# Software Requirements Specification

**for**

# <PixelForge>

**Version <1.0>**

**Prepared by**


**Group Name:** *<place your group name here>*

| | | |
|---|---|---|
| **Harshit Mehta** | **CE094** | **24ceuos905@gmail.com** |
| **Dhruv Jani** | **CE096** | **23ceuoz057@gmail.com** |


**Instructor:** **Professor Brijesh S Bhatt**

**Course:** **System Design Practices**

# Contents

# Revisions

| Version | Primary Author(s) | Description of Version | Date Completed |
|---|---|---|---|
| Draft Type and Number | Full Name | Information about the revision. This table does not need to be filled in whenever a document is touched, only when the version is being upgraded. | 00/00/00 |

# 1 Introduction

PixelForge is a locally deployed AI image generation system that produces images from textual prompts using diffusion-based deep learning models. The system is designed to operate entirely offline, executing all model inference and processing on the user's local machine without reliance on external APIs or cloud-based services.

This section introduces the purpose, scope, audience, terminology, conventions, and references related to this Software Requirements Specification (SRS). It provides the foundation required to understand the functional and non-functional requirements defined in later sections of this document.

## 1.1 Document Purpose

The purpose of this Software Requirements Specification (SRS) document is to define the functional and non-functional requirements for **PixelForge – Version 1.0**, a local AI-based image generation system. This document serves as a formal agreement between stakeholders regarding system behavior, constraints, and performance expectations.

This SRS covers the complete PixelForge system, including the Python-based backend responsible for image generation using Stable Diffusion, the LoRA fine-tuning mechanism, dataset preparation and captioning workflow, and the frontend user interface built using Next.js or React. The document is intended to guide system design, development, testing, evaluation, and future enhancements.

## 1.2 Product Scope

PixelForge is a standalone software application that enables users to generate high-quality images from textual descriptions using Stable Diffusion 1.5 executed locally. The system aims to provide a privacy-preserving, customizable, and efficient alternative to cloud-based image generation tools.

The primary goals of PixelForge are:

- To eliminate dependency on external APIs and internet connectivity
- To allow domain-specific fine-tuning using LoRA and curated datasets
- To provide users with control over image generation parameters
- To demonstrate practical application of modern generative AI techniques

The system is intended for educational use, research, experimentation, and portfolio development.

## 1.3 Intended Audience and Document Overview

This document is intended for the following audiences:

- **Course Instructor / Professor** – for evaluation and grading
- **Students / Developers** – for implementation guidance
- **Testers** – for validating system behavior against requirements

Document Organization:

- **Section 1** introduces the system and document structure
- **Section 2** provides an overall description of the system
- **Section 3** details specific functional requirements and use cases
- **Section 4** defines non-functional requirements
- **Section 5** lists additional requirements
- **Appendices** provide supporting data and documentation

Readers are encouraged to begin with Sections 1 and 2 for context before reviewing detailed requirements in Sections 3 and 4.

## 1.4 Definitions, Acronyms and Abbreviations

| ***Term*** | ***Description*** |
|---|---|
| *AI* | *Artificial Intelligence* |
| *API* | *Application Programming Interface* |
| *BLIP* | *Bootstrapping Language-Image Pretraining* |
| *CFG* | *Classifier-Free Guidance* |
| *CPU* | *Central Processing Unit* |
| *DFD* | *Data Flow Diagram* |
| *GPU* | *Graphics Processing Unit* |
| *LoRA* | *Low-Rank Adaptation* |
| *ML* | *Machine Learning* |
| *SRS* | *Software Requirements Specification* |
| *UI* | *User Interface* |
| *VAE* | *Variational Autoencoder* |

## 1.5 Document Conventions

This document follows IEEE SRS formatting guidelines.
The following conventions are used:

- **Font:** Arial, size 11 or 12
- **Text:** Single-spaced with 1-inch margins
- **Headings:** Bold, numbered according to template
- *Italics:* Used for comments or emphasis where required
- Requirements are labeled using clear, declarative statements

## 1.6 References and Acknowledgments

- ***References:***
- *IEEE 830-1998 Software Requirements Specification Standard*
- *Hugging Face Diffusers Documentation*
- *PyTorch Documentation*
- *Open Images V7 Dataset Documentation*
- *BLIP Image Captioning Research Paper*
- ***Acknowledgments:***

*The authors acknowledge open-source contributors to Stable Diffusion, Hugging Face, and PyTorch, as well as academic guidance provided by course instructors.*

# 2 Overall Description

## 2.1 Product Overview

PixelForge is a new, self-contained software system designed to generate images from textual prompts using locally executed diffusion-based machine learning models. The system is not a replacement or extension of any existing institutional software; rather, it is an independent application developed for educational and experimental purposes.

The product consists of three major components:

1. A **frontend user interface** built using Next.js or React that allows users to enter prompts, configure parameters, and view generated images.
2. A **Python-based backend** that manages model loading, image generation, and response handling.
3. A **local machine learning engine** based on Stable Diffusion 1.5, enhanced using LoRA fine-tuning on selected datasets.

All components execute on the user's local machine. The frontend communicates with the backend through local HTTP requests, and the backend interacts directly with the machine learning model and local storage. No external cloud services or APIs are involved in the operation of the system.

At a high level, the system operates as follows: the user provides a text prompt through the frontend interface, the backend processes the request using the diffusion model and optional LoRA weights, and the generated image is returned to the frontend for display and download.

## 2.2 Product Functionality

At a high level, PixelForge provides the following major functionalities:

- Accept textual prompts from the user to generate images.
- Allow users to configure image generation parameters such as resolution, inference steps, CFG scale, and seed value.
- Generate images locally using Stable Diffusion 1.5 without internet connectivity.
- Support domain-specific enhancement of image generation through LoRA fine-tuned models.
- Automatically caption dataset images using BLIP for training purposes.
- Display generated images in the user interface.
- Allow users to download generated images in standard image formats.

- Maintain a local history of generated images and associated metadata.

Detailed functional behavior is specified in **Section 3: Specific Requirements**.

## 2.3 Design and Implementation Constraints

The design and implementation of PixelForge are subject to the following constraints:

- **Local Execution Constraint:** All image generation and processing must occur on the local machine without reliance on cloud-based APIs or external inference services.
- **Hardware Constraint:** GPU acceleration is preferred for performance; however, the system must support CPU-based execution as a fallback.
- **Model Constraint:** Stable Diffusion 1.5 is used as the base generative model and its core weights must remain frozen during fine-tuning.
- **Fine-Tuning Constraint:** Domain adaptation must be performed using LoRA to reduce computational cost and prevent full model retraining.
- **Dataset Constraint:** Fine-tuning data is limited to selected classes from the Open Images V7 dataset.
- **Design Methodology Constraint:** The system design follows object-oriented principles and is documented using UML diagrams.
- **Programming Languages:** Python is used for backend and machine learning components, while JavaScript/TypeScript is used for frontend development.

## 2.4 Assumptions and Dependencies

The following assumptions and dependencies apply to PixelForge:

- **Assumptions:**
  - The user has access to a machine capable of running Python and the required machine learning libraries.
  - An NVIDIA GPU with compatible drivers is available for optimal performance.
  - Users possess basic familiarity with web-based interfaces.
- **Dependencies:**
  - The system depends on third-party open-source libraries such as PyTorch, Hugging Face Diffusers, Transformers, and FastAPI.

- The quality of fine-tuning depends on the availability and correctness of Open Images V7 data.
- Caption generation accuracy depends on the performance of the BLIP model.

If any of these assumptions change or dependencies fail, system performance or functionality may be affected.

# 3 Specific Requirements

## 3.1 External Interface Requirements

### 3.1.1 User Interfaces

PixelForge provides a web-based graphical user interface developed using Next.js or React. The interface allows users to interact with the system through standard web browser controls such as text inputs, sliders, dropdown menus, and buttons.

The main user interface components include:

- A text input field for entering image generation prompts
- An optional negative prompt input field
- Parameter controls for CFG scale, inference steps, image resolution, and seed value
- A dropdown menu for selecting LoRA fine-tuned models
- A “Generate Image” button to initiate image generation
- An image preview panel to display generated images
- A download button to save generated images locally
- A history section to view previously generated images

User interaction is performed using mouse and keyboard input. The interface is designed to be intuitive and responsive for desktop environments.

### 3.1.2 Hardware Interfaces

PixelForge interfaces with the following hardware components:

- **CPU:** Used for general computation and fallback image generation when GPU is unavailable.
- **GPU (Optional):** NVIDIA GPU used to accelerate diffusion inference and LoRA fine-tuning.
- **Local Storage:** Used to store generated images, prompt history, and LoRA weight files.

The system does not directly communicate with external hardware sensors or embedded devices.

### 3.1.3 Software Interfaces

PixelForge interacts with the following software components:

- **Operating System:** Windows or Linux-based systems.
- **Web Browser:** Chrome, Firefox, or Edge for frontend access.
- **Python Runtime:** Executes backend services and machine learning pipelines.
- **Third-party Libraries:** PyTorch, Diffusers, Transformers, FastAPI.

No external cloud services or third-party APIs are required.

## 3.2 Functional Requirements

### 3.2.1 Text-to-Image Generation

The system shall generate an image locally based on a textual prompt provided by the user using Stable Diffusion 1.5.

### 3.2.2 Parameter Configuration

The system shall allow users to configure image generation parameters including inference steps, CFG scale, image width, image height, and seed value.

### 3.2.3 Local Model Execution

The system shall execute all image generation processes locally without reliance on internet connectivity or cloud-based APIs.

### 3.2.4 LoRA Model Integration

The system shall allow loading and switching of LoRA fine-tuned models during image generation.

### 3.2.5 Image Preview and Download

The system shall display generated images in the user interface and allow users to download them in standard formats.

### 3.2.6 Prompt and Image History

The system shall store generated images along with associated prompts and parameters for later reference.

### 3.2.7 3.2.7 F7: Error Handling

The system shall notify users when invalid inputs or internal errors occur.

# 4 Other Non-functional Requirements

## 4.1 Performance Requirements

- Image generation time should be under **10 seconds** on a GPU-enabled system for a 512×512 image.
- The system shall remain responsive during image generation.
- LoRA training time shall be optimized using parameter-efficient fine-tuning.

## 4.2 Safety and Security Requirements

- The system shall not transmit user data externally.
- Generated images and prompts shall be stored locally.
- The system shall not perform any medical diagnosis or sensitive decision-making.
- User-generated content remains under user control.

## 4.3 Software Quality Attributes

### 4.3.1 Usability

The system shall provide an intuitive interface requiring minimal training.

### 4.3.2 Maintainability

The system shall use modular architecture allowing components such as models and UI to be updated independently.

### 4.3.3 Portability

PixelForge shall operate on multiple operating systems with minimal configuration changes.