**Software Requirements**

**Specification**

**for**

**CGAN-Driven Image Mapping**

**Prepared by:**

**Manali Jain 432034 22110360**

**Pushkar Patil 472040 22110200**

**Chetana Shinde 472057 22110601**

**Ayush Mundhe 473034 22110630**

**Vishwakarma Institute of Information Technology , Pune**

**Table of Contents**

[1. Introduction 3](#_Toc179983098)

[1.1 Purpose 3](#_Toc179983099)

[1.2 Document Conventions 3](#_Toc179983100)

[1.3 Intended Audience and Reading Suggestions 3](#_Toc179983101)

[1.4 Project Scope 4](#_Toc179983102)

[1.5 References 4](#_Toc179983103)

[2. Overall Description 5](#_Toc179983104)

[2.1 Product Perspective 5](#_Toc179983105)

[2.2 Product Features 5](#_Toc179983106)

[2.3 User Classes and Characteristics 6](#_Toc179983107)

[2.4 Operating Environment 7](#_Toc179983108)

[2.5 Design and Implementation Constraints 7](#_Toc179983109)

[2.6 User Documentation 7](#_Toc179983110)

[2.7 Assumptions and Dependencies 8](#_Toc179983111)

[3. System Features 8](#_Toc179983112)

[3.1 CGAN – Driven Image Mapping 8](#_Toc179983113)

[3.2 Features 9](#_Toc179983114)

[4. External Interface Requirements 10](#_Toc179983115)

[4.1 User Interfaces 10](#_Toc179983116)

[4.2 Hardware Interfaces 10](#_Toc179983117)

[4.3 Software Interfaces 10](#_Toc179983118)

[Appendix A: Analysis Models 11](#_Toc179983119)

# 1. Introduction

## 1.1 Purpose

The purpose of this SRS document is to define the functional and non-functional requirements for developing an image translation system using Conditional Generative Adversarial Networks (cGANs). The system focuses on translating images from one domain to another using a pix2pix model. The aim is to enhance the image quality and enable various translation tasks such as image colorization, style transfer, and image inpainting.

## 1.2 Document Conventions

This document is structured and formatted using the Times New Roman font in sizes 12 and 14, depending on the type of text. Diagrams such as Use Case, Class, Sequence, and Activity Diagrams are presented using UML (Unified Modeling Language) standards. All functional requirements are described with a numbered format for clarity. The system itself is developed using Python as the core programming language, utilizing machine learning frameworks such as TensorFlow or PyTorch for implementing the cGAN model.

## 1.3 Intended Audience and Reading Suggestions

This document is intended for a diverse audience involved in the development, deployment, and use of image translation systems based on cGANs. The audience may include but is not limited to:

* Researchers and Academics in the fields of machine learning, computer vision, and image processing.
* Developers working on image translation systems who will use this document to understand system requirements and specifications.
* Data Scientists involved in training and fine-tuning the cGAN models using various datasets.
* Healthcare Professionals or Business Analysts interested in exploring the system’s potential applications for medical image translation.
* IT Professionals responsible for system maintenance, integration, and scalability.

Readers are advised to have a fundamental understanding of neural networks, particularly Generative Adversarial Networks (GANs), as well as basic knowledge of machine learning concepts. While not necessary, familiarity with image processing tasks will help readers understand the system’s potential impact and utility in fields such as medical imaging, satellite imagery, and architectural design.

## 1.4 Project Scope

The project aims to develop an image translation system using a cGAN architecture that can efficiently perform high-quality image-to-image translations across different domains. The system focuses on:

* Translating images from one domain to another (e.g., transforming architectural facades into labeled maps, or converting satellite images into road maps).
* Enhancing image properties for tasks like colorization, style transfer, image inpainting, and object segmentation.
* Improving the training pipeline and performance through hyperparameter tuning, loss functions (such as GAN loss and L1 loss), and architecture optimizations like U-Net skip connections.

This cGAN-based system can be applied across various industries, including healthcare, urban planning, and satellite imagery. The flexibility of the system will allow developers to integrate new datasets and improve the performance for specific image translation tasks. The final system will consist of the following key components:

* Generator and Discriminator models for adversarial training.
* User Interface for users to upload images and view the results of image translation.
* Model Evaluation Metrics, including accuracy, generator/discriminator loss, and visual quality assessments.

By leveraging the pix2pix model and further enhancing it, this project aims to provide a robust solution to a wide range of image-to-image translation tasks, showcasing the power and versatility of Conditional Generative Adversarial Networks (cGANs).

## 1.5 References

1) Christopher hesse. https://affinelayer.com/ pixsrv/.

2) L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In ICLR, 2015.

3) E. Denton, S. Chintala, A. Szlam, and R. Fergus. Deep generative image models using a laplacian pyramid of adversarial networks. In NIPS, 2015

4) D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In ICCV, 2015.

5) L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. CVPR, 2016.

6) I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. WardeFarley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In NIPS, 2014.

7) L. Karacan, Z. Akata, A. Erdem, and E. Erdem. Learning to generate images of outdoor scenes from attributes and semantic layouts. arXiv preprint arXiv:1612.00215, 2016.

8) C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi. Photo-realistic single image super-resolution using a generative adversarial network.

9) X. Wang and A. Gupta. Generative image modeling using style and structure adversarial networks. In ECCV, 2016. In CVPR, 2017.

10) M. Mathieu, C. Couprie, and Y. LeCun. Deep multi-scale video prediction beyond mean square error. ICLR, 2016.

# 2. Overall Description

## 2.1 Product Perspective

The image translation system using Conditional Generative Adversarial Networks (cGANs) provides a robust framework for translating images from one domain to another. It is designed to handle various image-to-image translation tasks, including image colorization, style transfer, object segmentation, and super-resolution. This system focuses on high-quality image generation using a pix2pix model, where the generator network creates new images based on input images, and the discriminator distinguishes between real and generated images.

The system operates within a machine learning pipeline, where users can upload their image datasets, train the model on specific domains, and generate translated images that maintain key structural elements of the input. The system provides visualization and evaluation tools to assess the performance of the model, allowing users to fine-tune the model's parameters for improved results. Furthermore, this system integrates with image repositories and supports a range of datasets for training and testing the cGAN model, ensuring versatility in handling various image domains.

## 2.2 Product Features

The features of the product are designed to support the image translation tasks while ensuring an intuitive user experience. The following key features are provided:

1. Image Upload and Preprocessing:  
   Users can upload images in multiple formats (PNG, JPG) to the system, which will preprocess them (resize, normalize) before feeding them into the cGAN model.
2. Model Training:  
   The system allows users to train the generator and discriminator networks using paired datasets from different image domains. This includes adjusting hyperparameters like learning rate, loss function, and the number of training epochs.
3. Image Translation:  
   Once trained, the system enables users to input new images from a specific domain and generate the corresponding translated images in the target domain. Users can view real-time results of the generated images.
4. Evaluation Metrics:  
   The system provides feedback on the performance of the model using metrics such as generator loss, discriminator loss, and L1 loss. It also includes accuracy measurements and visual quality assessments through comparisons of generated images with real images.
5. Visualization Tools:  
   The system offers a graphical interface to display loss vs. epoch graphs, confusion matrices, and other statistical representations of the model’s performance over time.
6. Model Fine-Tuning:  
   Users can re-train the model on specific subsets of the dataset or adjust hyperparameters to improve the performance and accuracy of image translation results.

## 2.3 User Classes and Characteristics

The system is designed to be user-friendly, providing an intuitive interface for users from different backgrounds. The following are the primary user classes:

1. Researchers and Academics:  
   Users who are conducting research in the fields of machine learning, computer vision, or image processing. They may use the system to develop new models or analyze results for academic purposes.
2. Developers:  
   Software engineers or machine learning practitioners responsible for integrating cGAN models into larger systems or projects.
3. Data Scientists:  
   Professionals who work on dataset preparation, model training, and hyperparameter tuning to optimize the cGAN model for different domains.
4. Healthcare Professionals:  
   Users from the healthcare sector may use the system for tasks like medical image processing, e.g., translating MRI images into a different format or resolution.
5. General Users:  
   Users with basic computer skills who may experiment with the system to translate images from one domain to another, such as creating artistic styles or editing photos.

## 2.4 Operating Environment

The system operates on various hardware and software environments to ensure broad compatibility and performance efficiency. The following are the system’s operating environments:

1. Operating Systems:
   * Windows 10 or higher
   * macOS
   * Linux distributions (Ubuntu, CentOS)
2. Development Framework:
   * Python for implementing the cGAN model using TensorFlow or PyTorch
3. External Libraries:
   * Machine Learning Libraries: TensorFlow, PyTorch
   * Image Processing Libraries: OpenCV, NumPy
   * Visualization Libraries: Matplotlib, Seaborn

## 2.5 Design and Implementation Constraints

Design and implementation constraints arise due to technological limitations, system dependencies, and the complexity of training deep learning models. The key constraints are:

* High Computational Requirements:  
  Training cGAN models, particularly on large datasets, requires significant computational resources (GPUs). Users may experience delays during the training phase unless the system is deployed on high-performance hardware.
* Dataset Dependency:  
  The system relies on paired datasets for training. If the quality or size of the dataset is insufficient, it may affect the accuracy of the image translations.
* Model Performance:  
  The pix2pix model may experience challenges with mode collapse or slow convergence, leading to unreliable image translations. To address this, advanced GAN techniques or hyperparameter tuning will be necessary.
* Privacy and Security:  
  For healthcare-related applications (e.g., medical image translation), the system must adhere to data privacy regulations such as HIPAA, ensuring that all patient data is encrypted and stored securely.

## 2.6 User Documentation

The system will be accompanied by comprehensive documentation, covering the following aspects:

* Installation Guide:  
  A step-by-step guide on how to set up the system, including system requirements, software dependencies, and environment setup instructions.
* User Manual:  
  This manual will provide detailed instructions on how to use the system, including sections on uploading images, training the model, and visualizing results. It will include screenshots and diagrams to enhance user understanding.
* Troubleshooting Guide:  
  A guide for resolving common issues, such as performance bottlenecks during training or installation problems.
* API Documentation:  
  Documentation for developers who want to integrate the cGAN model into other applications or access the system programmatically.

## 2.7 Assumptions and Dependencies

The successful development and operation of the system depend on the following assumptions and dependencies:

* Dataset Availability:  
  It is assumed that users will have access to large, high-quality paired datasets to train the cGAN model effectively. Public datasets such as Cityscapes or CelebA may be used for testing and training.
* High-Speed Internet Connection:  
  The system assumes that the user has access to a stable internet connection to download datasets, train models on cloud-based resources (if needed), and communicate with external APIs.
* Hardware Resources:  
  It is assumed that the user has access to a high-performance computer with GPUs to facilitate fast training and inference. For large-scale image translation tasks, cloud services such as AWS or Google Cloud may be required.
* Software Dependencies:  
  The system relies on Python-based machine learning frameworks (e.g., TensorFlow, PyTorch), image processing libraries (e.g., OpenCV), and a web-based frontend (ReactJS). All dependencies must be correctly installed for the system to function.

# 3. System Features

The system features for the Conditional Generative Adversarial Network (cGAN) Image Translation System focus on delivering high-quality image-to-image translations, enabling model training and evaluation, and providing users with intuitive interaction for image processing. These features ensure that the system meets its goal of transforming images across different domains efficiently and effectively.

## 3.1 CGAN – Driven Image Mapping

**3.1.1 Name :** Image Translation System using CGAN.

**3.1.2 Goal :** The goal of this system is to translate images from one domain to another using the cGAN framework, specifically the pix2pix model. The system is designed to perform tasks such as image colorization, style transfer, object segmentation, and super-resolution. The generator creates translated images, while the discriminator evaluates their quality by distinguishing between real and generated images.

**3.1.3 Input :** The input to the system includes images from a specific domain, uploaded by the user. These images must be in a standard format such as PNG or JPG, and the system will preprocess them to prepare them for translation.

**3.1.4 Output :** The output consists of translated images in the target domain, generated by the cGAN model. These images will maintain the structural elements of the input while applying the target domain's properties (e.g., colorized images, maps derived from aerial photos).

### 3.2 Features

The key features of the system that enable efficient image translation and interaction are:

1. Image Upload and Preprocessing:

Users can upload images from a source domain (e.g., architectural facades, satellite images), and the system will automatically preprocess the images (resize, normalize, etc.) to prepare them for the translation task.

1. Model Training:

The system allows users to train the cGAN model on paired datasets. Users can adjust hyperparameters like learning rate, batch size, and the number of training epochs to fine-tune the model for optimal performance.

1. Image Translation:

After training the model, users can input new images, and the system will generate their translations in the target domain. For instance, inputting an architectural facade can generate a labeled map, or inputting a grayscale image can produce a colorized version.

1. Evaluation and Visualization:

The system provides performance metrics such as generator loss, discriminator loss, and L1 loss, allowing users to evaluate the accuracy and quality of the translated images. Visual tools like confusion matrices and loss vs. epoch graphs are available to help users understand the model’s performance.

1. Image Output Visualization:

Users can view the original input image, the target image (if available), and the translated image side-by-side. This feature helps in visually assessing the quality of the translation.

1. User Registration and Profile Management:

Users can register to create an account and store their datasets, trained models, and image translation results. This feature ensures that users can manage their work effectively and revisit their projects.

# 4. External Interface Requirements

The system relies on several external interfaces for seamless integration with hardware, software, and communication platforms. These external interfaces ensure that the cGAN image translation system can interact effectively with various components and platforms.

## 4.1 User Interfaces

The system should have a user-friendly web interface accessible via standard web browsers, providing an intuitive user experience.

## 4.2 Hardware Interfaces

1. Operating System: Windows 10 or higher, macOS, or Linux distributions.
2. Hard Disk: At least 1 TB (1024 GB) storage to accommodate large datasets and model checkpoints.
3. RAM: 16 GB or higher to handle training tasks and model inference.
4. GPU: Required for accelerated model training and inference (e.g., NVIDIA GPUs).
5. Keyboard and Mouse: Standard input devices for interacting with the web interface.
6. Internet Connection: Required for downloading datasets, communicating with cloud resources, and accessing the web interface.

## 4.3 Software Interfaces

1. Frontend:

The system’s user interface should be developed for a responsive and interactive experience.

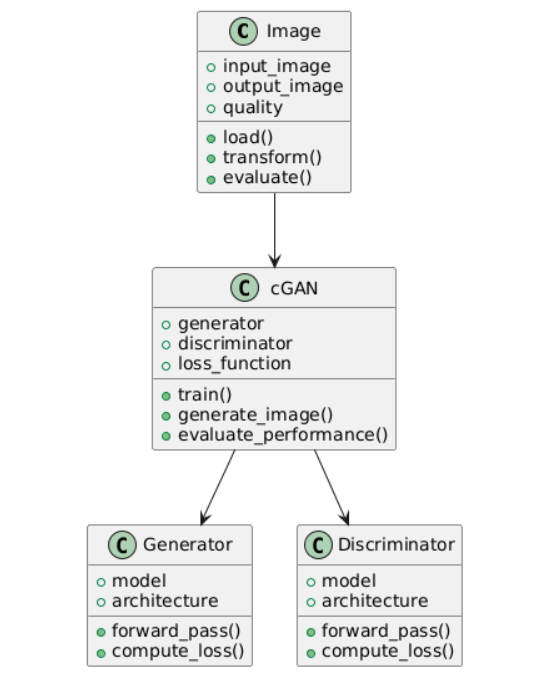
1. Backend:

The system utilizes Python for implementing the cGAN model with machine learning frameworks such as TensorFlow or PyTorch.

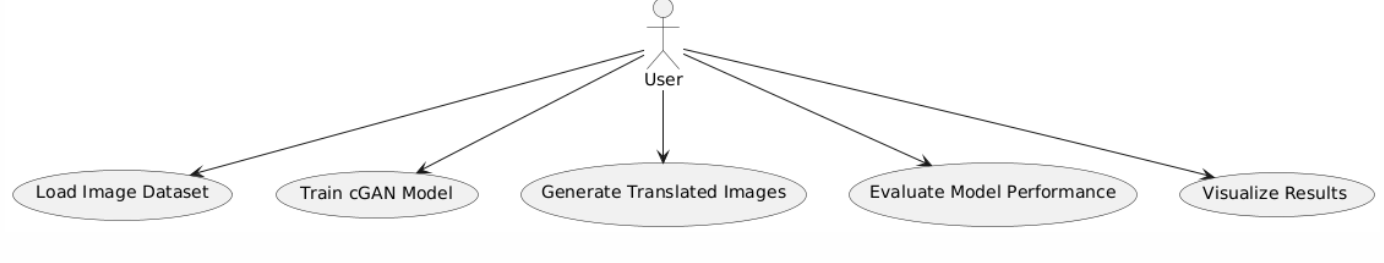
1. MongoDB or PostgreSQL is used for managing user accounts, datasets, and model checkpoints.
2. Application Framework:

The system runs on a Python-based application framework like Flask or Django for managing backend services and API communications.

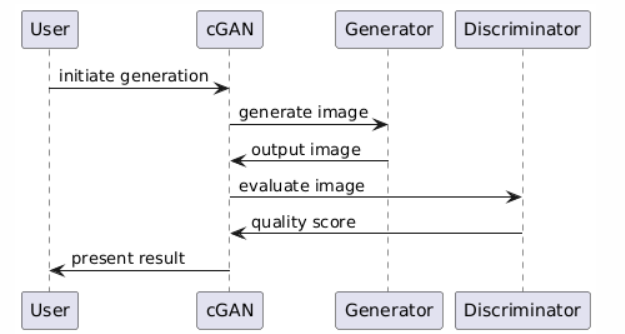
# Appendix A: Analysis Models



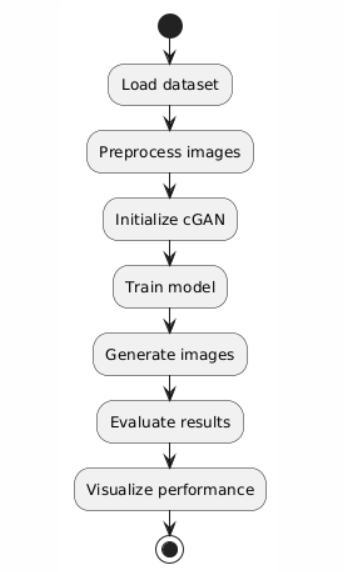
**Figure 1 : Class Diagram**



**Figure 2: Use Case Diagram**



**Figure 3: Sequence Diagram**



**Figure 4: Activity Diagram**