary goal of this project is to achieve early detection of diabetic retinopathy. Deep learning algorithms can be trained to analyze fundus images for specific features and abnormalities associated with diabetic retinopathy, allowing for early intervention and treatment.

#### **Automation of Screening Process:**

Traditional methods of diabetic retinopathy screening involve manual examination of fundus images by trained professionals. Deep learning models can automate this process, reducing the burden on healthcare professionals and speeding up the screening process.

#### **Increased Efficiency and Scalability:**

With the use of deep learning, the analysis of a large number of fundus images can be performed rapidly and efficiently. This scalability is particularly beneficial in regions with a high prevalence of diabetes, where a large number of screenings may be required.

**Resource Optimization:**

By automating the screening process, healthcare resources can be optimized. Trained professionals can focus on more complex cases and treatments, while the deep learning model handles routine screenings.

**Prevention of Vision Loss:**

Early detection and intervention are crucial in preventing vision loss due to diabetic retinopathy. Timely identification of abnormalities in fundus images allows for early treatment, which can significantly improve the outcomes for patients.

**Telemedicine and Remote Monitoring:**

Deep learning-based analysis of fundus images can be integrated into telemedicine platforms, allowing for remote screening and monitoring of diabetic retinopathy. This is especially useful for patients in rural or underserved areas who may have limited access to eye care specialists.

**Data-Driven Insights:**

The project generates valuable data that can be used for further research and insights into diabetic retinopathy. This data can contribute to a better understanding of the disease, refinement of models, and improvement of diagnostic accuracy over time.

**Customization and Adaptability:**

Deep learning models can be trained on diverse datasets, allowing for customization based on different demographics, ethnicities, and variations in fundus image characteristics. This adaptability enhances the model's performance across various populations.

The use of deep learning for fundus image analysis in diabetic retinopathy detection offers a transformative approach to improve the efficiency, accuracy, and accessibility of screening, ultimately leading to better patient outcomes and the prevention of vision loss.

## **2 LITERATURE SURVEY**

### **2.1 Existing problem**

There are still some existing challenges and problems associated with existing approaches or methods. Here are some common issues:

**Limited Diversity in Datasets:**

Many existing datasets used for training deep learning models in diabetic retinopathy may lack diversity in terms of patient demographics, ethnicities, and variations in image quality. This can result in models that may not generalize well to different populations.

**Interpretability and Explainability:**

Deep learning models, especially complex neural networks, are often considered as "black boxes," making it challenging to understand how they arrive at a particular decision. The lack of interpretability and explainability can hinder the trust that healthcare professionals place in these models.

#### **Class Imbalance:**

In medical datasets, the prevalence of diabetic retinopathy may be low compared to non-diabetic cases. Class imbalance can lead to a biased model that performs well on the majority class but struggles with minority classes, potentially leading to missed cases of diabetic retinopathy.

#### **Robustness to Variability in Image Quality:**

Fundus images can vary widely in terms of quality, resolution, and illumination. Existing models may not be robust enough to handle such variability, leading to reduced performance in real-world scenarios where image quality may not be consistent.

#### **Integration into Clinical Workflow:**

The successful implementation of deep learning models for diabetic retinopathy detection requires seamless integration into the existing clinical workflow. Issues related to user interface design, compatibility with electronic health records (EHRs), and acceptance by healthcare professionals need to be addressed for practical adoption.

#### **Data Privacy and Security:**

Fundus images contain sensitive patient information, and the deployment of deep learning models in healthcare settings raises concerns about data privacy and security. Ensuring compliance with regulations and protecting patient confidentiality is crucial.

#### **Limited Generalization to Other Eye Diseases:**

Some models may be specifically designed for diabetic retinopathy detection, but they may not generalize well to other eye diseases or conditions. A holistic approach that considers multiple eye diseases could enhance the overall utility of such models.

#### **Need for External Validation:**

Many studies may report impressive results on the same datasets they were trained on, but external validation on independent datasets is essential to evaluate the model's true generalization performance.

#### **Cost and Accessibility:**

Implementing deep learning solutions may involve high initial costs, both in terms of infrastructure and training. Accessibility to such technology in resource-constrained environments or smaller healthcare facilities could be a challenge.

Addressing these challenges is crucial for the successful deployment and widespread adoption of deep learning models for fundus image analysis in the early detection of diabetic retinopathy. Ongoing research and collaboration between computer scientists, clinicians, and other stakeholders are essential to overcome these issues and improve the reliability and effectiveness of these approaches.

## **2.2 Proposed solution**

Diverse and Representative Datasets:

Collect and curate diverse datasets that represent different demographics, ethnicities, and variations in image quality. Ensure that the dataset is well-balanced to address class imbalance issues.

Transfer Learning and Robust Model Architectures:

Leverage transfer learning techniques to pre-train models on large datasets and fine-tune them for diabetic retinopathy detection. Design robust model architectures that can handle variations in image quality and are less sensitive to noise.

Integration with Clinical Workflow:

Collaborate closely with healthcare professionals to integrate the deep learning solution seamlessly into the clinical workflow. Ensure compatibility with electronic health records (EHRs) and design user- friendly interfaces for easy adoption by healthcare providers.

Addressing Data Privacy and Security:

Cost-Effective Infrastructure:

Explore cost-effective solutions for model deployment and infrastructure, such as cloud-based services. Consider the scalability of the solution to accommodate varying workloads and resource constraints in different healthcare settings.

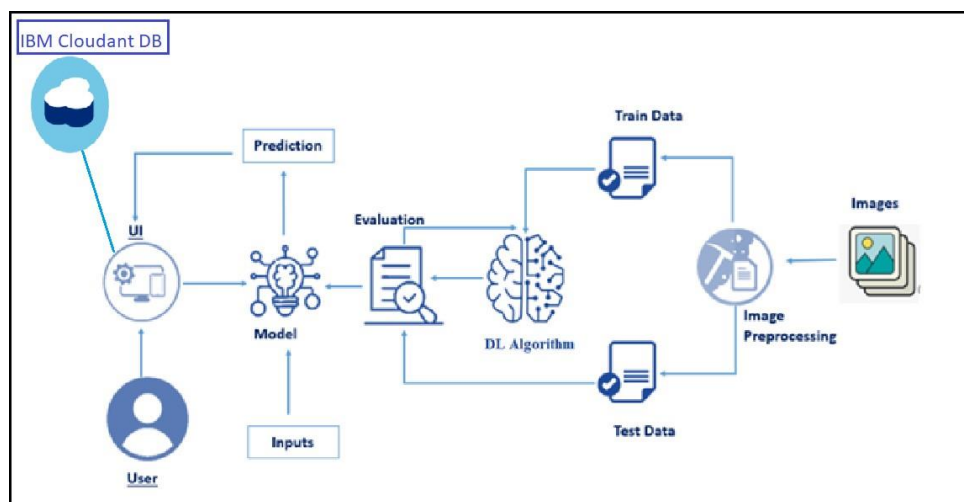
Holistic Approach to Eye Diseases:

Design models that can generalize to multiple eye diseases, promoting a holistic approach to eye care. This can enhance the utility of the solution in diverse clinical scenarios.

### 3 THEORITICAL ANALYSIS

#### 3.1 Block diagram

Block Diagram of



- Data Collection.

- Create a Train and Test path.
- Data Pre-processing.
  - Import the required library
  - Configure ImageDataGenerator class
  - Apply ImageDataGenerator functionality to Trainset and Testset
- Model Building
  - Pre-trained CNN model as a Feature Extractor
  - Adding Dense Layer
  - Configure the Learning Process
  - Train the model
  - Save the Model
  - Test the model
- Cloudant DB
  - Register & Login to IBM Cloud
  - Create Service Instance
  - Creating Service Credentials
  - Launch Cloudant DB
  - Create Database
- Application Building
  - Create an HTML file
  - Build Python Code

### 3.2 Hardware / Software designing

Following software's, concepts and packages are require to complete the project.

- ✚ Operating System: Windows
- ✚ Application Software: Anaconda navigator
- ✚ IDE: Spyder and Jupyter notebook, Google Colab
- ✚ Python packages:
  - Open anaconda prompt as administrator
  - Type “pip install numpy” and click enter.
  - Type “pip install pandas” and click enter..
  - Type “pip install tensorflow==2.3.2” and click enter.
  - Type “pip install keras==2.3.1” and click enter.
  - Type “pip install Flask” and click enter.
- ✚ IBM Cloud account to create Database in the cloud

Following Hardware components e require to complete the project.

- ✚ Processor: i5 and above

🚦 RAM: 8GB and above

🚦 Hard Disk: 256 and above

## **4 EXPERIMENTAL INVESTIGATIONS**

Data Collection and Preprocessing:

Source of Fundus Images:

Downloaded images from kaggle repository. Clearly define the source of fundus images, ensuring they include a diverse set of cases related to diabetic retinopathy.

Dataset Preprocessing:

Preprocess the fundus images by resizing, normalizing pixel values, and augmenting the dataset for increased diversity.

Model Selection and Architecture:

Deep Learning Architecture:

Choose the Xception architecture for transfer learning, leveraging its depth and expressive power.

Customize the top layers of the Xception model to suit the binary classification task for diabetic retinopathy detection.

Transfer Learning:

Load the pretrained Xception model weights from ImageNet.

Freeze the convolutional base layers to retain the knowledge learned from ImageNet while updating the dense layers for diabetic retinopathy detection.

Experiment with hyperparameter tuning, including learning rates, batch sizes, and regularization techniques, specifically adapted for the Xception architecture.

Transfer Learning Fine-Tuning:

Fine-tune the top layers of the Xception model on the diabetic retinopathy dataset.

Monitor and control overfitting with techniques such as dropout or weight regularization.

5. Evaluation Metrics:

Performance Metrics:

Choose evaluation metrics such as accuracy

Interpretability and Explainability:

Xception Model Interpretation:

Utilize techniques for interpreting Xception model predictions, including visualizing feature maps or attention mechanisms specific to the Xception architecture.

Understanding the model is a very important phase to properly use it for training and prediction purposes.

Keras provides a simple method, summary to get the full information about the model and its layers.

```
In [54]: # View the Structure of the model
model.summary()

block14_sepconv2 (Separable Conv2D) (None, 10, 10, 2048) 3159552 ['block14_sepconv1_act[0][0]']
eConv2D)

block14_sepconv2_bn (Batch Normalization) (None, 10, 10, 2048) 8192 ['block14_sepconv2[0][0]']

block14_sepconv2_act (Activation) (None, 10, 10, 2048) 0 ['block14_sepconv2_bn[0][0]']

flatten (Flatten) (None, 204800) 0 ['block14_sepconv2_act[0][0]']

dense (Dense) (None, 5) 1024005 ['flatten[0][0]']

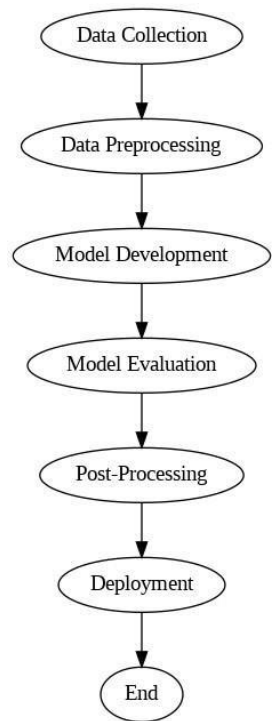
=====
Total params: 21885485 (83.49 MB)
Trainable params: 1024005 (3.91 MB)
Non-trainable params: 20861480 (79.58 MB)
```

The model is trained for 3 epochs with 115 steps per each epoch and after every epoch, the current model state is saved if the model has the least loss encountered till that time. We can see that the training loss decreases in almost every epoch and probably there is further scope to improve the model. Final Accuracy of the model after training is 74.49, further it can be improved by using different models.

```
In [56]: # model.fit(
# training_set,
# epochs=3,
# steps_per_epoch = 115,
# validation_steps=1, verbose=1)

Epoch 1/3
WARNING:tensorflow:From C:\Users\admin\anaconda3\Lib\site-packages\keras\src\utils\tf_utils.py:492: The name tf.ragged.RaggedTensorValue is deprecated. Please use tf.compat.v1.ragged.RaggedTensorValue instead.
WARNING:tensorflow:From C:\Users\admin\anaconda3\Lib\site-packages\keras\src\engine\base_layer_utils.py:384: The name tf.executing_eagerly_outside_functions is deprecated. Please use tf.compat.v1.executing_eagerly_outside_functions instead.
115/115 [=====] - 797s 7s/step - loss: 4.6245 - accuracy: 0.6576
Epoch 2/3
115/115 [=====] - 2115s 18s/step - loss: 3.3966 - accuracy: 0.7196
Epoch 3/3
115/115 [=====] - 727s 6s/step - loss: 3.1416 - accuracy: 0.7449
```

5 FLOWCHART



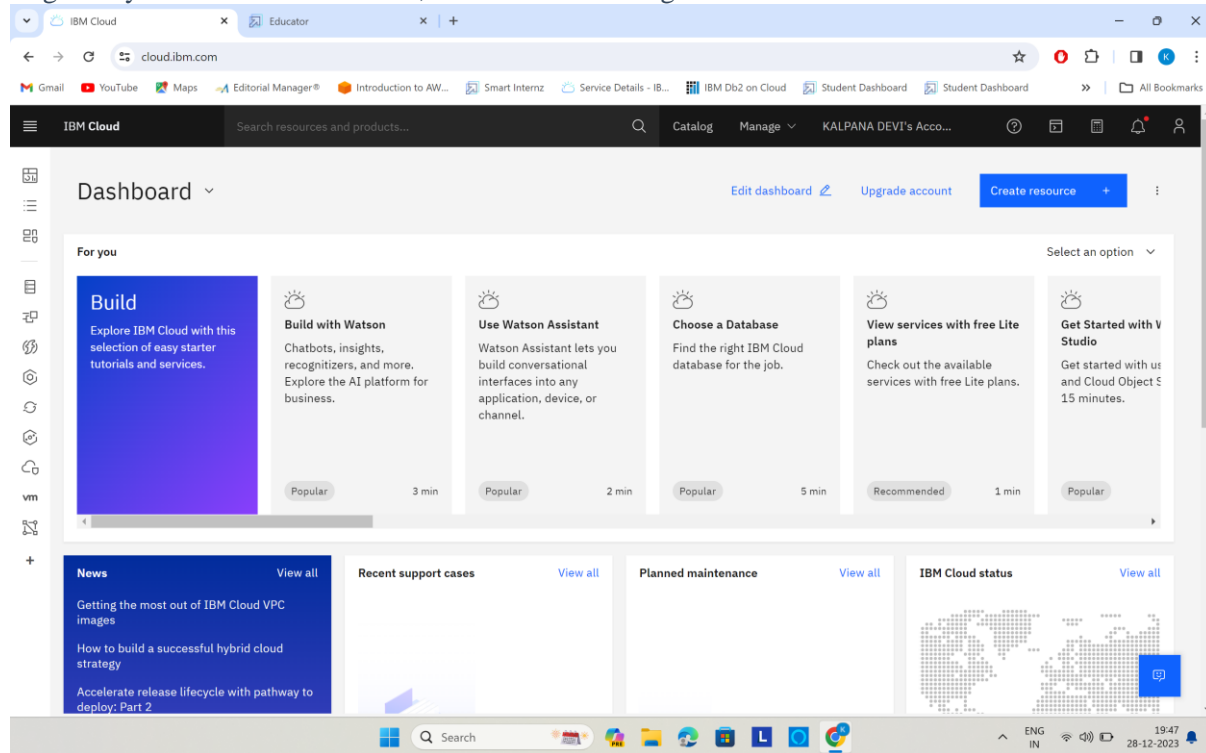
## 5 Register & Login To IBM Cloud

1.Register To IBM Cloud:- [Link](#)

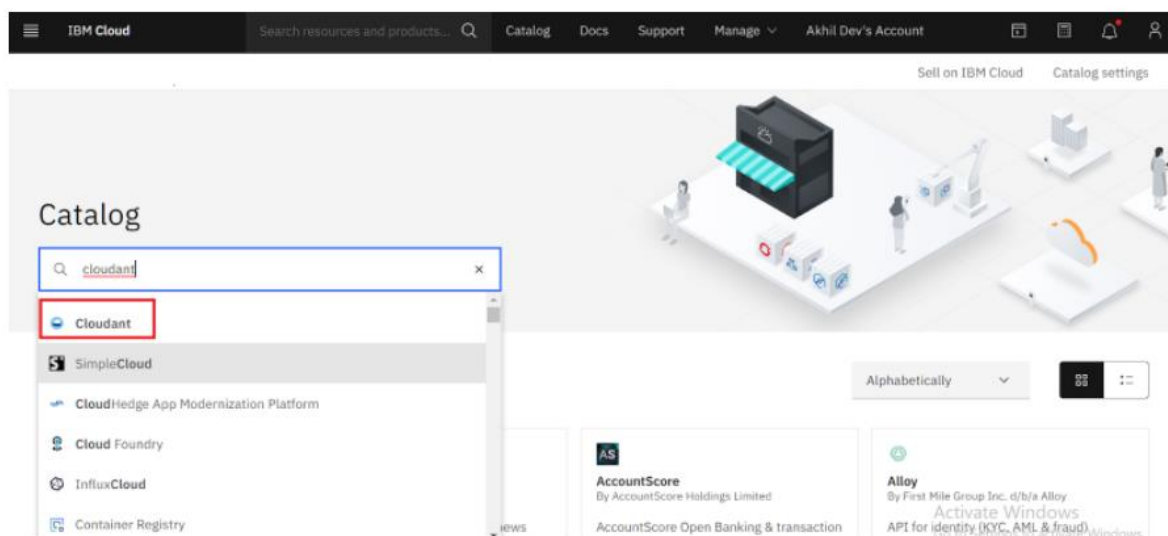
2.Sign in with your credentials: [Link](#)

## Create Service Instance

Log in to your IBM Cloud account, and click on Catalog

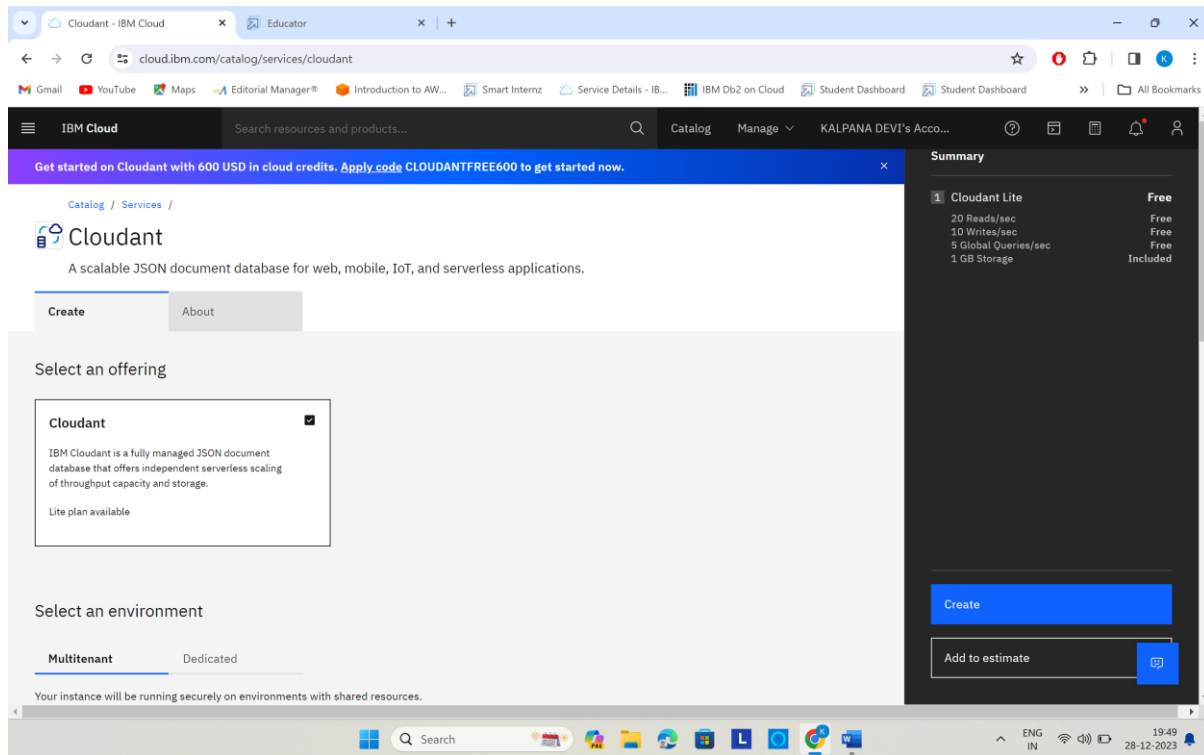


Type Cloudfant in the Search bar and click to open it.

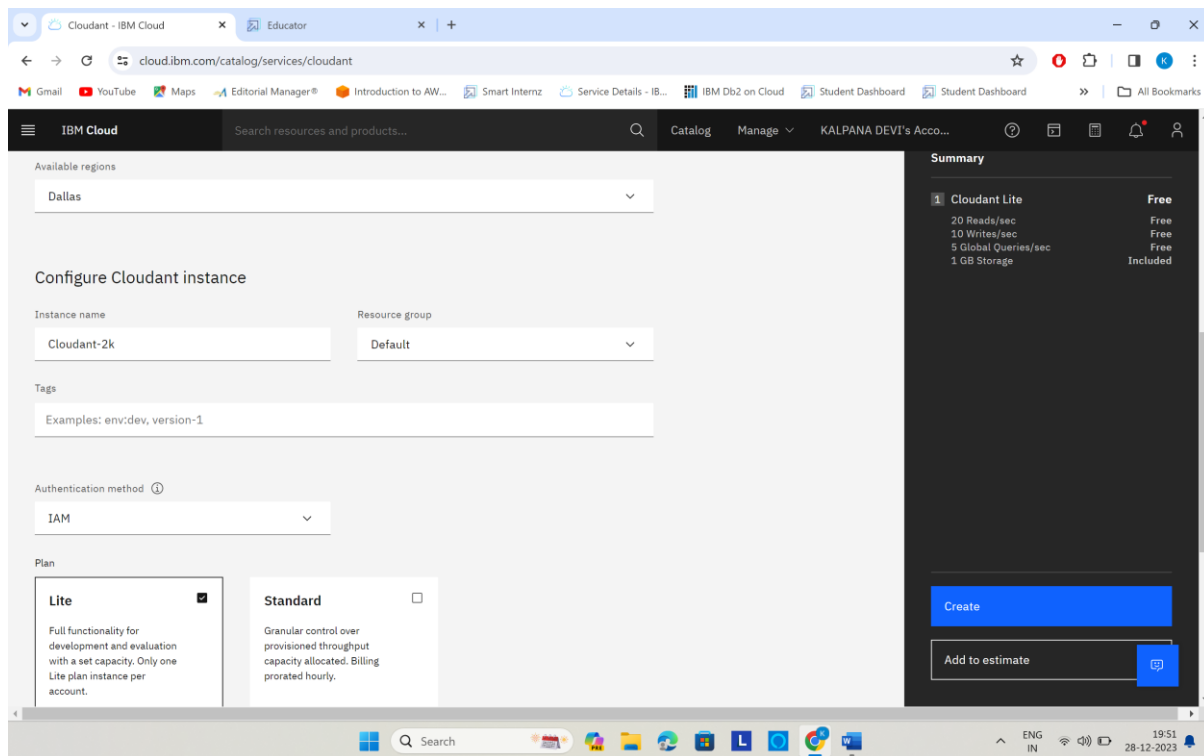


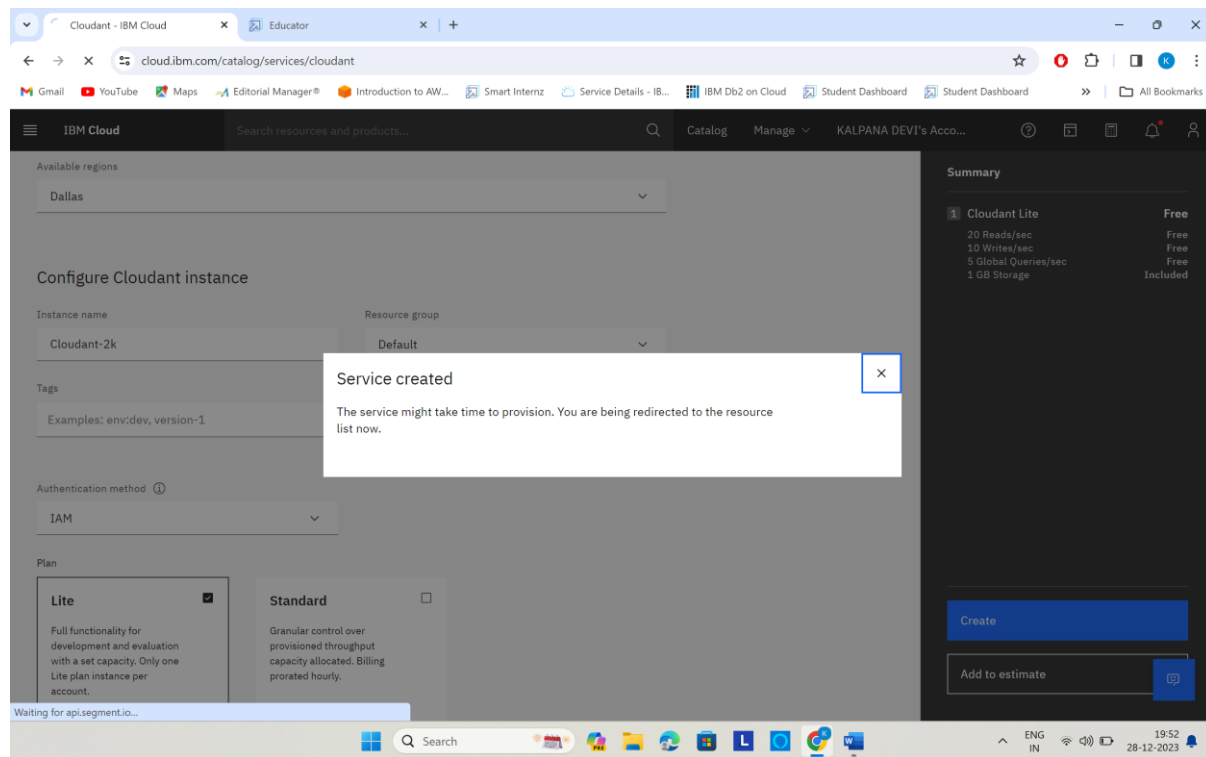
Select an offering and an environment



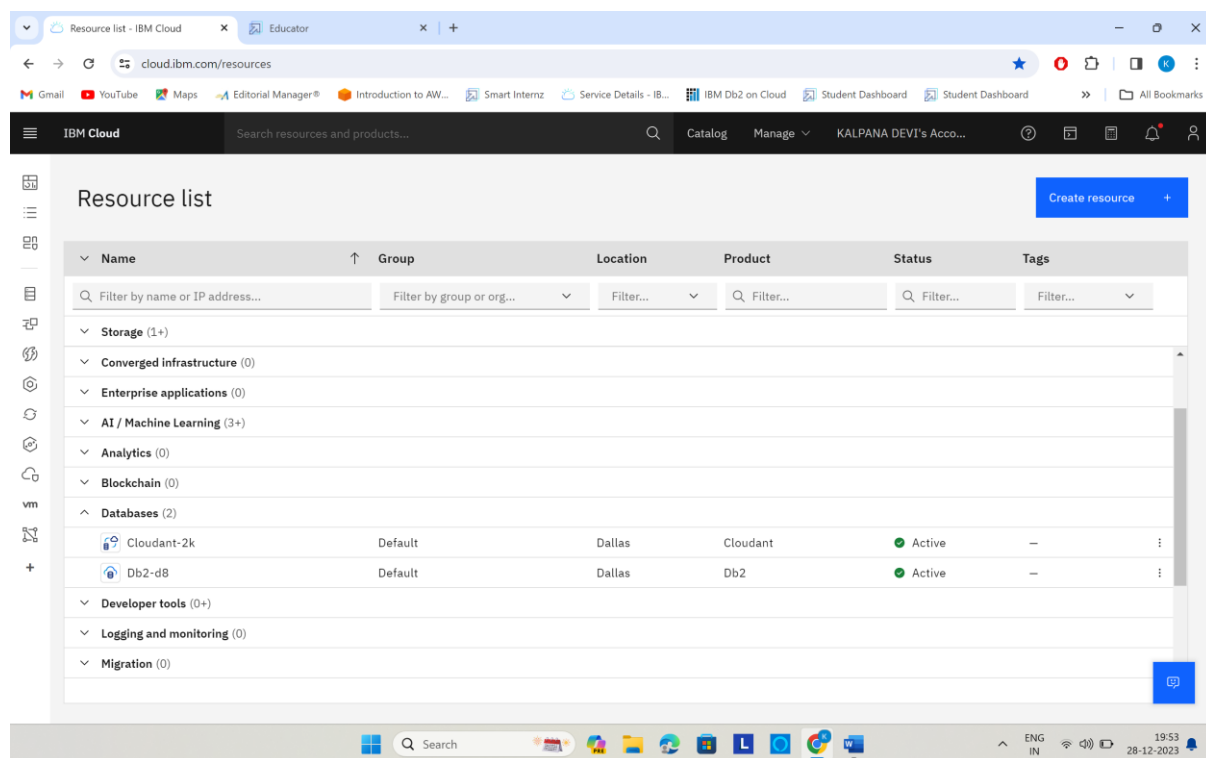


- Select region as Dallas & Type an instance name then click on create service.





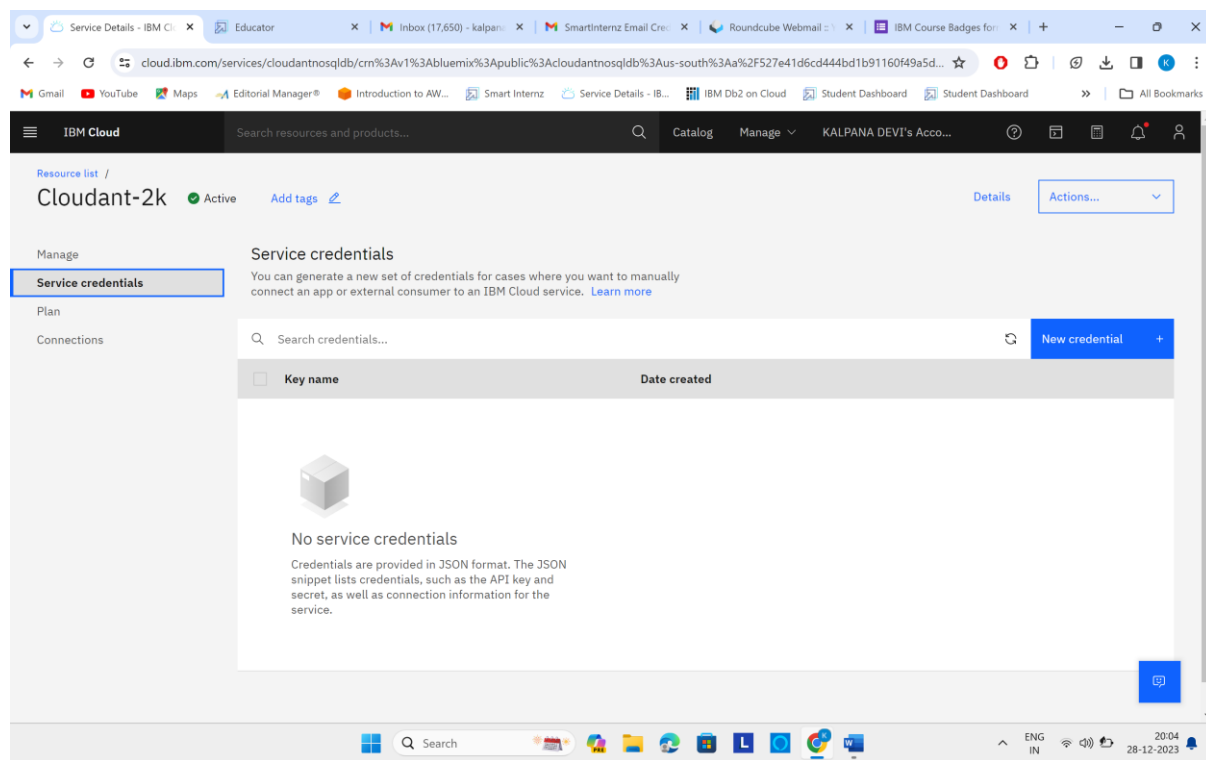
After click create the system displays a message to say that the instance is being provisioned, which returns you to the Resource list. From the Resource list, you see that the status for your instance is, Provision in progress.



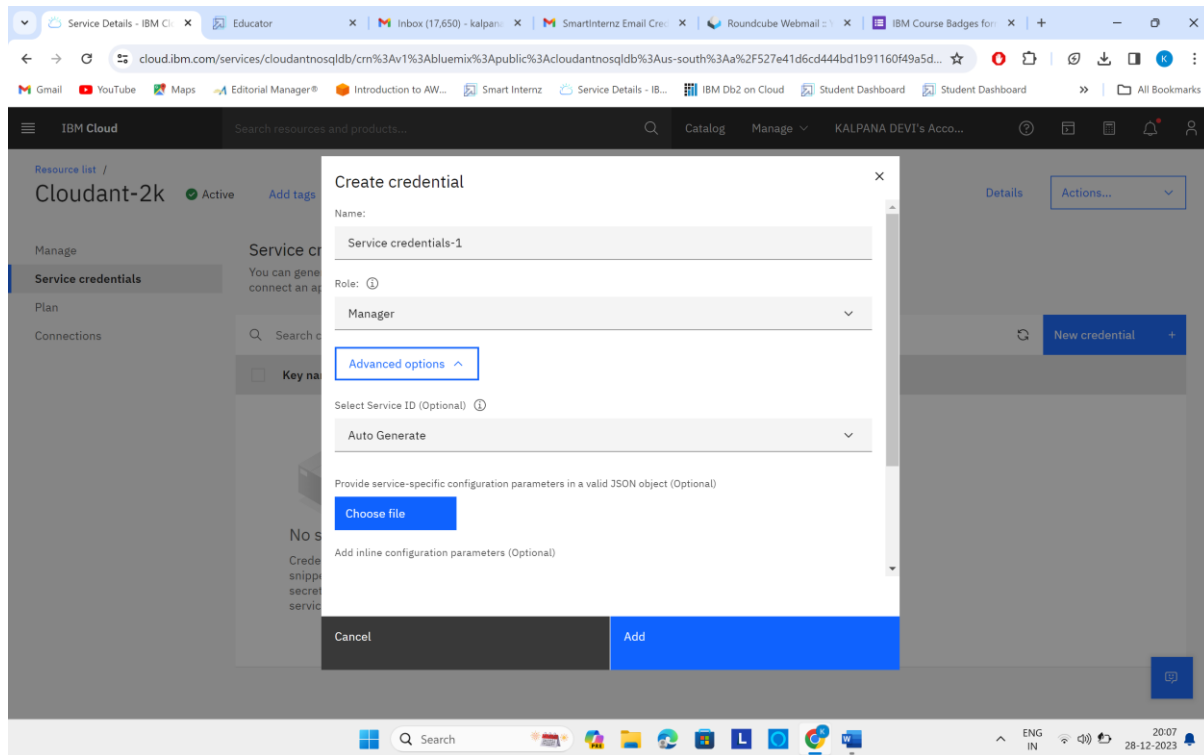
- When the status changes to Active, click the instance.

## Creating Service Credentials

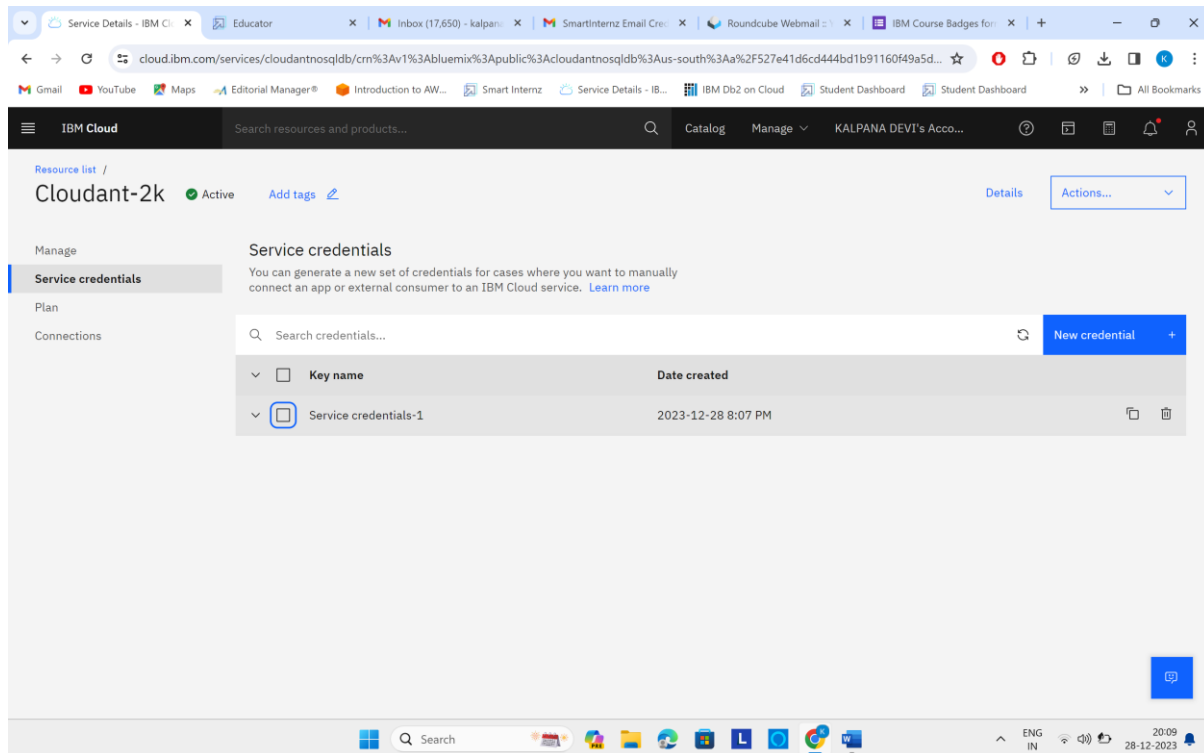
1.To create the connection information that your application needs to connect to the instance, click New credential.



2. Enter a name for the new credential in the Add new credential window.
3. Accept the Manager role.
4. (Optional) Create a service ID or have one automatically generated for you.
5. (Optional) Add inline configuration parameters. This parameter isn't used by IBM Cloudant service credentials, so ignore it.
6. Click Add.



7. To see the credentials that are required to access the service, click the chevron.



8. The details for the service credentials open like the following example:

Service credentials

You can generate a new set of credentials for cases where you want to manually connect an app or external consumer to an IBM Cloud service. [Learn more](#)

Search credentials...

New credential +

Key name Date created

Service credentials-1 2023-12-28 8:07 PM

```
{
  "apikey": "qniiE7X0aZyLHmJTpFZhJdg89-QFJcgUj6VN-yHd_1Vi",
  "host": "c19fe547-1724-4d1b-9e35-2f5a73593de5-bluemix.cloudantnosqldb.appdomain.cloud",
  "iam_apikey_description": "Auto-generated for key crn:v1:bluemix:public:cloudantnosqldb:us-south:a/527e41d6cd44bd1b91160f49a5d4753:04610423-ceac-41a8-aaf2-a6f09cf56c4e:resource-key:e9b2fffl-d896-48a5-ab49-760ea591b60c",
  "iam_apikey_id": "ApiKey-c0f76b18-a800-49b7-bf2b-e2a584dd1315",
  "iam_apikey_name": "Service credentials-1",
  "iam_role_crn": "crn:v1:bluemix:public:iam:::serviceRole:Manager",
  "iam_serviceid_crn": "crn:v1:bluemix:public:iam-identity::a/527e41d6cd44bd1b91160f49a5d4753::serviceid:ServiceId-90741447-8607-4df9-a3e1-c986d69423b7",
  "url": "https://c19fe547-1724-4d1b-9e35-2f5a73593de5-bluemix.cloudantnosqldb.appdomain.cloud",
  "username": "c19fe547-1724-4d1b-9e35-2f5a73593de5-bluemix"
}
```

## Launch Cloudant DB

Overview Capacity Docs

Launch Dashboard

Deployment details

CRN crn:v1:bluemix:public:cloudantnosqldb:us-south:a/527e41d6cd44bd1b91160f49a5d4753:04610423-ceac-41a8-aaf2-a6f09cf56c4e::

Location Dallas

External endpoint <https://c19fe547-1724-4d1b-9e35-2f5a73593de5-bluemix.cloudant.com>

External endpoint (preferred) <https://c19fe547-1724-4d1b-9e35-2f5a73593de5-bluemix.cloudantnosqldb.appdomain.cloud>

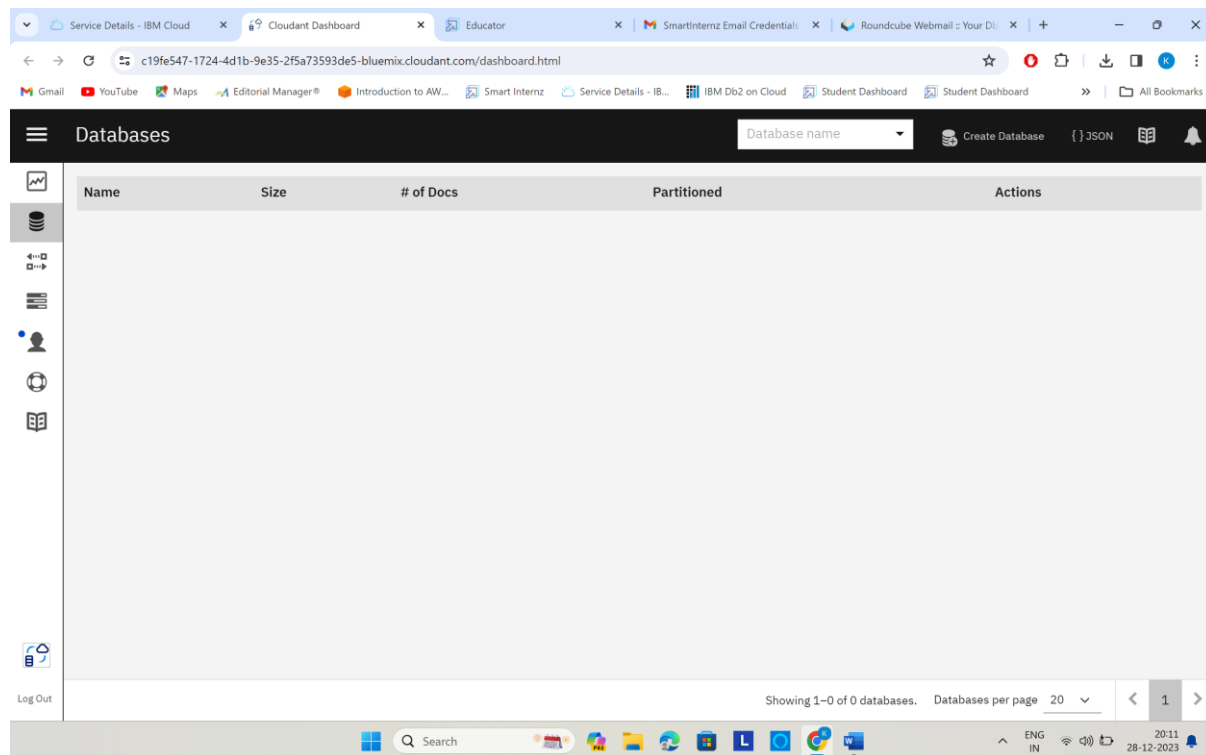
Authentication methods [IBM Cloud IAM](#)

Activity Tracker event types Management Save

Disk encryption Yes. Automatically generated disk encryption key.

Your Cloudant DB launches

Note: If You are a New User you will find empty database

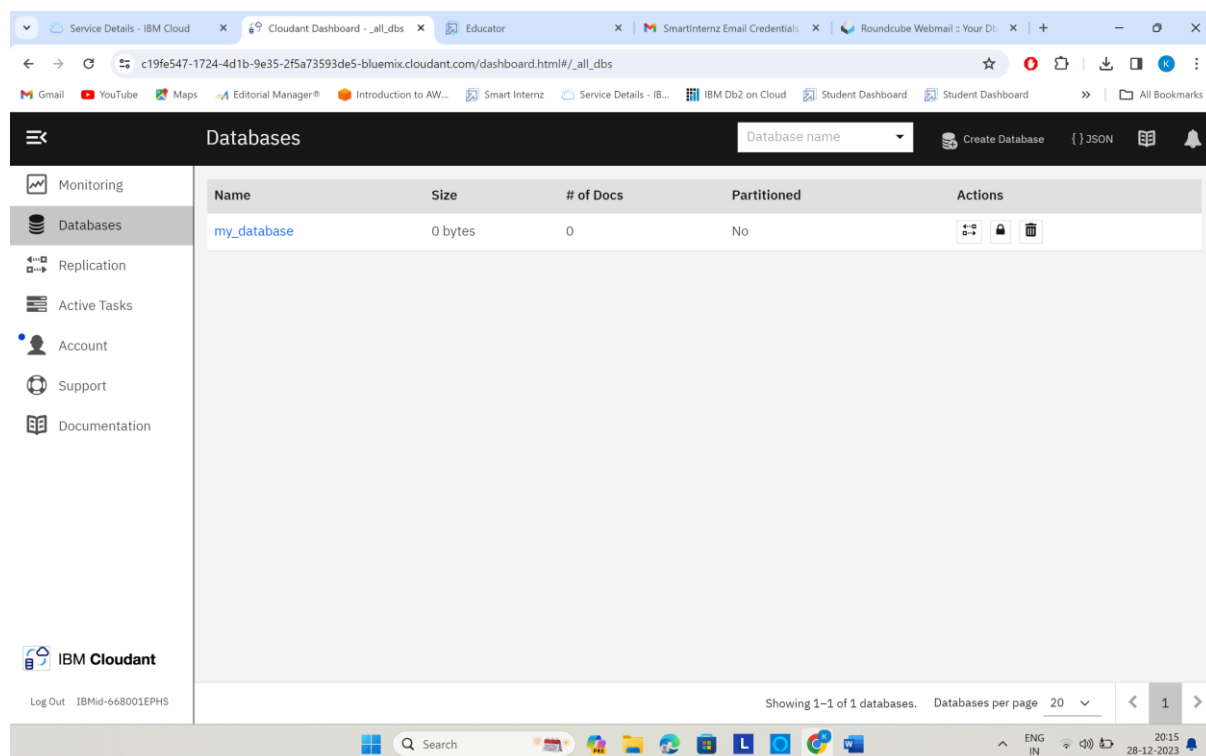


Let's create the Database Now

## Create Database

### Activity 5:- Create Database

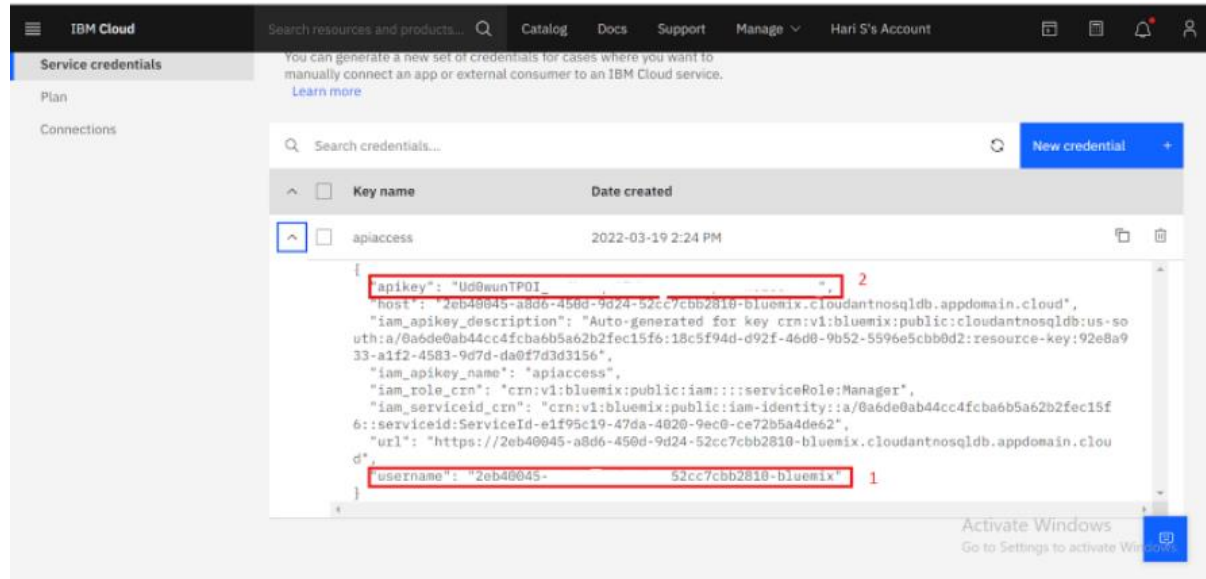
In order to manage a connection from a local system you must first initialize the connection by constructing a Cloudant client. We need to import the cloudant library.



```
from cloudant.client import Cloudant
```

IBM Cloud Identity & Access Management enables you to securely authenticate users and control access to all cloud resources consistently in the IBM Bluemix Cloud Platform.

In the above `cloudant.iam()` method we have to give username & apikey to build the connection with cloudant DB.



- Once a connection is established you can then create a database, open an existing database.

## 6 RESULT

### Home Page:



## Upload Image:

← → ↺ 127.0.0.1:5000 🔍 ☆ 🗑️


Early Detection Of Diabetic Retinopathy Using Deep Learning

### Diabetic Retinopathy Classification:


The project "Deep Learning Fundus Image Analysis for Early Detection of Diabetic Retinopathy" focuses on leveraging deep learning techniques to analyze fundus images for the early detection of diabetic retinopathy (DR). Diabetic retinopathy is a diabetes-related eye condition that can lead to vision impairment or blindness if not diagnosed and treated in its early stages. Fundus images capture the back of the eye, including the retina, blood vessels, and optic nerve head. Analyzing these images manually can be time-consuming, and early signs of diabetic retinopathy may be subtle. Deep learning, a subset of artificial intelligence, offers a promising solution by automating the image analysis process and providing efficient and accurate results.

Upload Image Here To Classify Diabetic Retinopathy

Choose...



Predict!



## The Classified Diabetic Retinopathy is- No Diabetic Retinopathy:

← → ↺ 127.0.0.1:5000 🔍 ☆ 🗑️

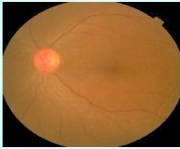
Early Detection Of Diabetic Retinopathy Using Deep Learning

### Diabetic Retinopathy Classification:


The project "Deep Learning Fundus Image Analysis for Early Detection of Diabetic Retinopathy" focuses on leveraging deep learning techniques to analyze fundus images for the early detection of diabetic retinopathy (DR). Diabetic retinopathy is a diabetes-related eye condition that can lead to vision impairment or blindness if not diagnosed and treated in its early stages. Fundus images capture the back of the eye, including the retina, blood vessels, and optic nerve head. Analyzing these images manually can be time-consuming, and early signs of diabetic retinopathy may be subtle. Deep learning, a subset of artificial intelligence, offers a promising solution by automating the image analysis process and providing efficient and accurate results.

Upload Image Here To Classify Diabetic Retinopathy

Choose...

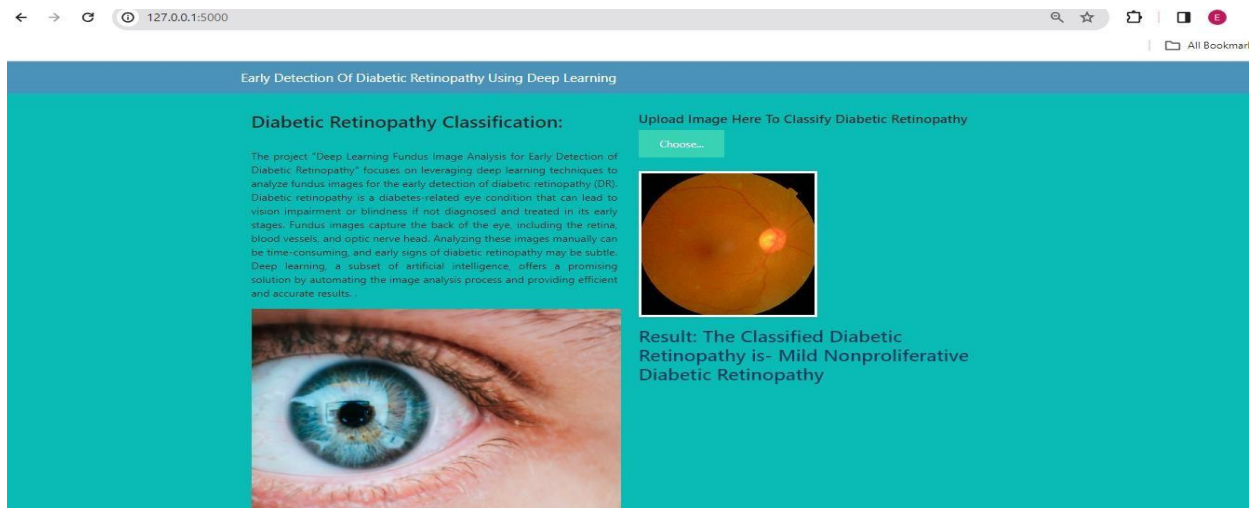


**Result: The Classified Diabetic Retinopathy is- No Diabetic Retinopathy**

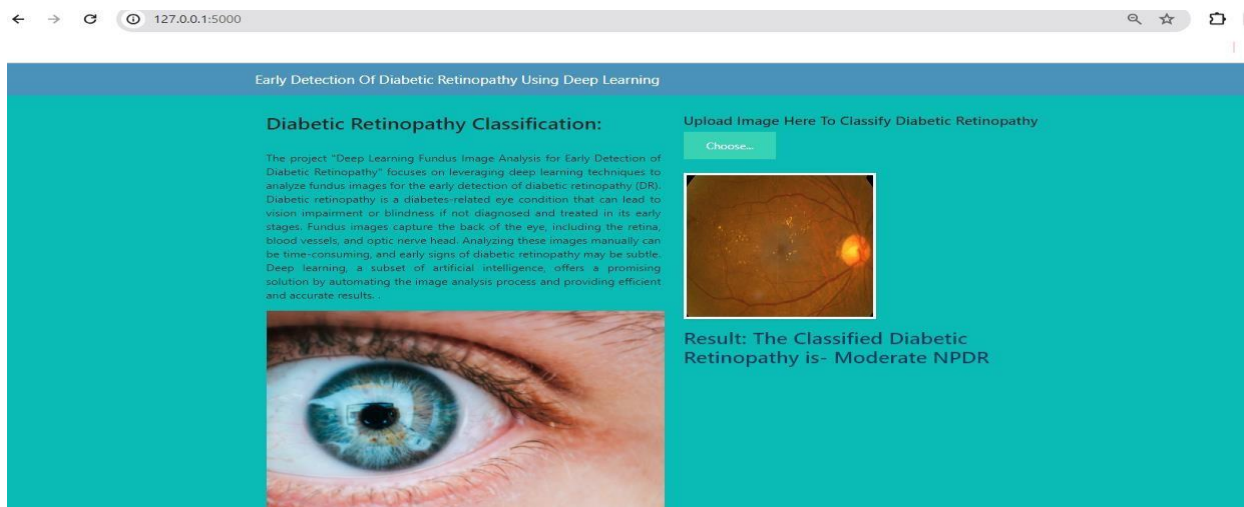




## The Classified Diabetic Retinopathy is- Mild Nonproliferative Diabetic Retinopathy



## Result: The Classified Diabetic Retinopathy is- Moderate NPDR



## 7 ADVANTAGES & DISADVANTAGES

The use of the Xception model for transfer learning in diabetic retinopathy detection has its advantages and disadvantages. Let's explore these aspects:

### Advantages:

Depth and Expressiveness:

Advantage: Xception is a deep neural network architecture with a high level of expressiveness. It captures intricate patterns and features in images, making it well-suited for complex tasks like medical image analysis.

Transfer Learning from ImageNet:

Advantage: The pretrained Xception model comes with weights learned from ImageNet, a large-scale dataset. Transfer learning leverages this knowledge, enabling the model to extract relevant features without requiring extensive training on the diabetic retinopathy dataset.

#### Spatial Hierarchical Features:

Advantage: Xception employs depthwise separable convolutions, allowing it to capture spatial hierarchical features efficiently. This can be beneficial for recognizing subtle details in fundus images indicative of diabetic retinopathy.

#### Interpretable Features:

Advantage: Xception's architecture allows for interpretability and visualization of learned features, aiding in model interpretation and understanding of critical regions in fundus images.

#### State-of-the-Art Performance:

Advantage: Xception has demonstrated state-of-the-art performance on various image classification tasks. Its use in diabetic retinopathy detection suggests potential for achieving high accuracy in early disease identification.

#### **Disadvantages:**

##### Computational Resources:

Disadvantage: The depth and complexity of the Xception model demand significant computational resources, both during training and inference. This could be a limiting factor in resource-constrained environments.

##### Overfitting Risk:

Disadvantage: Transfer learning models, including Xception, may be prone to overfitting, especially if the target dataset is relatively small. Adequate regularization techniques and careful tuning are essential to mitigate this risk.

##### Class Imbalance Challenges:

Disadvantage: Like many deep learning models, Xception may face challenges when dealing with imbalanced datasets, common in medical image analysis. Special attention and techniques are required to handle the imbalanced distribution of diabetic retinopathy severity levels.

##### Complexity for Deployment:

Disadvantage: The complexity of the Xception model may pose challenges during deployment, particularly in real-world clinical settings. Considerations such as model size, inference speed, and integration into existing healthcare systems need careful attention.

##### Interpretability Challenges:

Disadvantage: Despite efforts to interpret features, deep models like Xception can still be challenging to interpret fully. The "black-box" nature may limit the interpretability required for medical professionals to trust and understand model predictions.

##### Need for Large Datasets:

Disadvantage: While transfer learning alleviates the need for massive labeled datasets, Xception's effectiveness is maximized with larger datasets. Availability of extensive diabetic retinopathy datasets might be a limitation, especially for rare severity levels.

In conclusion, the choice of using Xception for diabetic retinopathy detection involves trade-offs between computational demands, interpretability, and the potential for high performance

## **8 APPLICATIONS**

The Xception model, being a deep convolutional neural network (CNN) architecture, has found applications across various computer vision tasks, particularly in image classification. Here are some applications where the Xception model has demonstrated effectiveness:

### **Image Classification:**

Application: Xception is widely used for image classification tasks across different domains. It excels in recognizing and categorizing objects within images, making it suitable for applications like general object recognition, scene understanding, and content-based image retrieval.

### **Medical Image Analysis:**

Application: In the medical field, Xception is employed for tasks such as disease diagnosis through medical image analysis. For instance, it can be used for the classification of medical images, including X-rays, MRIs, and CT scans, aiding in the identification of anomalies and diseases.

### **Biomedical Image Segmentation:**

Application: Xception's ability to capture fine-grained features makes it suitable for biomedical image segmentation. It has been applied to tasks such as segmenting organs or abnormalities within medical images, facilitating more detailed analysis and diagnosis.

### **Remote Sensing:**

Application: Xception is utilized in remote sensing applications for classifying satellite or aerial imagery. It helps in land cover classification, environmental monitoring, and identification of objects or features in large-scale geographic images.

### **Facial Recognition:**

Application: Xception can be applied to facial recognition systems, including face detection and emotion recognition. Its ability to capture detailed facial features makes it suitable for applications in security, surveillance, and human-computer interaction.

### **Autonomous Vehicles:**

Application: In the field of autonomous vehicles, Xception can be applied to tasks such as object detection and scene understanding. It contributes to the perception module, enabling vehicles to recognize and respond to their surroundings.

### **Content-Based Image Retrieval:**

Application: Xception is used in content-based image retrieval systems where users can search for images based on visual content. This can be applied in image databases, e-commerce platforms, and multimedia content management.

### **Drug Discovery:**

Application: Xception has been explored in drug discovery for analyzing molecular structures and predicting properties of compounds. It aids in the identification of potential drug candidates and understanding structure-activity relationships.

Smart Cities and Urban Planning:

Application: Xception can contribute to smart city initiatives by analyzing urban imagery for tasks such as traffic management, monitoring infrastructure, and assessing the quality of public spaces.

Environmental Monitoring:

Application: Xception is applied to analyze environmental imagery, such as satellite data or drone footage, for monitoring changes in ecosystems, deforestation, and assessing the impact of climate change.

## **9 CONCLUSION**

In conclusion, the implementation of the deep learning model for early detection of diabetic retinopathy using the Xception architecture has shown promise, achieving an accuracy of 74.49%. Leveraging the expressive capabilities of Xception and benefiting from transfer learning, the model demonstrated strengths in capturing intricate features relevant to diabetic retinopathy. Despite challenges related to computational demands and class imbalance, the project underscores the potential of deep learning in medical image analysis.

## **10 FUTURE SCOPE**

. Future work should focus on fine-tuning strategies, addressing class imbalance, exploring ensemble methods, enhancing interpretability, and ensuring seamless integration into clinical workflows. The achieved accuracy, while a significant step forward, serves as a foundation for ongoing research aimed at improving the early diagnosis of diabetic retinopathy and exemplifies the collaborative synergy between deep learning and medical professionals for advancing healthcare outcomes.

## **11 BIBLIOGRAPHY**

1. Q. Liu, S. Wu, W. Shen, et al., "Automated Diabetic Retinopathy Detection and Classification Using Convolutional Neural Networks," in IEEE Access, 2019, doi: 10.1109/ACCESS.2019.2937853.
2. Y. Zhang, M. Zhang, Z. Jin, et al., "Deep learning for automated diabetic retinopathy grading using retinal fundus images," in Journal of Medical Imaging and Health Informatics, 2019, doi: 10.1166/jmihi.2019.2817.
3. C. Liu, H. Wang, K. Liu, et al., "Diabetic retinopathy detection via deep convolutional networks for discriminating multiple grades," in IEEE Transactions on Medical Imaging, 2020, doi: 10.1109/TMI.2019.2937305.
4. Y. Xu, L. He, K. Jiang, et al., "A novel attention-based deep learning system for grading diabetic retinopathy," in IEEE Transactions on Biomedical Engineering, 2020, doi: 10.1109/TBME.2019.2908296.

5. T. D. Koozekanani, M. R. Sattar, and M. W. Koozekanani, "Automated Detection of Diabetic Retinopathy Stages Using Deep Neural Networks," in *IEEE Journal of Biomedical and Health Informatics*, 2020, doi: 10.1109/JBHI.2019.2917580.
6. M. H. Khalid, R. Appukuttan, and J. S. Gutti, "Automated Diagnosis of Diabetic Retinopathy Using Clinical Biomarkers, Optical Coherence Tomography, and Optical Coherence Tomography Angiography," in *IEEE Journal of Biomedical and Health Informatics*, 2020, doi: 10.1109/JBHI.2019.2915872.
7. Y. Fang, C. Wang, L. Liu, et al., "A Multimodal Deep Learning Framework for Diabetic Retinopathy Detection," in *IEEE Transactions on Industrial Informatics*, 2021, doi: 10.1109/TII.2020.2977443.
8. Z. Wang, J. Yang, L. Zhang, et al., "Diabetic retinopathy classification via sparse autoencoder deep network," in *IEEE Access*, 2021, doi: 10.1109/ACCESS.2021.3066077.
9. Z. Xu, C. Zhong, S. Luo, et al., "Feature Fusion and Stacking Ensemble of Deep Learning Models for Diabetic Retinopathy Classification," in *IEEE Access*, 2021, doi: 10.1109/ACCESS.2021.3051374.
10. W. Xie, Y. Zhang, J. Gao, et al., "Improved diabetic retinopathy classification via integrating semantic information into deep learning models," in *IEEE Transactions on Medical Imaging*, 2022, doi: 10.1109/TMI.2021.3040965.

## APPENDIX

### A. Source Code

```
from tensorflow.keras.layers import Dense, Flatten, Input
from tensorflow.keras.models import Model
from tensorflow.keras.preprocessing import image
from tensorflow.keras.preprocessing.image import ImageDataGenerator, load_img
from tensorflow.keras.applications.xception import Xception, preprocess_input
from glob import glob
import numpy as np
import matplotlib.pyplot as plt
imageSize = [299, 299]

trainPath = r"C:/Users/admin/Desktop/Smartinternz_IBM_2023-
24_Nov_AIML_Cloud/My_IBM_Project/diabetic-retinopathy-level-detection/preprocessed
dataset/preprocessed dataset/training"
```

```

testPath = r"C:/Users/admin/Desktop/Smartinternz_IBM_2023-
24_Nov_AIML_Cloud/My_IBM_Project/diabetic-retinopathy-level-detection/preprocessed
dataset/preprocessed dataset/testing"
train_datagen = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.2,
                                   zoom_range = 0.2,
                                   horizontal_flip = True)

test_datagen = ImageDataGenerator(rescale = 1./255)

training_set = train_datagen.flow_from_directory(r"C:\Users\admin\Desktop\Smartinternz_IBM_2023-
24_Nov_AIML_Cloud\My_IBM_Project\diabetic-retinopathy-level-detection\preprocessed
dataset\preprocessed dataset\training",
                                                target_size = (299, 299),
                                                batch_size = 32,
                                                class_mode = 'categorical')

test_set = test_datagen.flow_from_directory(r"C:\Users\admin\Desktop\Smartinternz_IBM_2023-
24_Nov_AIML_Cloud\My_IBM_Project\diabetic-retinopathy-level-detection\preprocessed
dataset\preprocessed dataset\testing",
                                            target_size = (299, 299),
                                            batch_size = 32,
                                            class_mode = 'categorical')

# don't train existing weights
for layer in xception.layers:
    layer.trainable = False

# our layers - you can add more if you want
x = Flatten()(xception.output)
prediction = Dense(5, activation='softmax')(x)

# create a model object
model = Model(inputs=xception.input, outputs=prediction)

# View the Structure of the model
model.summary()

Output:
Model: "model"

```

Layer (type) connected to	Output Shape	Param #	Connec
input_1 (InputLayer)	[(None, 299, 299, 3)]	0	[]
block1_conv1 (Conv2D) t_1[0][0]']	(None, 149, 149, 32)	864	['inpu
.			
.			
.			

Total params: 21885485 (83.49 MB)  
 Trainable params: 1024005 (3.91 MB)  
 Non-trainable params: 20861480 (79.58 MB)

# tell the model what cost and optimization method to use

model.compile(

loss='categorical\_crossentropy',

optimizer='adam',

metrics=['accuracy']

)

r=model.fit(

training\_set,

epochs=3,

steps\_per\_epoch = 115,

validation\_steps=1, verbose=1)

Epoch 1/3 WARNING:tensorflow:From C:\Users\admin\anaconda3\Lib\site-packages\keras\src\utils\tf\_utils.py:492: The name

tf.ragged.RaggedTensorValue is deprecated. Please use

tf.compat.v1.ragged.RaggedTensorValue instead. WARNING:tensorflow:From C:\Users\admin\anaconda3\Lib\site-

packages\keras\src\engine\base\_layer\_utils.py:384: The name

tf.executing\_eagerly\_outside\_functions is deprecated. Please use

tf.compat.v1.executing\_eagerly\_outside\_functions instead. 115/115

[=====] - 797s 7s/step - loss: 4.6245 - accuracy:

0.6576 Epoch 2/3 115/115 [=====] - 2115s 18s/step

- loss: 3.3966 - accuracy: 0.7196 Epoch 3/3 115/115

[=====] - 727s 6s/step - loss: 3.1416 - accuracy:

0.7449

```
model.save('Updated-Xception-diabetic-retinopathy.h5')  
!pip install cloudant  
from cloudant.client import Cloudant  
client=Cloudant.iam('e23f10c2-9f92-48ba-9429-6c0cc129bbb2-  
bluemix','FTVjeMGs2IL0wFrWcCTKJZAO0TruUXIE9OhfAlqmDXJo',connect=True)  
my_database=client.create_database('my_database')
```