PROJECT REPORT

on

Depth Spark

from

Digipodium

Towards partial fulfillment of the requirements

for the award of degree of

Bachelor of Computer Applications

(DS & AI)

from

Babu Banarasi Das University

Lucknow

Academic Session 2024 – 25

School of Computer Applications

I Floor, CA- BLOCK, BBD University, BBD City, Ayodhya Road, Lucknow (U. P.) INDIA 226028
PHONE: HEAD: 0522-6196242 **Dept. T&P Cell:** 9990462250

w w w . b b d u . a c . i n

PROJECT REPORT

on

Depth Spark

from

Digipodium

Towards partial fulfillment of the requirements for the award of degree of

Bachelor of Computer Applications (DS & AI)

From

Babu Banarasi Das University Lucknow

Developed and Submitted by
Uttam Tripathi
1220258280

Under Guidance of
Mr. Durga Shankar Shukla
Assistant Professor

Academic Session 2024 - 25

School of Computer Applications

UNDER TAKING

This is to certify that Project Report entitled

Depth Spark

being submitted by

Uttam Tripathi

Towards the partial fulfillment of the requirements
for the award of the degree of

**Bachelor of Computer Applications
(DS & AI)**

to

**Babu Banarasi Das University
Lucknow**

Academic Year 2024-25
is a record of the student's own work carried out at

Digipodium

and to the best of our knowledge the work reported herein does not form a part of any other thesis or work on the basis of which degree or award was conferred on an earlier occasion to this or any other candidate.

Authorized Signatory	Students Signature
Digipodium	Name: Uttam Tripathi
	Roll No: 1220258280

Registration no: ***Dp/2025/6768***
Certificate Id: ***23011b16-dcba-5d6c-adb0-6bc0014ed9d1***

digi**podium**

Project completion certificate

This is to certify that **Uttam Tripathi** has successfully concluded Internship with exceptional dedication and proficiency at Digipodium. The Internship details are as follows:

Project Title : **Depth Spark**

Technology : **Python Data Science**

Duration : **FEB 2025 - MAY 2025**

Uttam Tripathi exhibited commendable skills and commitment in contributing to the Depth Spark project, showcasing expertise in **Python Data Science**.

For any inquiries or additional information regarding Uttam's contributions, please feel free to contact the undersigned.

[illegible]

Director
Digipodium
9415082377

Digipodium, Raja ram Kumar Plaza,
Hazratganj, Lucknow-226001

DECLARATION

I, hereby declare that the report entitled **"Depth Spark"**, submitted to the **School of Computer Applications**, **Babu Banarasi Das University**, *BBD City, Ayodhya Road, Lucknow, Uttar Pradesh – 226028*, is submitted in partial fulfilment of the requirements for the award of the degree of **Bachelor of Computer Applications** in specialization with *Data Science & Artificial Intelligence.*

This report is the outcome of my own effort and has been completed under the guidance and supervision of **Mr. Durga Shankar Shukla**. I also declare that the contents of this report have not been submitted, either in full or in part, to any other university or institution for the award of any degree or diploma.

Place: Lucknow
Date: 19 May, 2025

Signature of the Student
Students' Name: Uttam Tripathi
University Roll No.: 1220258280

Acknowledgement

It gives us immense pleasure to express our sincere gratitude to all those who extended their support and guidance throughout the course of this project. The successful completion of this work would not have been possible without the encouragement and assistance of many individuals.

First and foremost, we would like to extend our profound gratitude to **Dr. Reena Srivastava**, Honourable *Dean*, School of Computer Applications, Babu Banarasi Das University, for her continued support, motivation, and encouragement. Her visionary leadership and guidance have always been a source of inspiration for us. We are also deeply thankful to **Dr. Prabhash Chandra Pathak**, *Head*, School of Computer Applications, for his constant support and valuable insights, which played a vital role in shaping our project and helping us stay focused on our objectives.

We extend our deep appreciation to our project guide, **Mr. Durga Shankar Shukla**, for his unwavering support, expert guidance, and timely feedback throughout the duration of the project. His willingness to assist us at every step and address our queries with patience greatly contributed to the successful completion of this work. We would also like to express our gratitude to the **Training and Placement Coordinators** for their support, cooperation, and guidance during the course of this project.

Last but not the least, we sincerely thank our **families and friends** for their constant support, understanding, and motivation. Their encouragement has been a strong pillar behind our consistent efforts.

Table of contents

1. Introduction

In the rapidly advancing digital world, visual content plays a critical role in capturing user attention and enhancing communication. Industries such as e-commerce, marketing, education, and entertainment increasingly rely on rich, interactive visuals to drive engagement, improve user experience, and deliver complex information effectively. While 2D images have long been a standard, the demand for immersive 3D experiences is steadily growing.

However, creating 3D models typically requires specialized technical expertise, expensive software tools, and significant time investment. For many small businesses, educators, students, and independent content creators, these challenges create a barrier to adopting 3D technology. Traditional 3D content development is often inaccessible to those without technical backgrounds or the financial means to afford professional services.

Depth Spark was conceptualized to address this gap. It is a web-based platform that simplifies and democratizes 3D model creation by enabling users to easily convert 2D images into 3D models — without any specialized knowledge or software. By combining a simple, user-friendly interface with powerful AI capabilities, Depth Spark empowers users across different sectors to leverage the power of 3D visualization, making it free, fast, and accessible to all.

1.1 Project Overview

Depth Spark is an innovative, web-based application built using Python (Flask Framework) that allows users to generate 3D models from 2D images in a few simple steps. The system integrates the Falcon AI API, which applies advanced artificial intelligence techniques to accurately interpret depth, structure, and shape from a single image and generate a realistic 3D model.

Users can interact with the system through a clean and responsive web interface, uploading images either via file selection or by providing a URL link. The system then processes the image in the background, with real-time progress tracking and user-friendly notifications. Once the model is generated, users can view it interactively — rotating, zooming, and moving the model — and download the output in GLB format, a compact, efficient file type widely supported across modern platforms.

Additional features such as secure authentication (login/signup), error handling, and cross-platform accessibility ensure that Depth Spark provides a seamless and secure user experience. With full support for common image formats like JPG, JPEG, PNG, and GIF, the application broadens its usability across different devices and user needs.

1.2 Problem Statement & Objectives

Problem Statement: Despite the widespread benefits and increasing demand for 3D content, creating high-quality 3D models remains a technical challenge. Conventional 3D modeling processes are costly, time-consuming, and require professional-grade software like Blender, Autodesk Maya, or Unity, which are not accessible to everyone. Moreover, individuals and businesses lacking technical expertise often find it difficult to create even simple 3D models, thus missing opportunities for innovation and engagement.

Objectives:

- To develop a free-to-use, user-friendly web platform that automates 2D-to-3D model conversion.

- To integrate AI technologies capable of accurately generating realistic 3D structures from 2D input.
- To provide downloadable output in a widely supported, lightweight 3D format (GLB).
- To ensure that the application is device-independent and accessible through any modern web browser.
- To create an interactive viewer that allows users to inspect and manipulate the generated models before downloading.
- To implement user authentication for secure and personalized experiences.
- To contribute toward making 3D technology accessible to non-technical users, small businesses, educational institutions, and independent creators.

1.3 Project Scope & Motivation

Scope: The initial scope of Depth Spark includes the ability to:

- Accept and validate 2D images uploaded via file or URL.
- Integrate Falcon AI technology for automated 2D-to-3D model generation.
- Allow users to interact with 3D models using an embedded viewer.
- Enable users to securely sign up, log in, and manage their activities.
- Provide final outputs in the GLB file format suitable for web, AR/VR, and gaming applications.

In future iterations, Depth Spark aims to expand by supporting animated 3D models, batch processing, API-based service extensions, integration with augmented reality (AR) platforms, and 3D printing compatibility.

Motivation: The primary motivation behind Depth Spark stems from the vision to democratize access to 3D content creation. While large organizations can invest heavily in 3D design, small businesses, educators, and creative individuals often find themselves constrained by budget, time, and technical hurdles. With Depth Spark, we wanted to remove these barriers — offering a simple, free, and efficient tool that brings 3D capabilities within reach of everyone.

Moreover, by harnessing artificial intelligence and focusing on usability, we believe Depth Spark can contribute toward wider adoption of 3D technologies in everyday digital interactions — from online shopping experiences to virtual classrooms and beyond.

1.4 Target Audience

Depth Spark is designed to serve a wide range of users across industries by making 3D content creation more accessible, efficient, and affordable. By simplifying the 2D to 3D conversion process, it offers value to the following groups:

Small and Medium E-Commerce Businesses: These businesses often lack the resources for costly 3D photography. Depth Spark allows them to create interactive 3D product visuals quickly and affordably, enhancing online shopping experiences and boosting customer engagement.

Marketing and Advertising Agencies: Creative professionals can use Depth Spark to build eye-catching, interactive visuals for campaigns, social media, and digital ads. It saves time and cost while adding a modern edge to their content.

Educational Institutions: Teachers and students can turn static images into 3D models to make learning more interactive. It’s ideal for explaining complex concepts in subjects like biology, engineering, and geography.

Independent Content Creators: Designers, influencers, and hobbyists can use Depth Spark to enhance their visual content across platforms. It enables creativity without requiring advanced technical skills or expensive software.

Developers and AR/VR Enthusiasts: For those building immersive applications or games, Depth Spark provides fast, low-effort 3D model generation—perfect for prototyping and experimentation.

Startups and Innovators: Entrepreneurs can use Depth Spark to build MVPs and product demos with interactive 3D visuals, helping them pitch ideas and attract investors without large design budgets.

Built for Everyone: With a focus on usability and device compatibility, Depth Spark empowers users of all backgrounds—technical or non-technical—to explore and leverage 3D technology. It opens up new creative and professional possibilities while keeping costs and complexity low.

2. <u>Literature Survey</u>

2.1 Existing Tools, Technologies, and Analysis

The evolution of 2D-to-3D content creation has accelerated rapidly with the development of AI-driven tools. Several platforms now aim to simplify the generation of 3D models from 2D images. However, despite these advancements, gaps remain in accessibility, affordability, and ease of use — particularly for non-technical users.

Existing Tools and Platforms

Meshy AI: Meshy AI provides an AI-powered solution for 2D-to-3D conversion. It excels at generating mesh structures quickly from single images. However, Meshy AI often requires additional refinement post-conversion, and some features are locked behind premium access.

3D AI Studio: 3D AI Studio offers a suite of AI-based 3D model creation tools designed for developers and businesses. Although it produces high-quality outputs, it typically targets commercial users with paid plans, making it less accessible to students and independent creators.

Hyper3D: Hyper3D specializes in creating fast, high-fidelity 3D models from limited visual input. While it reduces the technical barrier compared to traditional methods, its platform still requires understanding of 3D rendering settings, which can intimidate beginners.

Blender: Blender remains a dominant open-source tool for manual 3D modeling, texturing, and animation. It offers powerful capabilities but demands a steep learning curve. For users looking for an automated 2D-to-3D pipeline without manual modeling, Blender may feel overwhelming.

Tripo: Tripo is a newer AI tool designed to generate 3D models from 2D inputs. Its focus is on speed and realism. However, being a relatively new entrant, it still lacks some advanced controls and fine-tuning capabilities found in more established systems.

Core Technologies in Use

AI-Based Depth Estimation: Platforms like Meshy AI, Hyper3D, and Tripo use deep learning algorithms to predict depth information from 2D images, which enables single-image 3D model generation.

Photogrammetry (Limited Use): Some platforms still utilize traditional photogrammetry techniques when multiple images are available, although AI-first methods are now preferred for simplicity.

GLTF/GLB File Formats: Most modern systems export 3D models in .glb or .gltf formats for efficient use across web, AR/VR, and mobile applications.

Gaps Identified:

- Many platforms offer limited free access and prioritize paid features.
- Technical barriers remain for users unfamiliar with 3D workflows.
- Processing time and model fidelity vary widely depending on the tool.
- No single platform provides a fully free, easy, and universally accessible solution for simple 2D-to-3D needs.

2.2 Relevance of 3D in Current Applications

The integration of 3D technologies across industries is transforming user experiences and operational workflows. Today, realistic 3D models enhance communication, engagement, and understanding in ways that traditional 2D content cannot match.

Key Areas of 3D Application

E-Commerce: Online retailers increasingly utilize 3D models to offer interactive product views, enhancing customer trust and reducing return rates. Leading brands have shown that 3D visualizations can significantly boost engagement and sales.

Education and Training: In fields like medicine, engineering, and natural sciences, 3D models help students visualize complex structures and processes. Virtual labs and 3D simulations create more immersive, effective learning experiences.

Marketing and Advertising: 3D advertisements and AR marketing campaigns are becoming standard. Immersive product experiences foster emotional engagement and higher retention rates among audiences.

Gaming and Entertainment: 3D modeling is central to modern gaming environments, movies, VR experiences, and even social media filters. Realistic and detailed assets contribute heavily to user immersion.

Architecture and Real Estate: 3D walkthroughs and virtual property tours allow buyers to explore buildings remotely, improving customer experience and reducing the need for physical site visits.

Emerging Trends: Augmented Reality (AR) and Virtual Reality (VR): AR and VR platforms heavily depend on accurate and lightweight 3D models for immersive environments.

Metaverse Development: As companies invest in building interconnected virtual worlds, the demand for mass-scale 3D model generation is soaring.

Personalization in Consumer Products: Custom 3D models for avatars, wearables, and personalized products require easy-to-use 3D creation tools.

3. Need Identification & Problem Definition

3.1 Gaps in 2D Systems

Although 2D imagery has been the foundation of digital communication for decades, it faces increasing limitations in today's highly interactive and immersive world. Static images, while easy to create and share, fall short in delivering rich, engaging, and realistic experiences demanded by modern users.

Key Gaps Identified

Lack of Depth and Realism: 2D images provide only a flat representation of objects. They cannot convey spatial relationships, texture, or depth accurately, limiting the viewer's understanding and interaction.

Limited User Engagement: Studies show that interactive 3D models dramatically increase user engagement compared to static images. 2D content struggles to capture attention in competitive digital spaces like e-commerce and social media.

Inability to Support Immersive Technologies: Emerging technologies like Augmented Reality (AR), Virtual Reality (VR), and Metaverse platforms require 3D assets for full functionality. 2D images cannot be integrated into these environments effectively without significant manual conversion efforts.

Poor Representation of Complex Objects: In fields like architecture, medicine, and manufacturing, 2D diagrams and photos often fail to convey the complexity of real-world objects. 3D representations offer a more intuitive and comprehensive view.

Barrier to Innovation: As industries evolve towards personalization, virtual try-ons, and interactive demos, relying on 2D systems limits innovation and hinders customer experiences.

3.2 Market Demand and Challenges

The demand for 3D content has exploded in recent years, driven by advancements in AR/VR, online shopping, digital marketing, education, and entertainment. However, significant challenges prevent widespread adoption, especially among smaller businesses, educators, and independent creators.

Current Market Trends

E-Commerce Evolution: Retailers are shifting toward 3D product displays, virtual showrooms, and AR try-ons to enhance customer experience. Platforms like Amazon and Shopify are encouraging 3D model uploads to stay competitive.

AR/VR Expansion: AR applications in mobile shopping, virtual real estate tours, educational tools, and even healthcare simulations are growing rapidly, each requiring high-quality 3D models.

Remote Learning and Training: Post-pandemic, education has embraced digital tools, and interactive 3D content is proven to improve engagement, comprehension, and retention.

Gaming, Animation, and Metaverse Development: As more companies enter the Metaverse and online gaming industries, the demand for high-volume, cost-effective 3D content generation is becoming urgent.

Major Challenges in Meeting Demand

High Cost of Professional 3D Creation: Traditional 3D modeling tools require expensive software licenses and expert skills, putting them out of reach for many smaller players.

Steep Learning Curves: Software like Blender, Maya, and ZBrush, though powerful, demand months or years of practice to master, discouraging casual or urgent 3D content creation.

Limited Automation in Existing Solutions: While some platforms offer AI assistance, most still require manual intervention for model cleaning, texture mapping, or optimization.

Inaccessibility for Non-Technical Users: Many entrepreneurs, educators, or marketers lack the technical background needed to work with complex 3D software.

Time-Consuming Processes: Creating high-fidelity 3D models from scratch remains a time-intensive task, slowing down project timelines and increasing costs.

Opportunity for Depth Spark

Given these conditions, there is an undeniable opportunity for a solution like Depth Spark that:

- Provides free access to high-quality 3D model generation.
- Uses AI to automate and simplify the process.
- Requires no prior 3D modeling skills.
- Outputs ready-to-use 3D files compatible with web and mobile applications.
- Is fast, reliable, and user-friendly.

By filling these critical gaps, Depth Spark is positioned to empower a wide range of users — from small startups and students to content creators and educators — thereby democratizing access to 3D technologies.

4. Feasibility Study

Feasibility studies are essential in any project to evaluate whether the proposed system can be developed successfully within given constraints and resources. For the Depth Spark project, we assess feasibility from three major perspectives: Technical, Operational, and Economic. This chapter provides an in-depth analysis of each dimension to ensure the project's viability.

4.1 Technical, Operational, and Economic Feasibility

A) Technical Feasibility
Technical feasibility focuses on determining whether the current technical resources are sufficient for the successful development of Depth Spark.

Key Considerations
Technology Stack Availability:

- Frontend: Development using HTML, CSS, and JavaScript ensures cross-browser compatibility.
- Backend: Flask (Python) is lightweight, robust, and highly suitable for building web applications.
- Database: SQLAlchemy is chosen for flexible, scalable database management.
- AI Integration: The project leverages Falcon AI API for advanced 2D-to-3D model generation.
- 3D Viewer: WebGL/GLTF frameworks are available for rendering 3D models interactively in browsers.

Ease of Development: Developers are skilled in the required technologies. Libraries and open-source packages needed (Model Viewer, Flask extensions) are readily available.

Platform Independence: Depth Spark is a browser-based application, ensuring accessibility across Windows, Linux, Android, and iOS devices without the need for special installations.

Hardware Requirements:
- Client-Side: Minimal requirements (Dual-Core CPU, 4 GB RAM) make it usable even on basic devices.
- Developer-Side: Moderate hardware (Quad-Core CPU, 8 GB RAM) ensures smooth development and testing.

Conclusion on Technical Feasibility: Depth Spark is technically feasible. The required technologies are mature, supported by the community, and compatible with project goals. No major technical barriers are foreseen.

B) Operational Feasibility
Operational feasibility examines how effectively the proposed system solves real-world problems and meets user expectations.

Key Considerations
User-Friendly Interface: The web-based UI is designed to be intuitive and responsive, allowing users with little to no technical background to upload images and generate 3D models effortlessly.

Accessibility: Being browser-based, Depth Spark requires no downloads or installations, greatly increasing accessibility for users across different regions and device types.

Security and Authentication: Secure login/signup mechanisms ensure that user data is protected. Role-based access controls are implemented to protect API usage and manage resources efficiently.

Performance: Real-time progress tracking, automatic error handling, and lightweight output (GLB format) ensure fast, responsive user experiences without lags.

Scalability: The modular architecture allows future expansion to include batch uploads, customization options for models, and integration with cloud storage platforms.

Conclusion on Operational Feasibility: Depth Spark is operationally feasible. The platform aligns well with user needs, enhances workflows for different industries, and promotes widespread adoption through simplicity and reliability.

C) Economic Feasibility

Economic feasibility evaluates whether the project's expected benefits outweigh its costs, considering both short-term investment and long-term sustainability.

Key Considerations
Development Costs:
Minimal direct costs as open-source tools (Flask, SQLAlchemy) are used.

- External costs are limited to Falcon AI API usage (manageable within budget).

No User Costs:

- As a completely free service, Depth Spark enhances its reach without imposing financial barriers on users.

Return on Investment (ROI):

- While Depth Spark itself is free, future expansions (such as premium features, API subscriptions, or B2B services) offer monetization possibilities if desired.
- Building brand reputation and user trust can open pathways for grants, sponsorships, or educational partnerships.

Conclusion on Economic Feasibility: Depth Spark is economically feasible. The initial and recurring costs are manageable, and the potential societal and branding benefits far outweigh the investments.

5. Requirement Analysis

Requirement analysis plays a crucial role in the successful development of any software system. For Depth Spark, a detailed understanding of both functional and non-functional requirements ensures that the final product aligns with user needs, technical capabilities, and project objectives. This chapter outlines the critical requirements that guide the system design and development of the Depth Spark web application.

5.1 Functional and Non-Functional Requirements

A) Functional Requirements

Functional requirements describe the core functionalities and services the system must deliver to its users.

Key Functional Requirements

User Authentication:

- Users can sign up and log in securely through a registration and login system.
- Sessions are protected to ensure secure access and user privacy.

2D Image Upload:

- Users can upload 2D images either via direct file upload or by providing an image URL.
- Supported image formats: JPG, JPEG, PNG, GIF.

Input Validation:

- Uploaded images are validated for file format, file size, and image dimensions before processing.

AI-Powered 3D Generation:

- Depth Spark integrates with Falcon AI to convert 2D images into realistic 3D models.

3D Model Viewer:

- Users can interactively view the generated 3D model using a built-in GLTF viewer with rotate, zoom, pan, and screenshot functionalities.

Download 3D Models:

- Generated 3D models can be downloaded in GLB format for web compatibility and reuse.

Progress Tracker:

- A real-time tracker displays the current status of model generation (e.g., uploading, processing, completed, failed).

Error Handling and Notifications:

- Users receive clear feedback when uploads fail, formats are invalid, or the conversion process encounters issues.

Queue Management:

- Background tasks such as model conversion are handled using a queue system to optimize resource use and ensure fairness.

Logout Functionality:

- Users can log out at any point, securely ending their session.

B) Non-Functional Requirements

Non-functional requirements define the overall quality and performance characteristics of the system.

Key Non-Functional Requirements

Usability:

- The interface is designed to be minimalist, intuitive, and accessible even for non-technical users. The light theme enhance user experience.

Performance:

- The system ensures fast uploads, efficient AI processing, and quick downloads to deliver a seamless experience.

Scalability:

- The architecture supports scaling, both vertically and horizontally, to handle an increasing number of users and AI requests.

Security:

- All user interactions and data transmissions are handled over HTTPS.
- Authentication and API keys are managed securely to prevent unauthorized access.

Portability:

- The application works smoothly across major browsers including Chrome, Firefox, Edge, and is mobile-friendly.

Availability:

- The system is designed for 99% uptime, ensuring high availability with minimal maintenance downtime.

Maintainability:

- Depth Spark follows modular coding standards, includes error logs, and is well-documented, ensuring easy maintenance and future enhancement.

5.2 Software Requirement Specification (SRS)

The Software Requirement Specification (SRS) is a formal documentation of the functional and non-functional expectations from the Depth Spark system.

5.2.1 Purpose

The purpose of this document is to clearly define what the system will do, how users will interact with it, and under what constraints it will operate.

5.2.2 Scope

Depth Spark is a web-based application that allows users to upload 2D images and obtain interactive 3D models using AI-powered conversion. It is designed to be free, accessible, and user-friendly across all major devices and platforms.

5.2.3 Intended Audience

- Developers and Designers of Depth Spark
- Project Mentors and Instructors
- Future Maintenance Teams
- General Users (Students, Small Businesses, Educators, Content Creators)

5.2.4 System Features

Table 5.1 | System Features of Depth Spark

Feature	Description
User Registration and Login	Secure creation and authentication of user accounts.
Image Upload & Validation	Upload images via file or URL and ensure correct input formats.
AI Model Generation	Automatic conversion of 2D images into 3D models using Falcon AI API.
3D Model Viewer	Interactive viewer for rotating, zooming, and examining generated 3D models.
Model Download	Download the 3D model in GLB format.
Progress Tracking	Real-time status updates during model generation.
Error Handling	Display user-friendly error messages for any issues encountered.
Queue Management	Handle multiple requests and background processes efficiently.

5.2.5 External Interfaces

User Interface: Web browser (HTML/CSS/JS) interaction through responsive design.

API Interface: External API (Falcon AI) for sending 2D images and receiving 3D model output.

Database Interface: SQLAlchemy used for user authentication and session management.

5.2.6 Hardware and Software Requirements

Table 5.2 | Hardware and Software Requirements for Depth Spark

Client-Side Requirements	Developer-Side Requirements
Dual-Core CPU	Quad-Core CPU
4 GB RAM	8 GB RAM
720p Display	1080p Display
Web Browser (Chrome, Firefox, Edge)	VS Code IDE + Python Environment
Windows 7/Linux/Android/iOS	Windows 10/Linux

5.2.7 Constraints

- Dependency on external Falcon AI API for 2D-to-3D generation.
- Single-user 3D model generation session (batch processing planned for future updates).

5.2.8 Assumptions and Dependencies

- Users will have a stable internet connection to interact with Depth Spark.
- The Falcon AI API service will maintain availability and performance.

6. Planning and Scheduling

Effective planning and scheduling are vital to the successful execution of any project. For Depth Spark, systematic project planning ensures that tasks are completed on time, resources are utilized optimally, and risks are minimized. This chapter outlines the overall project strategy, detailed timelines, and resource management approaches.

6.1 Project Planning Overview

The planning of Depth Spark follows a phased development model, ensuring that each critical milestone is achieved before moving to the next stage. The project is divided into logical stages, each focusing on key deliverables.

Major Phases:
Phase 1: Project Proposal (16 Feb - 26 Feb)

- Define project goals, objectives, and deliverables.
- Obtain project approval from mentors and supervisors.

Phase 2: Analysis Phase (27 Feb - 05 Mar)

- Perform a detailed literature survey.
- Identify the gaps in current 2D systems.
- Finalize technical feasibility and solution approach.

Phase 3: Design Phase (06 Mar - 17 Mar)

- Create system architecture diagrams.
- Design UI/UX wireframes.
- Plan database schema and API integration flow.

Phase 4: Development Phase (18 Mar - 17 Apr)

- Code frontend (HTML, CSS, JS) and backend (Flask).
- Implement AI integration with Falcon AI API.
- Build 3D viewer and download functionalities.

Phase 5: Testing Phase (18 Apr - 27 Apr)

- Conduct unit testing, system testing, and integration testing.
- Identify and fix bugs.
- Validate performance and load handling.

Phase 6: Implementation (28 Apr - 08 May)

- Final system demonstration and user acceptance testing.

Planning Methodology:

- Weekly milestones were set.
- Continuous integration and testing were emphasized to catch issues early.
- Agile-inspired flexibility was maintained to allow for design adjustments based on testing outcomes.

6.2 Gantt Chart and Resource Allocation

A) Gantt Chart

Table 6.1 | Gantt Chart Timeline for Depth Spark

Task	Duration	Start Date	End Date
Project Proposal	11 days	16 Feb	26 Feb
Analysis Phase	7 days	27 Feb	05 Mar
Design Phase	12 days	06 Mar	17 Mar
Development Phase	31 days	18 Mar	17 Apr
Testing Phase	10 days	18 Apr	27 Apr
Implementation	11 days	28 Apr	08 May

Visual Overview:

- Initial stages (proposal, analysis) occupy ~3 weeks.
- Majority of time (~1 month) is allocated for development.
- Testing and implementation phases are tightly scheduled to ensure timely project delivery.

B) Resource Allocation

Table 6.2 | Resource Allocation Plan for Depth Spark

Team Member	Responsibilities
Uttam Tripathi	AI-Based 3D Model Generator, 3D Model Exporter, Error Handling and Logging, GLTF viewer, API Integration Module
Parul Sharma	User Interface Module, Image Handler, 3D Model Viewer, Queue and Progress Tracker, Authentication
Areeba Shakeel	Instructor Guidance, Periodic Reviews, Feedback
Sakshi Pandey	Mentor Support, Technical Problem Resolution, Quality Assurance

Hardware Resources:

- Development Systems: 8 GB RAM, Quad-Core CPUs, 1080p Displays.
- Testing Devices: Laptops, smartphones (Android/iOS), and tablets.

Software Resources:

- IDE: Visual Studio Code
- Backend: Python 3.x with Flask framework
- Database: SQLAlchemy
- Frontend: HTML5, CSS3, JavaScript
- Version Control: Git

Conclusion
The well-structured planning and scheduling framework adopted for Depth Spark ensures a smooth, systematic approach to development.
With clearly defined timelines, deliverables, and resource allocations, the project minimizes risks and maximizes the probability of on-time, high-quality delivery.

7. Software Development Life Cycle Model

The Software Development Life Cycle (SDLC) represents a structured process used by development teams to build high-quality software efficiently and systematically. It provides a roadmap for planning, creating, testing, and deploying software, while ensuring that it meets user requirements and performs reliably in real-world conditions. For Depth Spark, we adopted the Agile development methodology—an iterative, flexible approach that encourages collaboration, adaptability, and continuous improvement. This chapter discusses the SDLC model implemented in the project, highlights key Agile principles followed, and explains how each development cycle was structured to ensure timely and effective delivery.

7.1 Agile Methodology Overview

Agile is a modern software development methodology that promotes adaptive planning, evolutionary development, early delivery, and continuous improvement. It emphasizes teamwork and the ability to respond quickly to change.

The Agile methodology was chosen for Depth Spark due to the following reasons:

- It allowed flexibility in refining requirements during development.
- It encouraged iterative feedback from mentors and testing phases.
- It suited the academic project environment where frequent improvements were necessary.
- It facilitated better team collaboration through regular standups and communication.

7.2 Agile Process Applied to Depth Spark

The development process of Depth Spark followed a simplified Agile workflow with the following structure:

- Product Backlog: A complete list of requirements and features was prepared at the beginning of the project, such as:
    - User authentication system
    - 2D image upload and validation
    - AI-powered 3D model generation
    - Interactive 3D viewer
    - Download functionality
    - Progress tracking and error handling

- Sprint Planning: The overall timeline was broken down into sprints of 1 to 2 weeks, with specific goals and deliverables for each sprint. Each sprint began with planning meetings to decide the tasks to be completed.

- Sprint Execution: During each sprint, team members focused on developing and integrating specific modules. At the end of each sprint, a working version of the software with partial functionality was ready for testing and review.

- Daily Standups (Communication Sync): Informal daily meetings or check-ins were held between Uttam Tripathi and Parul Sharma to discuss progress, identify obstacles, and align work for the day.

- Sprint Review and Feedback: After each sprint, the completed modules were reviewed with guidance from instructors (Areeba Shakeel and Sakshi Pandey). Feedback was recorded and used to refine the product backlog or improve existing components.
- Sprint Retrospective: Lessons learned and challenges faced during the sprint were analyzed briefly to improve productivity and teamwork in the next cycle.

- Continuous Integration: Code developed in each sprint was integrated into the main project after peer review and testing to ensure consistency, avoid conflicts, and maintain progress transparency.

7.3 Phases of Agile SDLC in Depth Spark

Although Agile is iterative, the following general SDLC phases were embedded within each sprint:

- Requirement Gathering and Analysis: Requirements were initially identified during brainstorming and evolved based on mentor feedback and early user testing.

- Design: Wireframes, UML diagrams, and system flow designs were created in early sprints and refined over time.

- Development: Features were built incrementally. For example:
    - Sprint 1: User login/signup and image upload
    - Sprint 2: AI API integration and queue manager
    - Sprint 3: 3D viewer and download system
    - Sprint 4: Error handling and final UI refinements

- Testing: Each module was tested during the sprint in which it was developed using unit, integration, and system tests. Bugs were fixed before moving to the next sprint.

- Deployment: Although there was no continuous live deployment, a final integrated version was tested and presented at the end.

- Maintenance and Enhancement: Suggestions and bugs from the final user acceptance testing were handled in the final sprint. Future enhancement plans were documented for post-project improvements.

7.4 Benefits of Using Agile in Depth Spark

- Early delivery of working modules enabled continuous testing and faster feedback.
- Collaboration between team members remained consistent and efficient.
- Changes suggested by mentors were quickly implemented in subsequent sprints.
- The modular design ensured smooth integration and scalability of components.
- Time management was easier due to weekly sprint goals and tracking.

8. System Design

System design plays a crucial role in transforming project requirements into a blueprint for development. For Depth Spark, system design ensures a modular, scalable, and user-friendly architecture, supporting smooth 2D-to-3D model conversion.

8.1 System Architecture

The architecture of Depth Spark is based on a multi-tier (client-server) model. It includes the Presentation Layer, Application Layer, and Data Layer.

Key Components
Client Side (Frontend):

- Developed using HTML5, CSS3, and JavaScript.
- Allows users to register/login, upload images (file or URL), view 3D models interactively, and download results.
- Communicates with the server using REST APIs.

Server Side (Backend):

- Built with Flask (Python micro-framework).
- Handles user authentication, image validation, API requests to Falcon AI, and 3D model processing.
- Manages queue and progress tracking for model generation tasks.

External Integration:

- Integrates Falcon AI API for converting 2D images into 3D models.
- Uses Model Viewer library for real-time 3D model rendering in the browser.

Database Layer:

- Uses SQLAlchemy to manage user data, session management, and image processing records.

High-Level Flow:
1. User uploads an image or provides a URL.
2. Backend validates and forwards the image to Falcon AI API.
3. Falcon AI returns and saves a 3D model.
4. Model is displayed via an embedded 3D viewer.
5. User downloads the final GLB file.

8.2 UML Diagrams

Unified Modeling Language (UML) diagrams help visualize the system structure and interaction between components. The following diagrams are essential for understanding Depth Spark:

8.2.1 Use Case Diagram

User Use Case Summary: Users of the Depth Spark system can register, log in, and submit images for 2D to 3D conversion. They can track the progress of their conversion requests and download the final 3D output. This ensures a user-friendly and interactive experience.

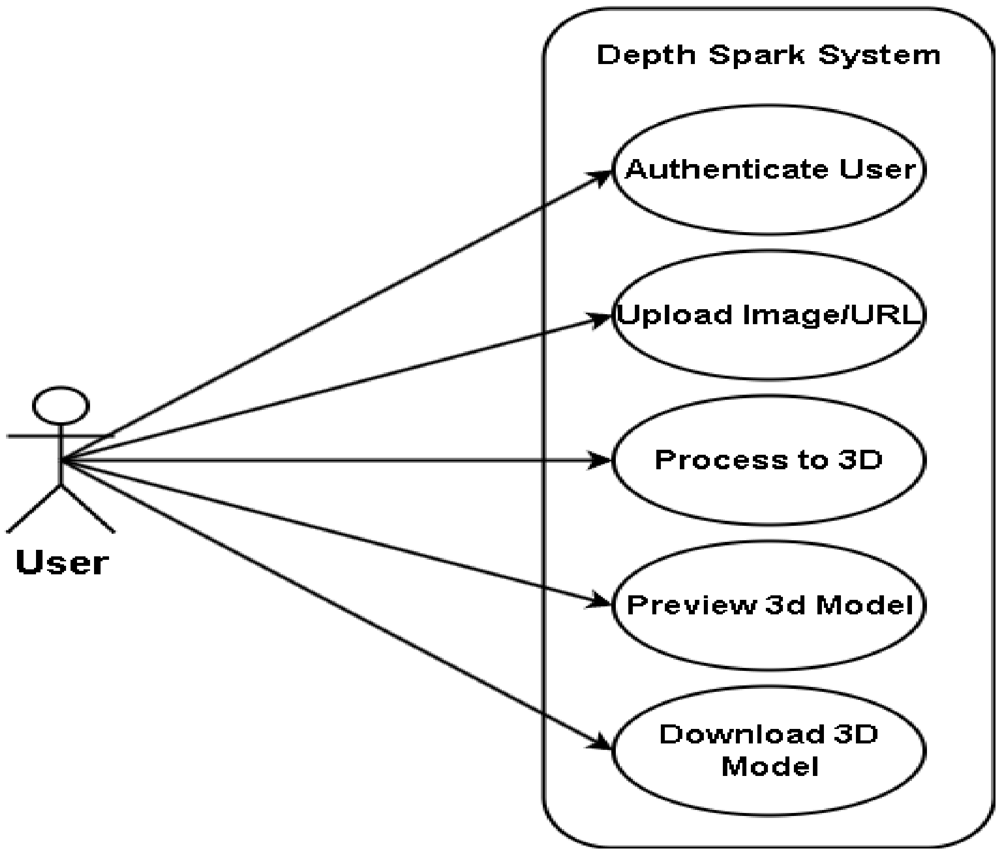


Figure 8.1 | User Use Case Diagram of Depth Spark

Admin Use Case Summary: The admin panel of the Depth Spark system provides essential control and monitoring features. Admins can manage user accounts, track system activity through logs, handle submitted contact forms, and oversee model processing requests. Additionally, they can configure API settings to ensure seamless system integration and performance. These features support the smooth and secure operation of the platform.

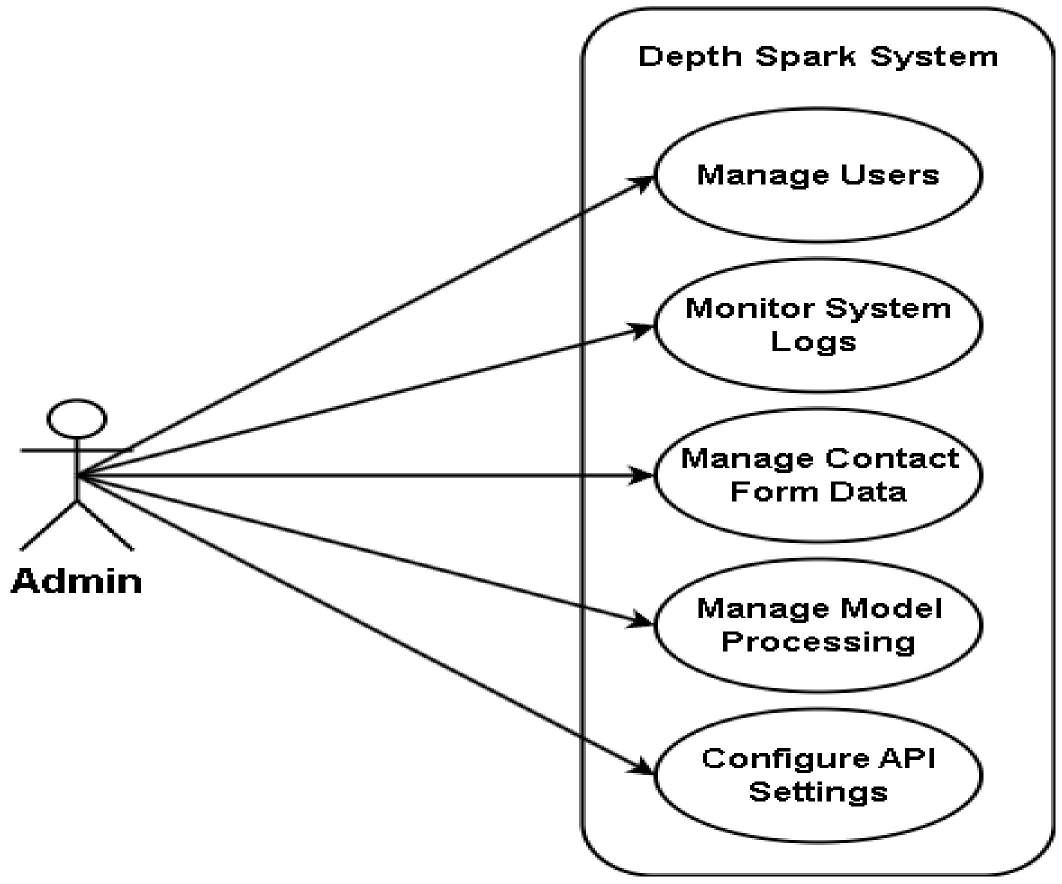


Figure 8.2 | Admin Use Case Diagram of Depth Spark

8.2.2 Data Flow Diagram

Data Flow Diagrams (DFDs) illustrate how information flows through the Depth Spark system. They provide a visual representation of the data inputs, processing stages, external entities, and data outputs within the platform. Depth Spark's DFD is structured across two levels:

Level 0 DFD

The Level 0 DFD represents the entire Depth Spark system as a single process interacting with two external entities: the User and the Falcon AI API.

External Entities:

- User: Initiates image upload and interacts with the system to view or download the generated 3D model.
- Falcon AI API: Processes the input 2D image and returns a generated 3D model.

Major Data Flows:

- The User uploads a 2D image to the Depth Spark System.
- The System sends the image to the Falcon AI API for 3D model generation.
- The API returns a processed 3D model to the System.
- The System allows the User to preview and download the model.

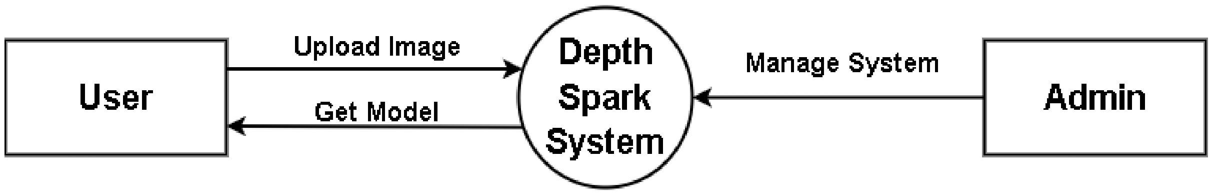


Figure 8.3 | Level 0 DFD of Depth Spark

Level 1 DFD

The Level 1 DFD expands the internal workings of the Depth Spark system by breaking down the main process into five sub-processes.

Processes:

- User Authentication – Handles login and signup operations.
- Image Upload & Validation – Accepts image uploads and verifies format and size.
- AI Model Generation – Sends validated image to the Falcon AI API and receives the 3D model.
- Model Export and Storage – Converts and stores the 3D model in GLB format.
- Model Preview & Download – Displays the model in a viewer and allows users to download it.

Data Stores:

- D1: User Database – Stores user credentials and session data
- D2: 3D Model Repository – Stores generated 3D models in downloadable format.

External Entities:

- User
- Falcon AI API

Major Data Flows:

- User provides login info → Authentication process → User DB
- Image uploaded by User → Validation process
- Image sent to API → 3D model returned → Model Repo
- Final 3D model → Displayed to User → Download

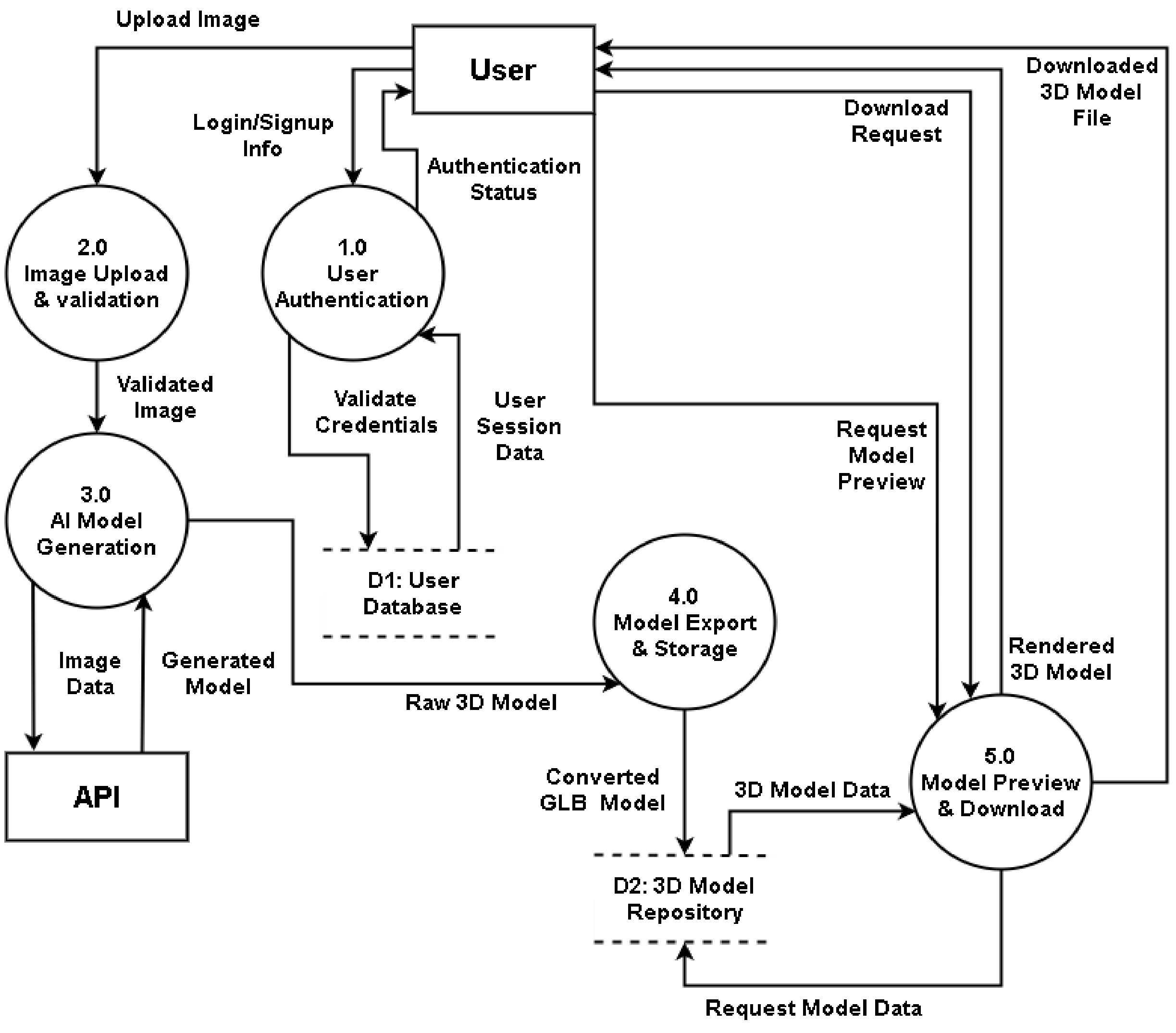


Figure 8.4 | Level 1 DFD of Depth Spark

8.2.3 Proposed System Diagram

User Flow: Users sign up, upload images, track processing, view, and download 3D models.
Admin Functions: Admins monitor activity, view logs, manage APIs, and review feedback.

Core Features:

- Real-time queue and progress tracking
- Interactive 3D model viewing
- Secure authentication system
- Error logging and recovery

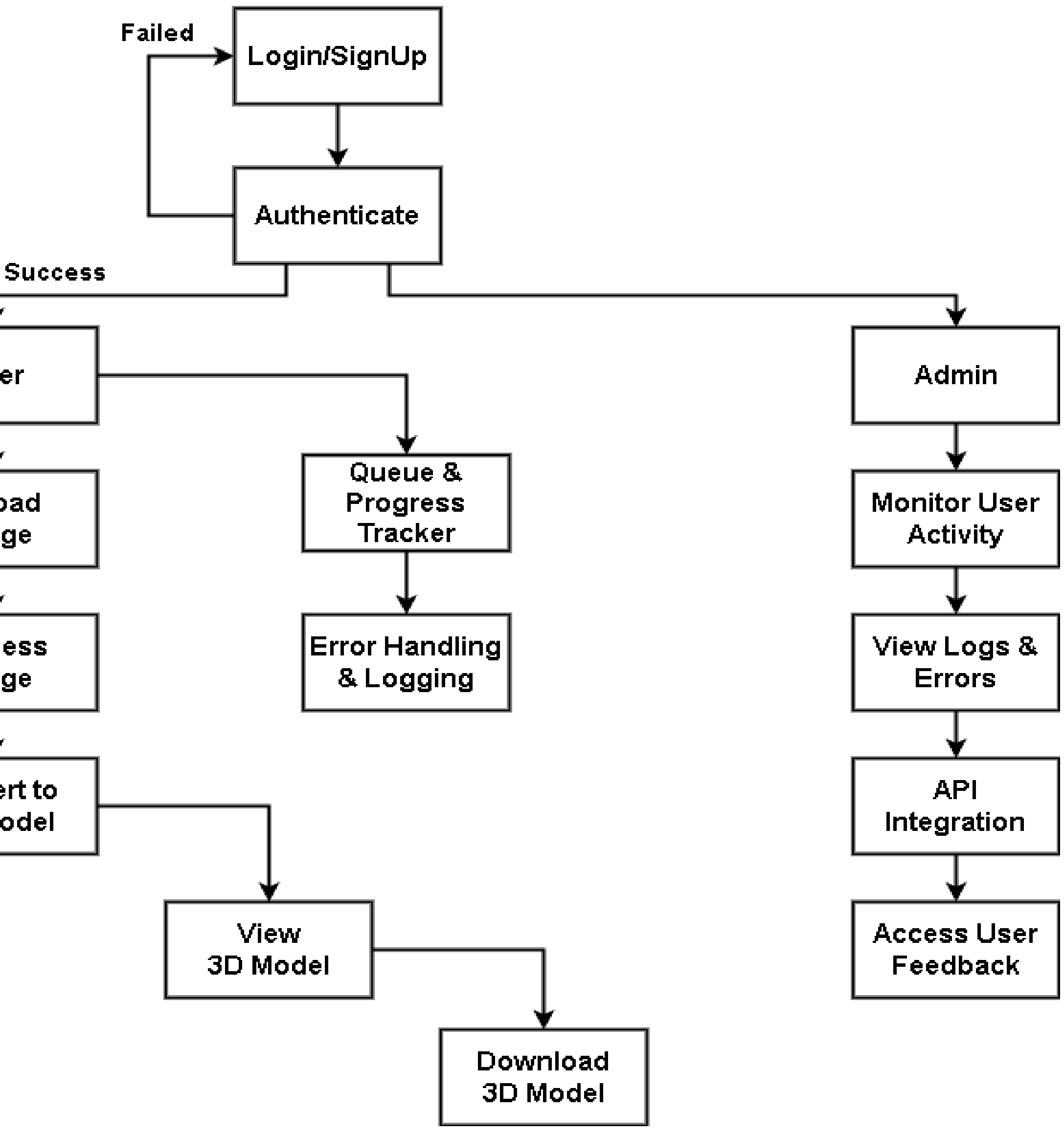


Figure 8.5 | Proposed System Diagram of Depth Spark

8.2.4 Class Diagram

- UserInterface & Authentication: Handle user login/signup, secure access, and front-end operations like image upload and 3D model viewing.
- ImageHandler: Validates and prepares uploaded images for processing.
- AIModelGenerator: Converts 2D images into 3D models using AI.
- APIIntegration: Connects to third-party services like Falcon AI.
- QueueProcessor & ErrorHandler: Manages task queues and logs errors.
- ModelExporter & ModelViewer: Prepares and renders downloadable 3D models.
- GLTFViewer: Offers advanced viewing for GLTF file formats.

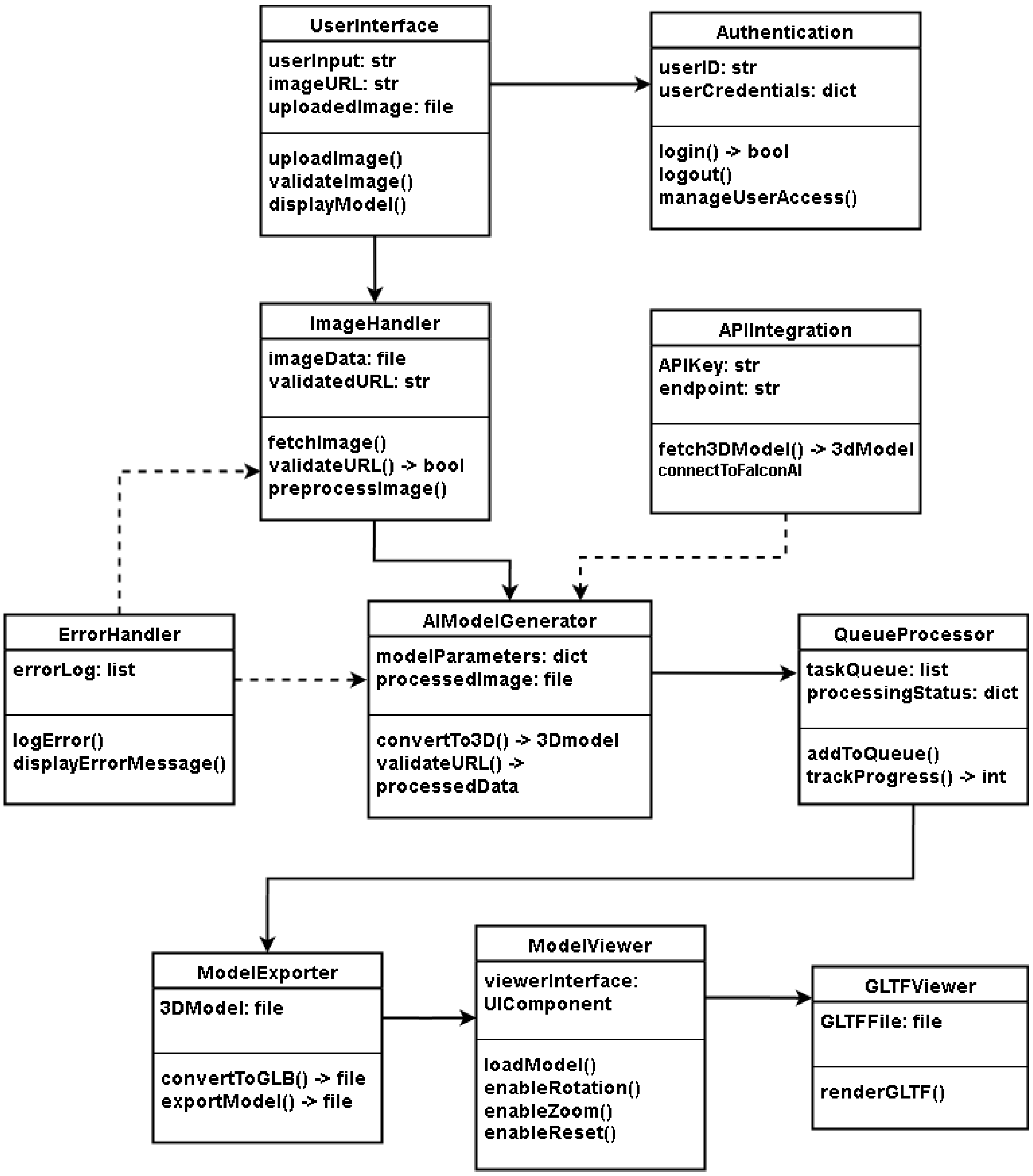


Figure 8.6 | Class Diagram of Depth Spark

8.3 Data Dictionary & Interface Design

A) Data Dictionary

Table 8.1 | Data Dictionary for Depth Spark

Table Name	Field Name	Data Type	Description
User	id	Integer (PK)	Unique user identifier
User	username	Varchar(50)	Username of the user
User	email	Varchar(100)	Email address of the user
User	password	Varchar(100)	Encrypted password for authentication
User	reset_token	Varchar(6)	Token sent to user for password reset
User	Reset_token_expiry	Datetime	Expiration time of the reset token
Contact	id	Integer (PK)	Unique identifier for contact form data
Contact	name	Varchar(100)	Name of the user
Contact	email	Varchar(100)	Email address of the user
Contact	subject	Varchar(200)	Subject for contacting
Contact	message	Text	Message of the user

B) Interface Design

Key User Interfaces

Home Page:

- Welcome to Depth Spark: An Introduction
- Unlock the Power: Our Key Features
- Getting Started: Simple Steps to Use Depth Spark
- Your Questions Answered: FAQs
- Ready to Dive In? Take Action Now!
- Join Our Community: Login or Sign Up

Conversion Page:

- Upload Your Image: File or URL
- See Before You Convert: Image Preview
- Stay Informed: Real-time Progress

About Page:

- The Depth Spark Journey: Our Story
- Why We Exist: Mission & Vision
- Our Foundation: Key Values
- The People Who Make It Happen: Our Team
- What Drives Us: Our Technology

3D Model Viewer:
- Experience Your Creation: View the 3D Model
- Take Control: Rotate, Zoom, Reset, Move
- Set the Scene: Skybox Options
- Illuminate Your Vision: Environment Lighting Choices
- Capture and Keep: Screenshot & Download

Authentication Pages:
- Welcome Back: Login (with helpful validation)
- Join Depth Spark: Sign Up (with password security tips)
- Trouble Signing In? Forgot Password (with email verification)
- Confirm Your Identity: Email Verification Page
- Reset Your Password: Enter and Confirm New Password Page

Contact Us Page:
- Reach Out: Contact Us Directly

Error Messages:
- Oops! Something Went Wrong: (Followed by friendly, specific details for invalid uploads, failed model generation, or system downtime)

Design Considerations:
- Mobile Responsive
- Minimalistic UI with simple navigation
- Color Scheme: Clean (white/light backgrounds) with accent colors for buttons and highlights

Conclusion

The system design of Depth Spark focuses on simplicity, modularity, and scalability.
Through clear system architecture, structured UML diagrams, a well-planned data dictionary, and intuitive interface designs, the platform ensures a seamless user experience and robust backend performance.

9. Module Description

The architecture of Depth Spark is divided into several well-defined modules, each responsible for handling specific system functionalities. This modular approach ensures that the system remains scalable, maintainable, and efficient. Each module works in coordination with others to deliver a seamless user experience for 2D-to-3D model conversion. The following sections describe the main modules of the Depth Spark system.

Table 9.1 | Modules Distribution

Team Member	Modules
Uttam Tripathi (ME)	AI-Based 3D Model Generator, 3D Model Exporter, Error Handling and Logging, GLTF viewer, API Integration Module
Parul Sharma	User Interface Module, Image Handler, 3D Model Viewer, Queue and Progress Tracker, Authentication

9.1 AI-Based 3D Model Generator

The AI-Based 3D Model Generator module is the core engine that transforms 2D images into 3D models. Using sophisticated AI algorithms, this module interacts with external AI platforms to automate the generation of realistic 3D structures from single 2D inputs.

The generator is responsible for:

- Sending validated images to the integrated Falcon AI API.
- Managing asynchronous requests for 3D model generation.
- Receiving the generated 3D models and preparing them for export.

By utilizing advanced depth estimation techniques, the module enables quick, scalable, and reliable 3D content creation without manual modeling efforts.

9.2 3D Model Exporter

Once a 3D model has been generated, the 3D Model Exporter module processes and converts it into a standard format suitable for web visualization. The chosen output format is GLB (GL Transmission Format Binary), known for its compact size and efficient performance.

Primary tasks of this module include:

- Converting AI-generated models into the GLB format.
- Optimizing the model files for smooth viewing and fast downloads.
- Managing storage for download availability.

This module ensures that users receive lightweight, high-quality 3D assets ready for use in various applications.

9.3 Error Handling and Logging

To ensure system reliability and ease of maintenance, the Error Handling and Logging module monitors and records system activities and issues. It provides meaningful feedback to both users and developers.

Key functionalities include:

- Displaying user-friendly error messages for invalid uploads or processing failures.
- Logging system events, API responses, and operational anomalies.
- Assisting in debugging and future system improvements by maintaining detailed logs.

Proper error handling improves user trust and contributes to system robustness.

9.4 GLTF Viewer

The GLTF Viewer module is specifically designed to handle the rendering and display of 3D models in the GL Transmission Format (.gltf or .glb). This module complements the 3D Model Viewer but focuses on adhering to standardized 3D viewing specifications.

The GLTF Viewer provides:

- Lightweight and fast loading of 3D models.
- Compatibility with various devices and web browsers.
- High-fidelity rendering of textures, materials, and lighting.

By utilizing GLTF standards, Depth Spark ensures maximum model portability and quality.

9.5 API Integration Module

The API Integration Module handles communication between Depth Spark and external AI services such as the Falcon AI platform. This integration is crucial for offloading the heavy computational task of 2D-to-3D conversion to specialized AI servers.

Responsibilities include:

- Establishing secure HTTP connections with external APIs.
- Managing data transmission, error handling, and API authentication.
- Receiving and validating the 3D models returned by the API.

This module ensures seamless interoperability and smooth functionality across the system.

9.6 User Interface Module

The User Interface (UI) Module is the first point of contact between the user and the Depth Spark system. It is responsible for providing a simple, intuitive, and responsive interface where users can upload 2D images either by file selection or entering an image URL. The design of the UI ensures that users with minimal technical expertise can easily navigate the platform.

Key functionalities offered through the User Interface Module include:

- Providing forms for image upload (file and URL-based).
- Displaying real-time progress during model generation.

- Offering interactive 3D model viewing features.
- Allowing easy download of the generated 3D models.

The user interface follows a clean and minimalist design to enhance usability across desktop and mobile platforms.

9.7 Image Handler

The Image Handler module plays a crucial role in managing the images uploaded by users. It is responsible for validating the input image, whether provided through a direct upload or a URL, ensuring that the system processes only valid and supported formats.

The main responsibilities of the Image Handler include:

- Verifying image formats (JPG, JPEG, PNG, GIF).
- Fetching images from URLs provided by the user.
- Checking the image size and dimensions before processing.
- Temporarily storing validated images securely for conversion.

This module acts as a gatekeeper to ensure that only clean, compatible, and safe images enter the AI processing pipeline.

9.8 3D Model Viewer

The 3D Model Viewer provides users with the ability to interact with their generated models directly through the web interface. It offers real-time visualization capabilities, enabling users to examine the models from every angle.

The Viewer allows:

- Rotation, zooming, and panning of the 3D models.
- Capturing screenshots of specific model views.
- Smooth rendering of GLB models using Model Viewer technology.
- Users can change the background environment (skybox image) to enhance visual presentation.
- Users can toggle between different lighting configurations (e.g., studio light, natural light) to suit the model display needs.

Designed for performance, the Viewer ensures an immersive and responsive experience even on lower-end devices.

9.9 Queue and Progress Tracker

Depth Spark's Queue and Progress Tracker module ensures efficient management of background tasks, especially when multiple users are processing models simultaneously.

Its main responsibilities are:

- Managing the queuing of AI processing requests.
- Monitoring task status (pending, processing, completed, failed).
- Sending real-time progress updates to the frontend for user display.

This module is essential for maintaining system stability and providing users with continuous feedback during the model generation process.

9.10 Authentication

The Authentication module secures user access to the platform's features. It allows users to create accounts, log in securely, and maintain private access to their uploads and models.

The Authentication system supports:

- Secure account registration with encrypted password storage.
- Session management to ensure authenticated access.
- Protection against unauthorized access and common security threats.

This module plays a fundamental role in ensuring user privacy and maintaining overall platform security.

10. Implementation Strategy

The successful execution of any software project depends heavily on a well-planned and efficient implementation strategy. For Depth Spark, the implementation approach was divided into clearly defined stages, ensuring that development activities were structured, predictable, and manageable. This chapter discusses the tools selected for development, the phased development plan, and the strategy for integrating various modules into a unified, functional system.

10.1 Tools and Development Phases

The choice of tools and technologies for Depth Spark was made to balance ease of development, system performance, scalability, and future extensibility. A careful selection of reliable and modern tools helped streamline both backend and frontend development.

The major tools used in Depth Spark include:

- Programming Language: Python 3.x, selected for its simplicity, rich ecosystem, and strong support for AI and web development.
- Backend Framework: Flask, a lightweight and flexible framework ideal for building scalable web applications.
- Frontend Technologies: HTML5, CSS3, and JavaScript, enabling a responsive and interactive user interface.
- Database: SQLAlchemy, offering a clean and efficient way to interact with databases using Python.
- External APIs: Falcon AI API, integrated to handle the 2D-to-3D conversion process through AI.
- 3D Visualization: Model Viewer, used to render 3D models in the browser interactively.
- Development Environment: Visual Studio Code, selected for its extensive plugin support, ease of use, and compatibility with Python development.
- Version Control: Git, ensuring code management, collaboration, and version tracking throughout the project.

The implementation of Depth Spark was divided into the following development phases:

Phase 1: Setup and Environment Configuration
This phase involved setting up the required development environments, installing libraries, and configuring servers for both frontend and backend development.

Phase 2: Frontend UI Development
During this phase, the frontend interface was designed and built. Responsive pages for uploading images, viewing progress, and displaying 3D models were created.

Phase 3: Backend API Development
In parallel with frontend development, the core server functionalities were developed, including user authentication, image upload handling, Falcon API integration, and model management.

Phase 4: 3D Model Viewer Integration
The Model Viewer was integrated into the frontend, allowing real-time interaction with the generated models.

Phase 5: Testing and Debugging
Comprehensive unit tests, system tests, and integration tests were carried out to identify and fix bugs, verify feature completeness, and ensure the overall stability of the system.

Phase 6: Implementation and User Acceptance Testing
After completing internal validations, Depth Spark underwent simulated user acceptance testing in a controlled environment, followed by adjustments based on feedback.

By following these organized development phases, the project maintained a systematic workflow, reducing potential risks and ensuring a consistent pace of progress.

10.2 Module Integration

Integrating various independent modules into a cohesive and smoothly functioning system was a critical part of the implementation strategy for Depth Spark. Each module was developed in isolation to ensure that it performed its core functionality independently and then integrated systematically during the later stages.

The module integration strategy involved the following steps:

Interface Matching: The input/output structure between modules (e.g., Image Handler ➔ AI Generator ➔ Model Exporter) was standardized early during design to ensure smooth data flow.

API Connectivity: The backend integration with the Falcon AI API was tested independently before connecting it to the frontend upload systems to avoid cascading errors.

Incremental Integration: Modules were not integrated all at once. Instead, they were added one by one (starting from authentication, followed by image handling, AI generation, model export, and viewer integration), allowing targeted testing at each stage.

Middleware Layer: A small middleware layer within the backend handled task queuing, progress tracking, and coordinated communications between the user interface and AI processing pipeline.

Testing After Each Merge: After integrating each major module, regression testing was performed to ensure that previously working features remained stable.

Error Handling and Recovery: Special attention was given to error propagation between modules so that failures in one part (such as AI processing) could be gracefully reported to users without crashing the entire system.

Through careful planning, interface standardization, incremental module addition, and rigorous testing, Depth Spark achieved a seamless integration of its independent modules into a single, coherent application that delivers reliable performance to end users.

Conclusion
The implementation strategy of Depth Spark emphasized robust tool selection, phased development, and cautious module integration.
This approach allowed the project to maintain flexibility, adapt to challenges efficiently, and deliver a polished, scalable, and user-friendly web application.

11. Testing

Testing is a crucial phase of the software development lifecycle, ensuring that the final product meets functional requirements, maintains quality standards, and delivers a smooth user experience. For Depth Spark, extensive testing activities were carried out at multiple stages of development to detect and resolve bugs early, verify system integration, and guarantee stable performance across different devices and environments. This chapter discusses the various testing methodologies adopted, details the testing phases, and outlines the bug reporting and resolution process followed during the project.

11.1 Testing Methodologies

The testing strategy for Depth Spark incorporated both manual and automated methods to validate functionality, performance, security, and usability. The following testing methodologies were employed:

Black Box Testing: Focused on testing the system's functionality without knowledge of the internal code structure. User actions like uploading images, interacting with the 3D viewer, and downloading models were tested for correctness.

White Box Testing: Involved testing the internal logic of individual modules, especially for API handling, database operations, and queue management, ensuring that all functions behaved as expected.

Regression Testing: Conducted whenever new features were integrated to ensure that existing functionalities were not broken due to new updates.

Performance Testing: Focused on measuring the application's response time, particularly during AI model generation and 3D rendering, ensuring that the system remained responsive under typical load conditions.

Security Testing: Validated that user authentication mechanisms were robust, sessions were properly managed, and no unauthorized data access was possible.

The combination of these testing approaches ensured a well-rounded evaluation of Depth Spark before its deployment.

11.2 Unit, System and Integration Testing

Testing was organized into three main phases to ensure thorough coverage of both individual components and their interaction within the overall system.

Unit Testing: Unit tests were created for individual modules to verify that each function performed correctly in isolation. For example:

- Image validation functions in the Image Handler were tested for various file formats and size limits.
- Authentication functions were tested for secure login and registration workflows.
- API communication functions were tested to handle successful and failed responses from Falcon AI gracefully.

Unit testing was essential in quickly identifying coding errors early during development.

System Testing: System testing was conducted after integrating all modules to ensure that Depth Spark, as a complete system, fulfilled the specified requirements. This involved end-to-end testing of workflows, including:

- User registration ➔ Image upload ➔ Model generation ➔ 3D model preview ➔ Download process.
- Handling of invalid scenarios, such as uploading unsupported file formats or losing internet connection during model processing.

System testing helped identify any inconsistencies or mismatches between modules and provided a real-world validation of system behavior.

Integration Testing: Integration testing focused on verifying the correct interaction between different modules. Special attention was given to:

- Data passing between the Image Handler and AI Generator.
- Proper synchronization between Queue Manager and Progress Tracker.
- Seamless interaction between the 3D Model Exporter and 3D Viewer.

Testing individual integration points minimized potential issues that could arise from module dependencies, ensuring a smooth and reliable user experience.

11.3 Bug Reporting and Resolution

A structured bug reporting and resolution process was followed to maintain high software quality during the development and testing phases. The bug management process involved the following steps:

Bug Tracking: All bugs were logged systematically using a bug-tracking spreadsheet, noting details such as description, severity (Critical, Major, Minor), status (Open, In Progress, Resolved), and assigned developer.

Bug Prioritization: Critical bugs that blocked essential functionality (such as login failures or AI integration issues) were given the highest priority and resolved immediately. Minor UI inconsistencies were scheduled for later corrections.

Regular Bug Reviews: Daily or bi-weekly team reviews were held to update bug statuses, reassign unresolved issues, and monitor overall progress toward stabilization.

Resolution and Retesting: Once bugs were fixed, the relevant modules were retested to verify the correctness of the fix and to ensure that no new issues had been introduced (regression testing).

Some examples of reported and resolved bugs during the project include:

- Uploading large images sometimes caused timeouts (resolved by optimizing file size validation).
- Progress Tracker not updating correctly in slow internet conditions (fixed by improving backend event handling).
- Mobile responsiveness issues in the 3D Viewer (resolved through enhanced CSS media queries).

12. Security Implementation

Security is a fundamental aspect of modern web applications. In Depth Spark, particular emphasis was placed on protecting user information, securing communication with external services, and ensuring the integrity of the system. This chapter outlines the key measures taken to implement a secure authentication system, access control, and secure API integration within Depth Spark.

12.1 Authentication and Access Control

Authentication and access control mechanisms were designed to protect user accounts, manage user sessions securely, and prevent unauthorized access to system features. The authentication system includes the following major features:

Secure Account Registration: Users can create accounts using a valid email and password. Passwords are stored in the database using encryption.

Login and Session Management: During login, the system verifies user credentials securely and establishes authenticated sessions using server-side session handling.

Access Restrictions: Sensitive routes, such as image uploads, model generation, and downloads, are protected to allow access only to authenticated users. Unauthorized access attempts are automatically redirected to the login page.

Session Expiry and Logout: Sessions expire after a period of inactivity to reduce security risks. Users can also manually log out at any time, invalidating their active session immediately.

These authentication and access control mechanisms ensure that user data remains private and that only authorized users can interact with Depth Spark's protected resources.

12.2 Secure API Integration

Since Depth Spark relies heavily on an external service, Falcon AI API, for its core 2D-to-3D conversion functionality, it was critical to implement a secure API integration strategy. Security practices for API integration include:

Secure API Keys Management: API credentials are not hardcoded in the application code. Instead, they are stored securely in server-side environment variables (.env files) and accessed through the backend only. This prevents accidental exposure of sensitive keys.

HTTPS Communication: All communications between Depth Spark and the Falcon AI API are carried out over secure HTTPS channels, ensuring that data in transit (such as user images and generated models) is encrypted and protected from eavesdropping or tampering.

Request Validation: Before sending any data to the API, requests are thoroughly validated to ensure they meet expected formats and security requirements, reducing the risk of malformed or malicious data submission.

Error Handling and Logging: API responses are monitored carefully. Any unauthorized, timeout, or unexpected responses from the API are logged securely for review without exposing sensitive error information to end-users.

Rate Limiting and Throttling: Depth Spark implements request rate limiting when communicating with the Falcon AI API to prevent abuse and maintain service availability, both for Depth Spark and the API provider.

These measures ensure that Depth Spark's integration with Falcon AI remains secure, stable, and resilient against threats targeting third-party service communication.

Conclusion
By implementing strong authentication practices, access control measures, and secure API integration techniques, Depth Spark ensures that both user data and system integrity are well-protected.

13. Performance Analysis

Performance analysis is a critical step in evaluating the efficiency, speed, and reliability of a software application under various conditions. For Depth Spark, performance testing was conducted to ensure that the system could handle real-world usage scenarios, process user requests promptly, and deliver a seamless experience even when multiple users interacted with the platform simultaneously. This chapter describes the approach taken to measure Depth Spark's performance and the metrics collected during load testing.

13.1 Load Testing and Metrics

Load testing was performed to evaluate how well Depth Spark manages concurrent user activities, particularly focusing on image uploads, AI model generation, 3D model rendering, and file downloads. The testing aimed to ensure that the system maintained acceptable response times and did not crash or slow down excessively under normal and moderately heavy load conditions.

During load testing, simulated user sessions were created, mimicking realistic behaviors such as logging in, uploading 2D images, generating 3D models, interacting with the 3D viewer, and downloading the final models. Both single-user and multi-user environments were tested to monitor performance under different levels of activity.

The key performance metrics measured included:

- Average Response Time: The time taken by the system to respond to typical user actions. For Depth Spark, the average response time during normal load remained within 2-3 seconds for image upload and under 5 seconds for initiating 3D model generation.
- Model Generation Time: The time taken by Falcon AI to process an image and return a 3D model varied depending on image size, averaging around 18–25 seconds, which was acceptable for the complexity of the operation.
- System Throughput: The number of user requests handled successfully per minute. Depth Spark maintained a steady throughput even with 30 concurrent users without significant degradation in performance.
- Error Rate: The number of failed or dropped user actions during testing. The error rate was consistently below 1%, mainly caused by intentional network disconnections during testing scenarios.
- 3D Viewer Frame Rate: When interacting with the 3D model, the viewer maintained smooth rendering at 45–60 FPS (frames per second) across most modern devices, ensuring a responsive visualization experience.

The testing environment included simulations over different network conditions (Wi-Fi, 4G mobile data) and device types (desktop, laptop, smartphone) to assess the performance variability. Minor latency increases were observed on mobile networks; however, they did not critically impact usability.

In addition to functional load testing, basic stress tests were also conducted by rapidly increasing the number of simultaneous upload requests. Depth Spark demonstrated resilience by queuing tasks properly without crashing or rejecting user inputs unnecessarily, thanks to the implemented queue management system.

From the collected metrics and observed system behavior, it was concluded that Depth Spark meets its performance objectives under typical usage conditions. With minor optimizations and further scaling on the server side, it can comfortably support a growing number of users over time without compromising quality.

Conclusion
Depth Spark's performance analysis confirmed that the application remains stable, responsive, and efficient under load.
By maintaining low response times, high throughput, minimal error rates, and smooth 3D model interaction, the platform successfully delivers a user experience aligned with modern web application standards.

14. Cost Estimation

Cost estimation is a critical component of project planning that helps determine the financial feasibility, sustainability, and resource requirements of a solution. For Depth Spark, a web-based app for converting 2D images into 3D models, cost control was a major consideration from the outset. Our team placed a strong emphasis on affordability while ensuring the project maintained its functional and technical quality. This chapter provides a comprehensive breakdown of the cost factors considered during the development and post-deployment planning stages.

14.1 Estimation and Budget Analysis

The cost estimation for Depth Spark was divided into key categories including development costs, third-party API usage, and maintenance provisions. The goal was to keep the project as cost-effective as possible without compromising performance, security, or user experience. The cost estimation for Depth Spark was methodically categorized into the following areas:

1. Development Costs
2. Third-Party API Integration
3. Maintenance and Upgrade Provisions

Each component was analyzed to identify opportunities for optimization and efficiency, with the goal of building a budget-friendly, high-performance product.

Development Costs

The heart of Depth Spark's affordability lies in its use of open-source technologies. The backend of the application was developed using Flask, a lightweight Python framework known for its speed and flexibility. For database operations, we utilized SQLAlchemy, and for 3D visualization, we integrated Model Viewer, an open-source WebGL-based viewer. Key benefits of using open-source tools:

- Zero licensing fees, reducing software acquisition costs.
- Wide community support, aiding rapid problem resolution and updates.
- Compatibility across platforms, accelerating integration and testing.

Moreover, the project was executed by two in-house student developers—Uttam Tripathi and Parul Sharma—as part of their final-year academic curriculum. This contributed to major cost savings, as:

- No salaries or freelance hiring expenses were needed.
- Development timelines were managed efficiently without the need for external consultants.

These choices significantly reduced the financial burden while maintaining high development standards.

Third-Party API Usage

A defining feature of Depth Spark is its ability to convert 2D images into 3D models, made possible through the Falcon AI API. This API enables automatic model generation with minimal input, enhancing the user experience. API Cost Considerations:

Free Tier: During the initial development and testing phases, the Falcon AI API's free tier provided sufficient access to test and demonstrate functionality. This allowed the team to validate the core feature without incurring upfront expenses.

To plan for this, we factored in:

- API pricing flexibility for startups and educational use
- Tiered budgeting for gradual platform growth
- Options to monitor and control API usage to avoid cost overruns

This forward-thinking API strategy ensures that Depth Spark remains affordable now, with clear budgeting pathways for future scale.

Maintenance and Upgrade Provisions
After deployment, ongoing maintenance is essential to keep Depth Spark functional, secure, and user-friendly. Even if the initial development costs are minimal, the long-term health of the platform depends on consistent updates and system monitoring. Key Maintenance Activities:

- Software updates: Regular updates of dependencies like Flask, SQLAlchemy, and Model Viewer to ensure compatibility and security.
- Security patches: Monitoring vulnerabilities and deploying patches to keep user data and operations secure.
- Bug fixes and performance optimization: Addressing user-reported issues and improving loading times or model accuracy.
- User data management: Backups and data validation to ensure data integrity and compliance.
- Feature enhancements: Incremental upgrades based on user feedback and evolving technology trends.

Initially, these activities will be handled internally by the developers themselves, making the process cost-effective and agile. No external maintenance contracts or third-party service agreements are planned in the early phase.

Conclusion
By strategically selecting free and open-source technologies, relying on in-house talent, and taking advantage of free-tier services like Falcon AI, Depth Spark has been designed to operate with minimal upfront investment. This resourceful approach ensures that the platform:

- Delivers strong performance and functionality,
- Stays secure and user-friendly,
- And remains scalable without introducing major financial strain.

In essence, Depth Spark exemplifies how innovation, resourcefulness, and budget-conscious planning can come together to deliver a technically sophisticated solution without requiring extensive financial backing. This makes it a strong model for student-led tech initiatives and an inspiration for lean development practices in academic and startup environments.

15. Results and Discussion

After completing the design, development, and testing phases of the Depth Spark project, the final system output was carefully evaluated against the initial objectives. The platform successfully delivers a functional and user-friendly web application that converts 2D images into interactive 3D models using AI-based technology. This chapter discusses the final system behavior, key outcomes, and the overall workflow of Depth Spark as experienced during the final implementation and testing stages.

Depth Spark demonstrates that with the right combination of AI, simple web technologies, and open-source tools, it is possible to democratize 3D model generation without requiring specialized hardware or technical expertise. Users are able to interact seamlessly with the platform, from uploading images to downloading ready-to-use 3D models in a lightweight GLB format. This fulfills the project's core objective of making 3D technology accessible, free, and easy for everyone, especially students, educators, and small businesses.

15.1 Final System Output and Workflow

The final version of Depth Spark provides a clean, responsive, and efficient system that smoothly handles the complete workflow from 2D image input to 3D model output. Each module interacts seamlessly with others, ensuring a reliable experience for users with minimal waiting times and intuitive navigation. The standard workflow of the final Depth Spark system can be summarized as follows:

1. User Registration and Authentication: New users can quickly create an account using their email and a secure password. Existing users can log in and access the platform instantly. Sessions are managed securely, ensuring that users remain authenticated across their interactions.

2. Image Upload or URL Submission: Authenticated users can either upload a 2D image file (JPG, PNG, GIF formats) or submit an image URL. The system validates the file for size, format, and integrity before proceeding further. Any invalid inputs are promptly flagged with clear error messages.

3. AI-Based 3D Model Generation: Once the image is successfully uploaded, Depth Spark sends it securely to the Falcon AI API for 2D-to-3D conversion. This process is handled in the background, and users can see real-time progress updates displayed through the progress tracker on the interface.

4. Interactive 3D Model Preview: After successful model generation, users are directed to the 3D viewer. The GLB model is rendered interactively in the browser, allowing users to rotate, zoom, pan, and inspect the 3D object in real time. The viewer maintains high frame rates for a smooth, responsive interaction.

5. Model Download: Users can easily download the generated 3D model in GLB format with a single click. The downloaded files are optimized for use in e-commerce websites, educational presentations, AR/VR applications, and other platforms.

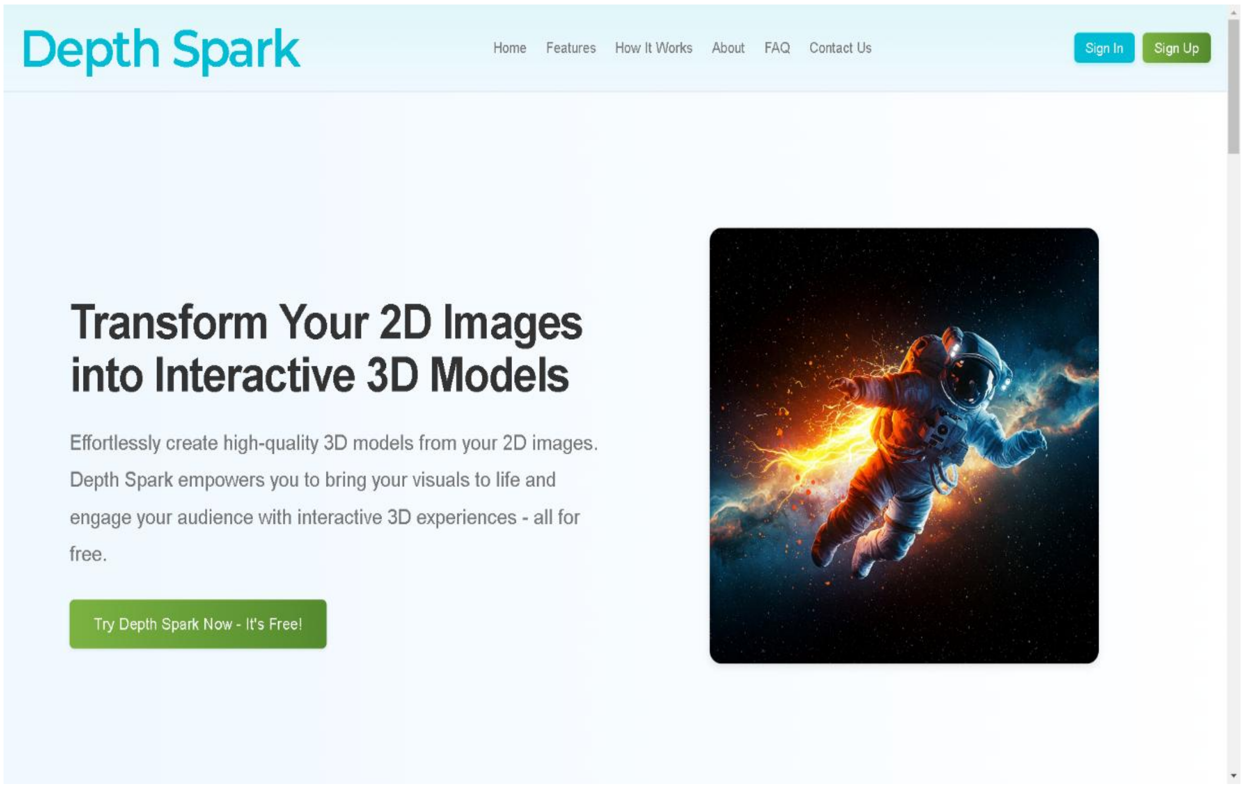
6. Session Management and Logout: Users must manually log out after completing their tasks. Sessions do not expire automatically, so users are encouraged to log out to maintain account security.

Throughout the workflow, the system maintains high responsiveness, minimal error rates, and smooth transitions between modules. The background queue management system ensures that simultaneous requests are handled properly without server crashes or significant slowdowns, even under moderate user load conditions. The final system output meets or exceeds the expected requirements in the following ways:

- Accuracy: Generated 3D models closely match the structures of the input 2D images, offering realistic depth and proportions.
- Performance: Average response and generation times remain within acceptable thresholds even during stress testing.
- Security: User accounts, uploaded images, and generated models are handled securely with proper authentication and encrypted API communications.
- Usability: The web interface is simple, mobile-responsive, and intuitive, making it accessible even to users without prior experience with 3D software.

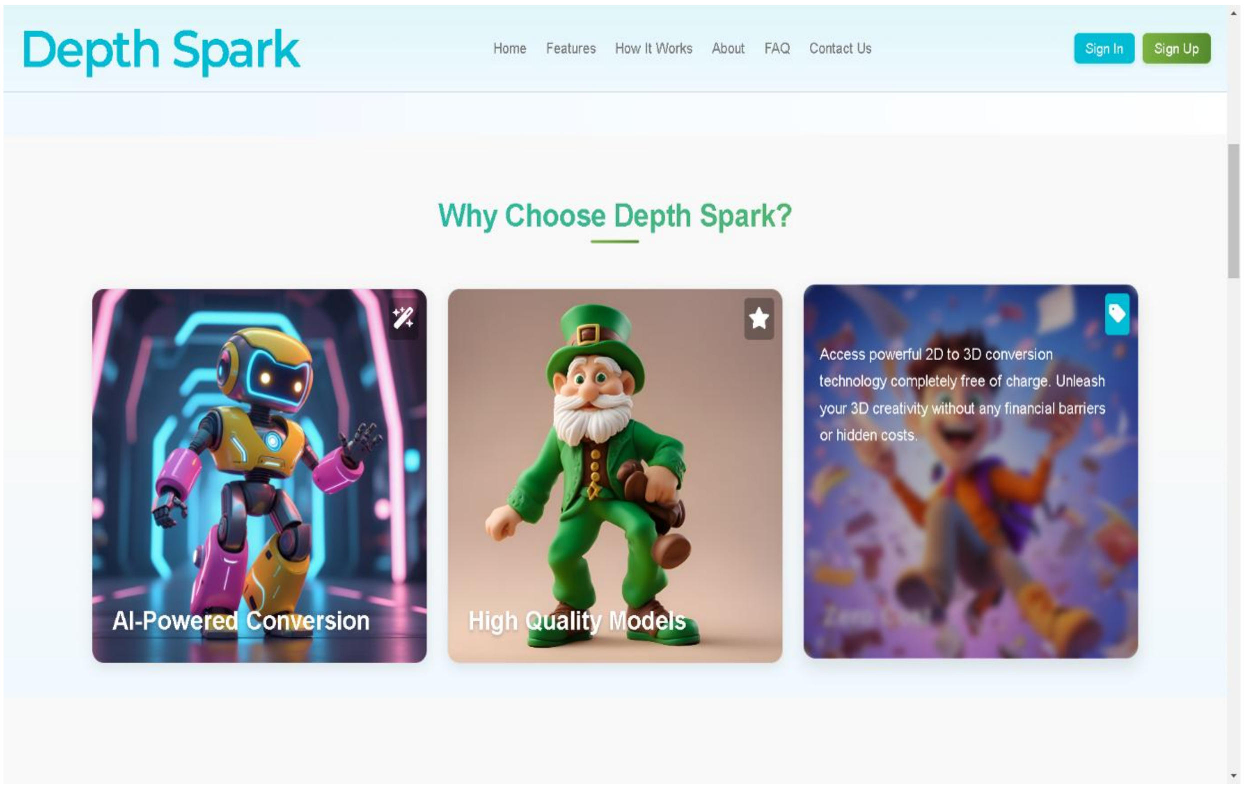
Moreover, during testing with real users, feedback indicated that the platform was easy to use, visually engaging, and delivered satisfying results without any need for external assistance or tutorials.

15.2 Website Interface Screenshots



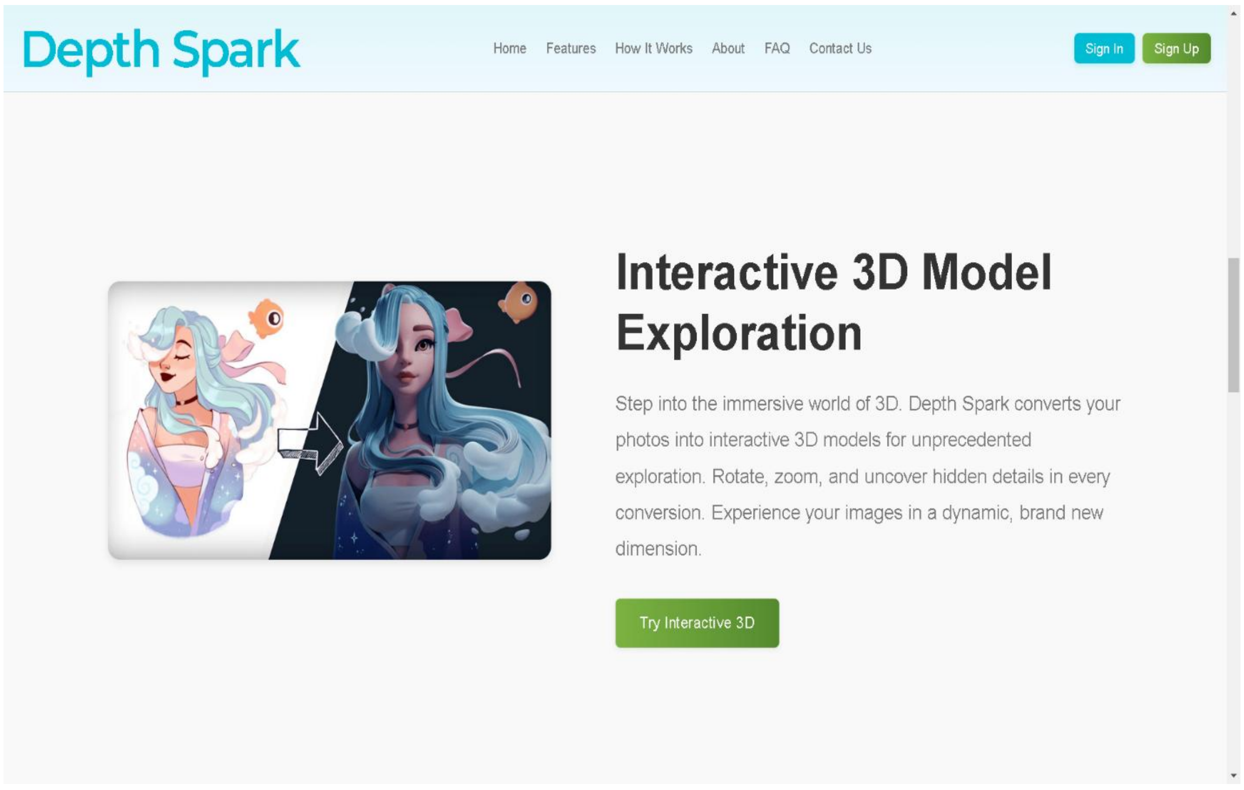
Transform Your 2D Images into Interactive 3D Models

Effortlessly create high-quality 3D models from your 2D images. Depth Spark empowers you to bring your visuals to life and engage your audience with interactive 3D experiences - all for free.



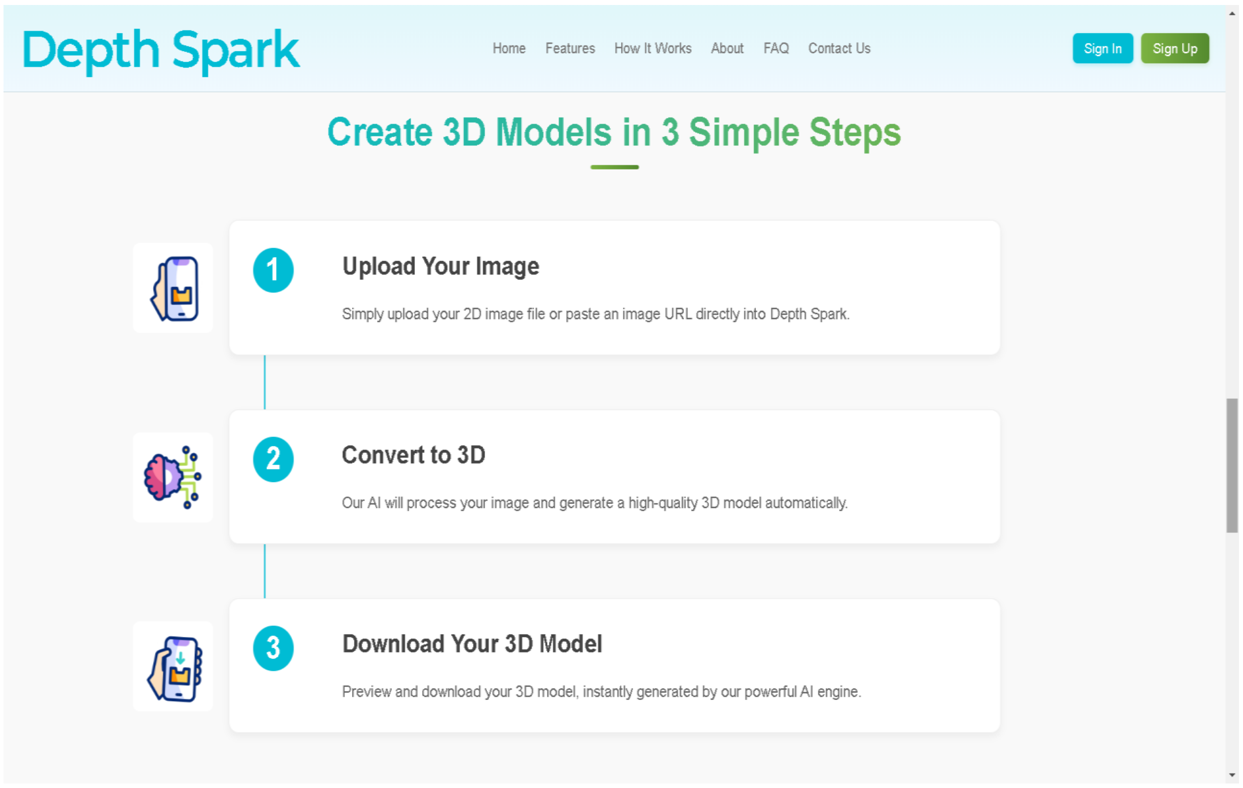
Why Choose Depth Spark?

Access powerful 2D to 3D conversion technology completely free of charge. Unleash your 3D creativity without any financial barriers or hidden costs.



Interactive 3D Model Exploration

Step into the immersive world of 3D. Depth Spark converts your photos into interactive 3D models for unprecedented exploration. Rotate, zoom, and uncover hidden details in every conversion. Experience your images in a dynamic, brand new dimension.



Create 3D Models in 3 Simple Steps

1 Upload Your Image

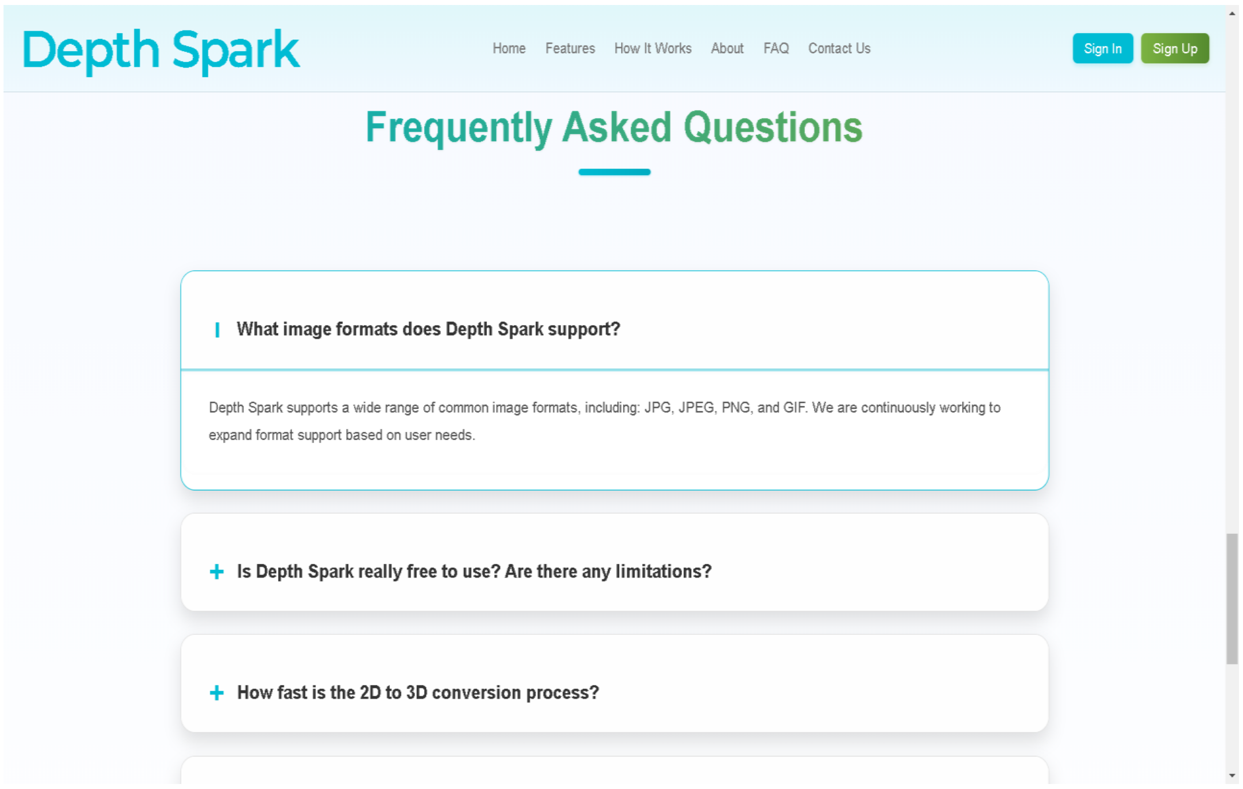
Simply upload your 2D image file or paste an image URL directly into Depth Spark.

2 Convert to 3D

Our AI will process your image and generate a high-quality 3D model automatically.

3 Download Your 3D Model

Preview and download your 3D model, instantly generated by our powerful AI engine.



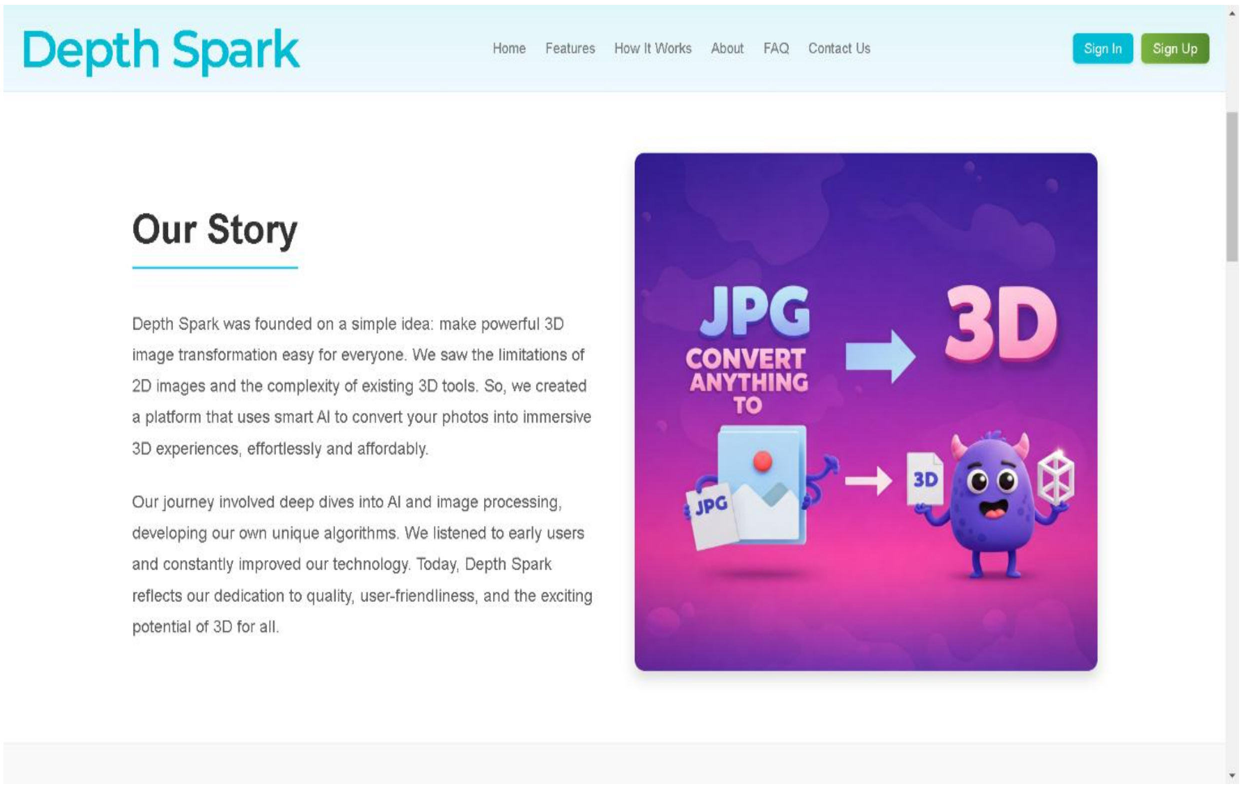
Frequently Asked Questions

What image formats does Depth Spark support?

Depth Spark supports a wide range of common image formats, including: JPG, JPEG, PNG, and GIF. We are continuously working to expand format support based on user needs.

+ Is Depth Spark really free to use? Are there any limitations?

+ How fast is the 2D to 3D conversion process?



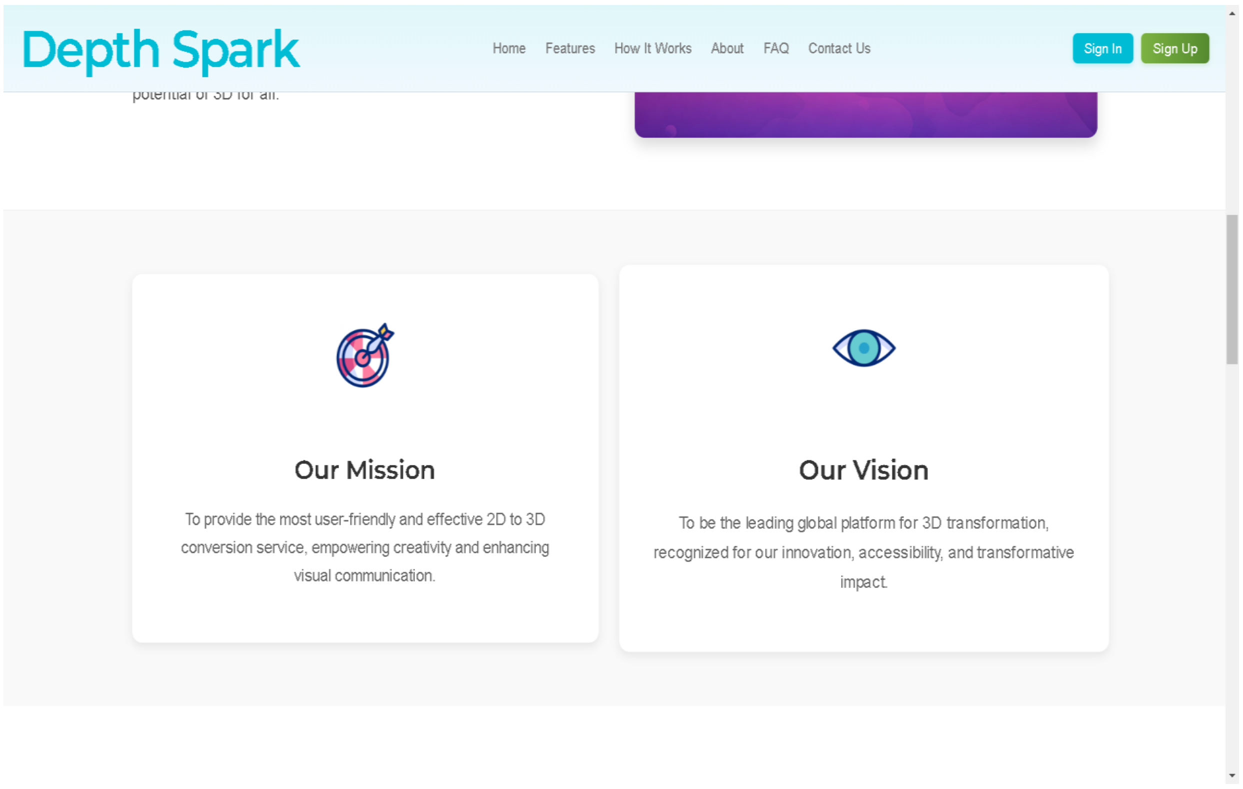
Depth Spark

Home Features How It Works About FAQ Contact Us Sign In Sign Up

Our Story

Depth Spark was founded on a simple idea: make powerful 3D image transformation easy for everyone. We saw the limitations of 2D images and the complexity of existing 3D tools. So, we created a platform that uses smart AI to convert your photos into immersive 3D experiences, effortlessly and affordably.

Our journey involved deep dives into AI and image processing, developing our own unique algorithms. We listened to early users and constantly improved our technology. Today, Depth Spark reflects our dedication to quality, user-friendliness, and the exciting potential of 3D for all.



Depth Spark

Home Features How It Works About FAQ Contact Us Sign In Sign Up

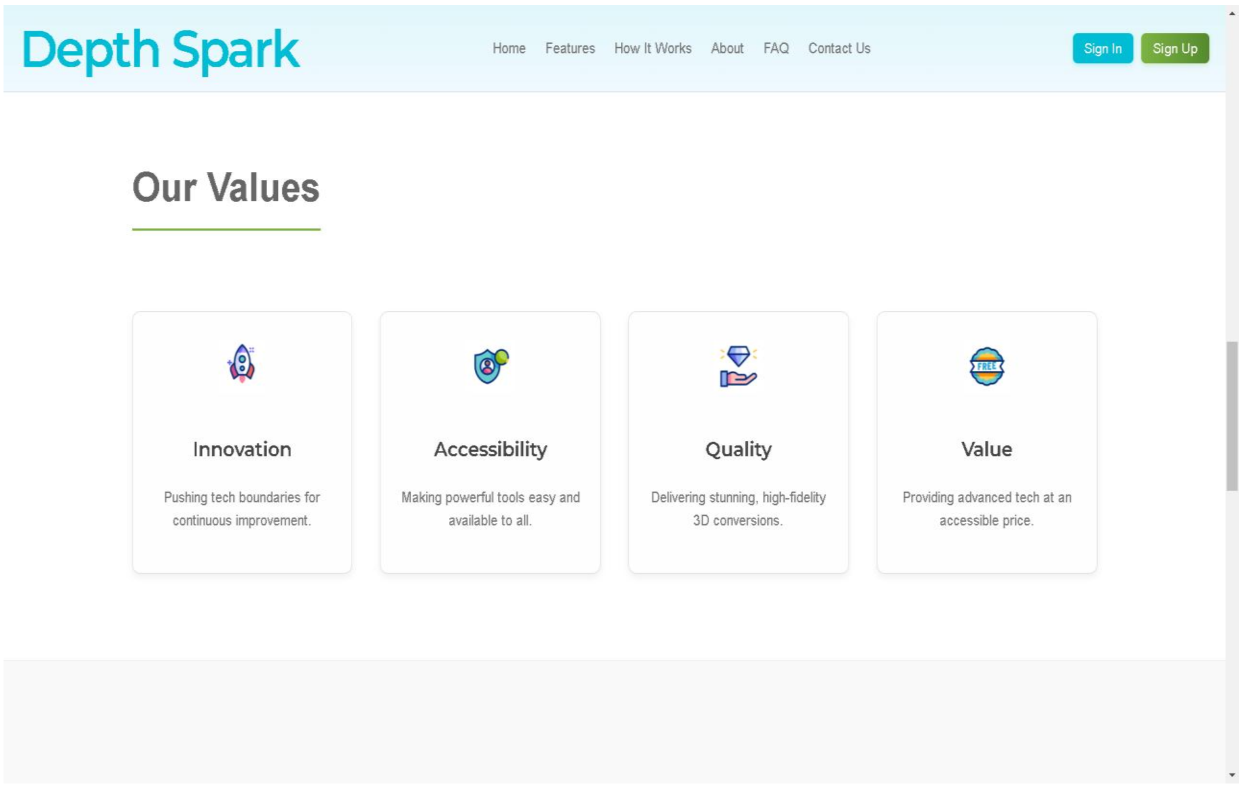
potential of 3D for all.

Our Mission

To provide the most user-friendly and effective 2D to 3D conversion service, empowering creativity and enhancing visual communication.

Our Vision

To be the leading global platform for 3D transformation, recognized for our innovation, accessibility, and transformative impact.



Our Values

Innovation

Pushing tech boundaries for continuous improvement.

Accessibility

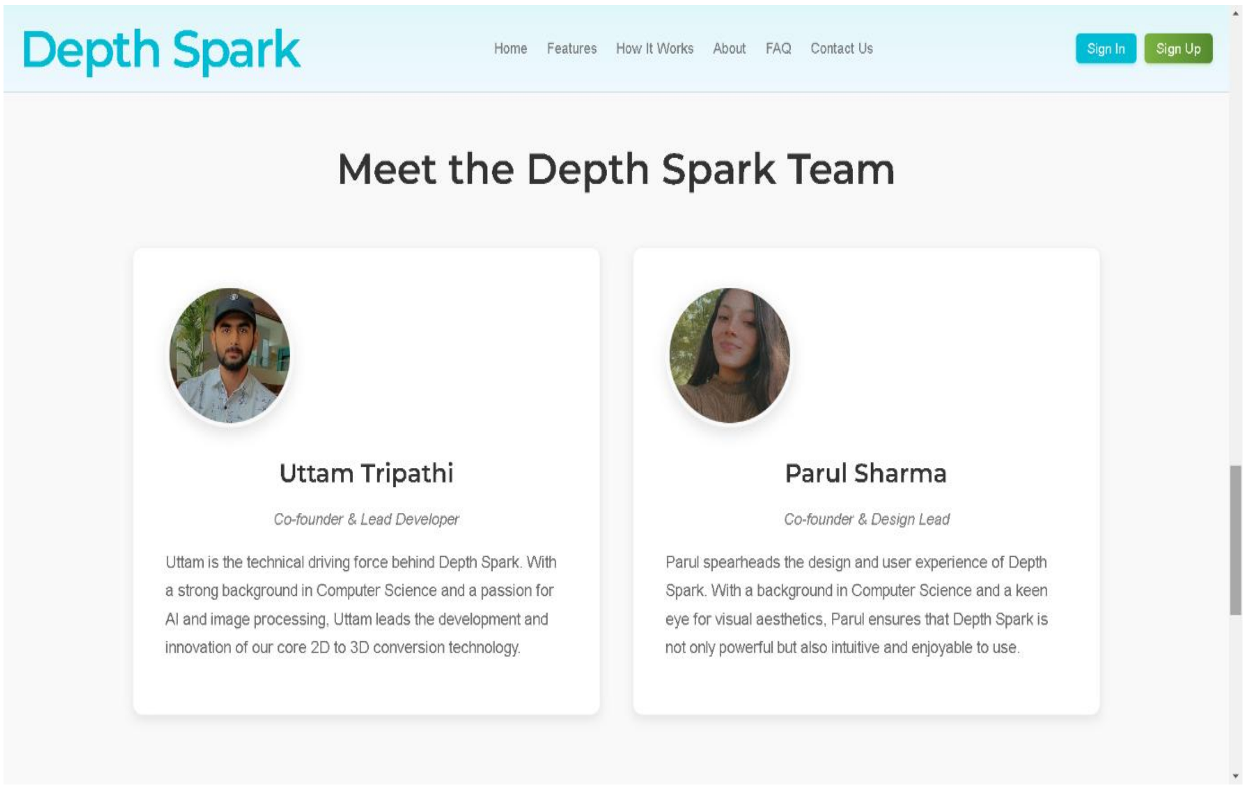
Making powerful tools easy and available to all.

Quality

Delivering stunning, high-fidelity 3D conversions.

Value

Providing advanced tech at an accessible price.



Meet the Depth Spark Team

Uttam Tripathi

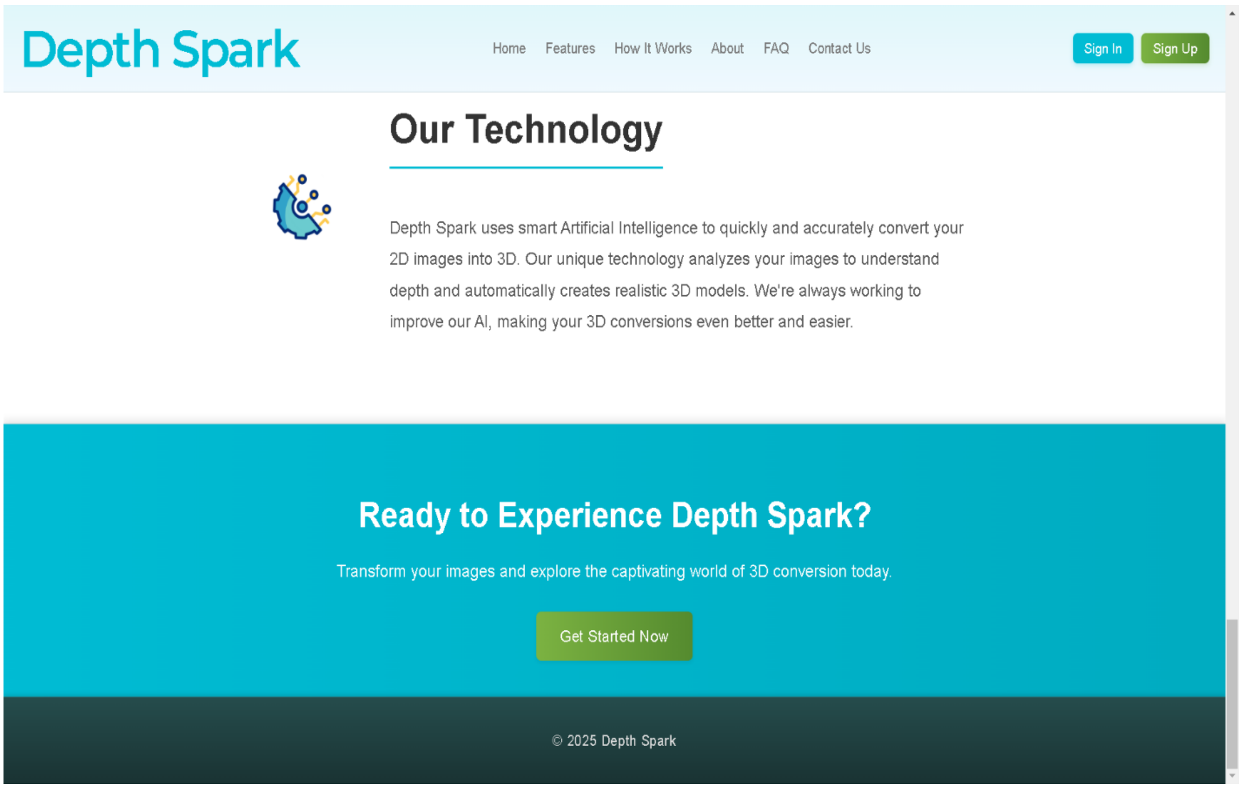
Co-founder & Lead Developer

Uttam is the technical driving force behind Depth Spark. With a strong background in Computer Science and a passion for AI and image processing, Uttam leads the development and innovation of our core 2D to 3D conversion technology.

Parul Sharma

Co-founder & Design Lead

Parul spearheads the design and user experience of Depth Spark. With a background in Computer Science and a keen eye for visual aesthetics, Parul ensures that Depth Spark is not only powerful but also intuitive and enjoyable to use.



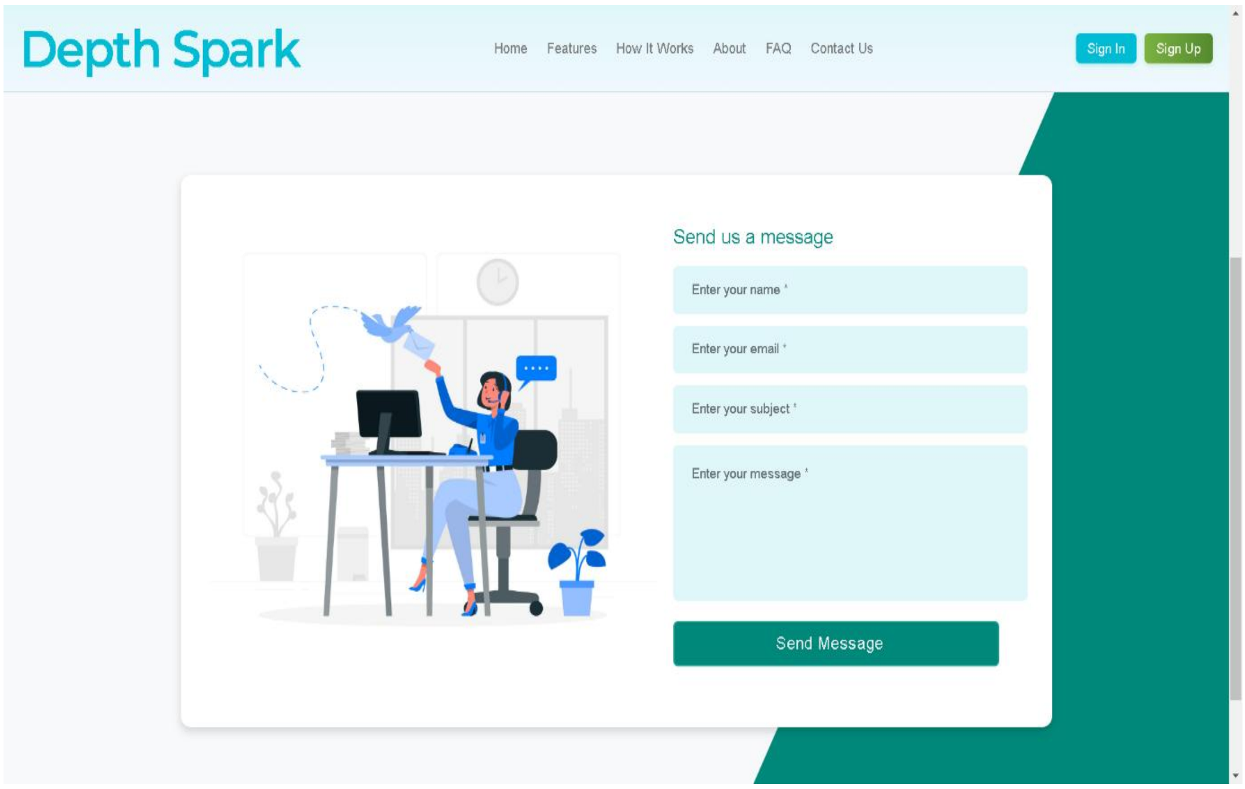
Our Technology

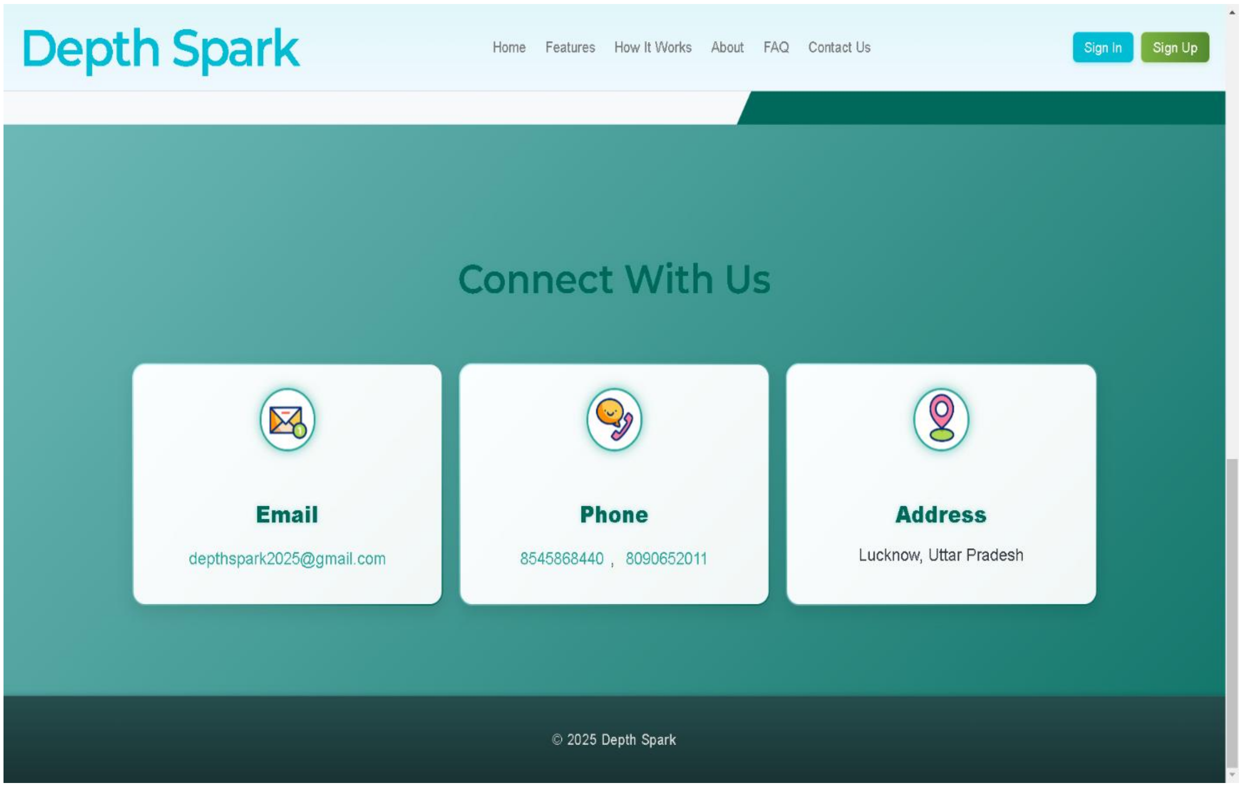
Depth Spark uses smart Artificial Intelligence to quickly and accurately convert your 2D images into 3D. Our unique technology analyzes your images to understand depth and automatically creates realistic 3D models. We're always working to improve our AI, making your 3D conversions even better and easier.

Ready to Experience Depth Spark?

Transform your images and explore the captivating world of 3D conversion today.

© 2025 Depth Spark





Connect With Us

Email

depthspark2025@gmail.com

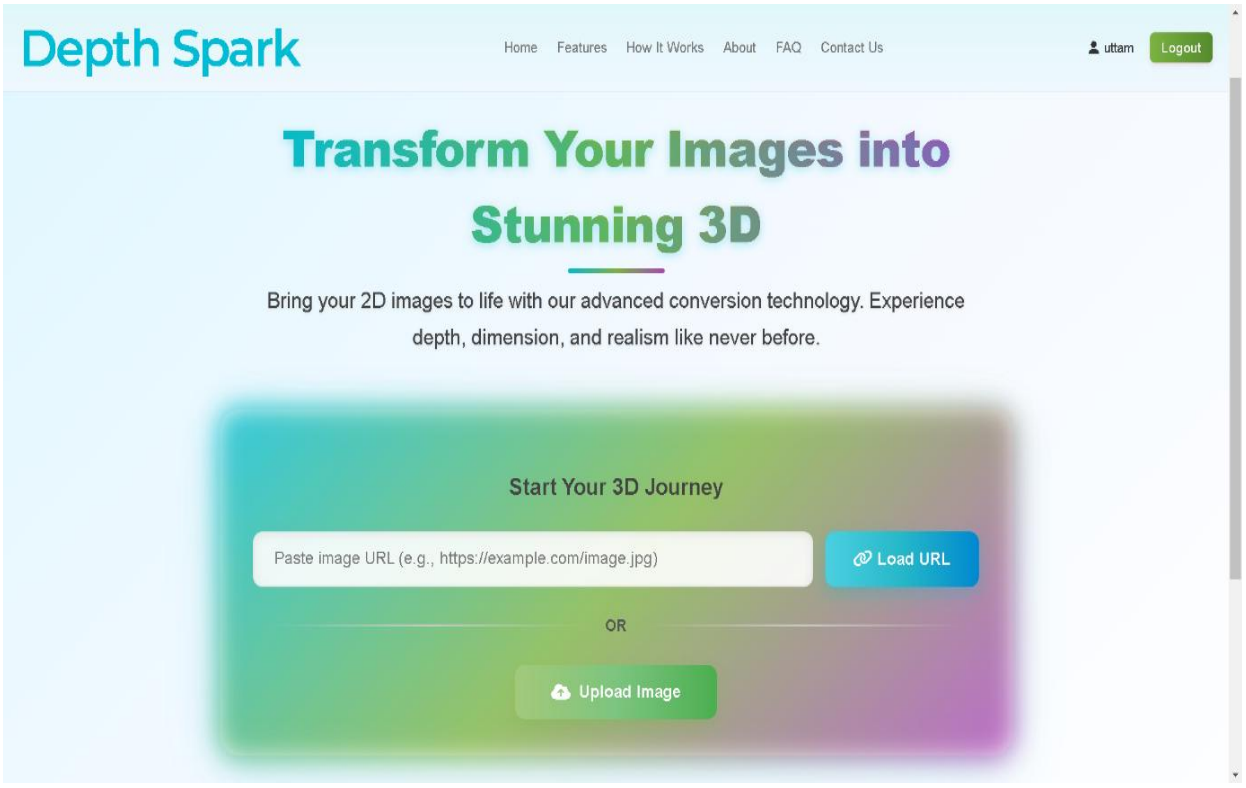
Phone

8545868440 , 8090652011

Address

Lucknow, Uttar Pradesh

© 2025 Depth Spark

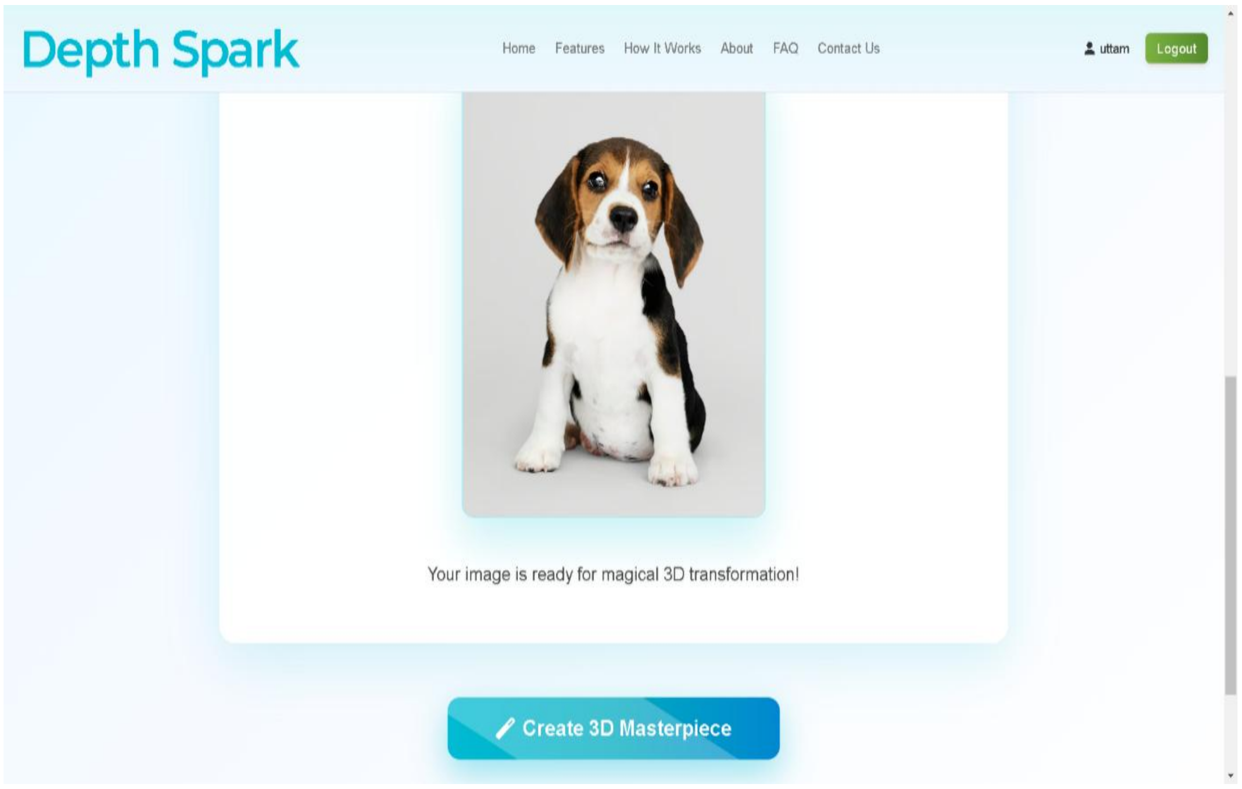


Transform Your Images into Stunning 3D

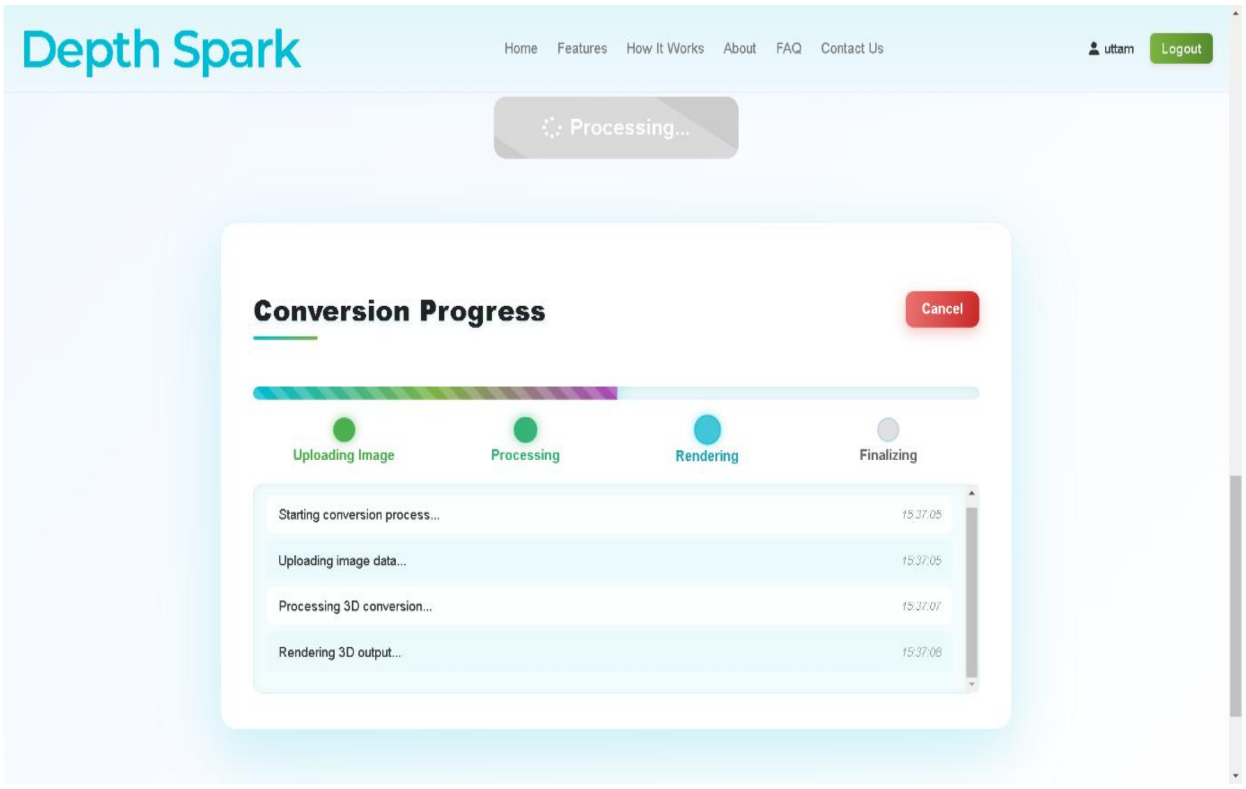
Bring your 2D images to life with our advanced conversion technology. Experience depth, dimension, and realism like never before.

Start Your 3D Journey

Paste image URL (e.g., https://example.com/image.jpg)

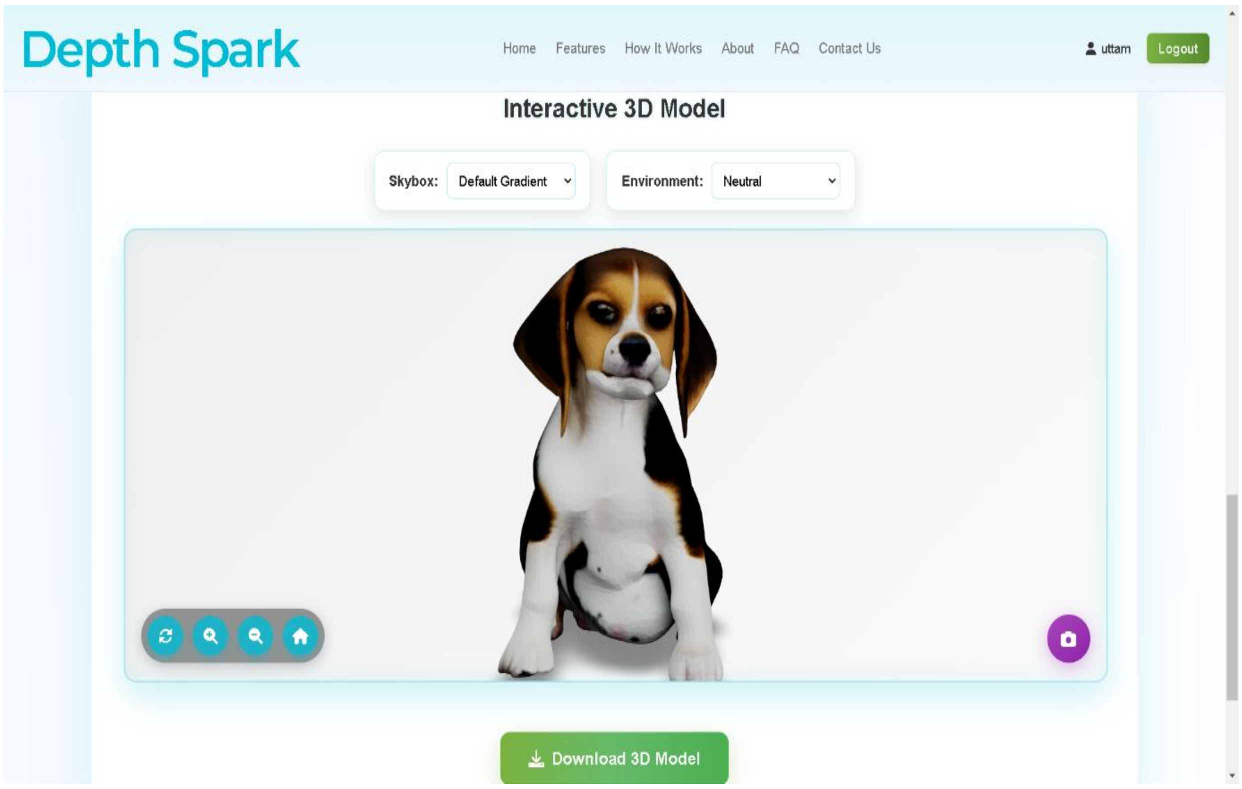


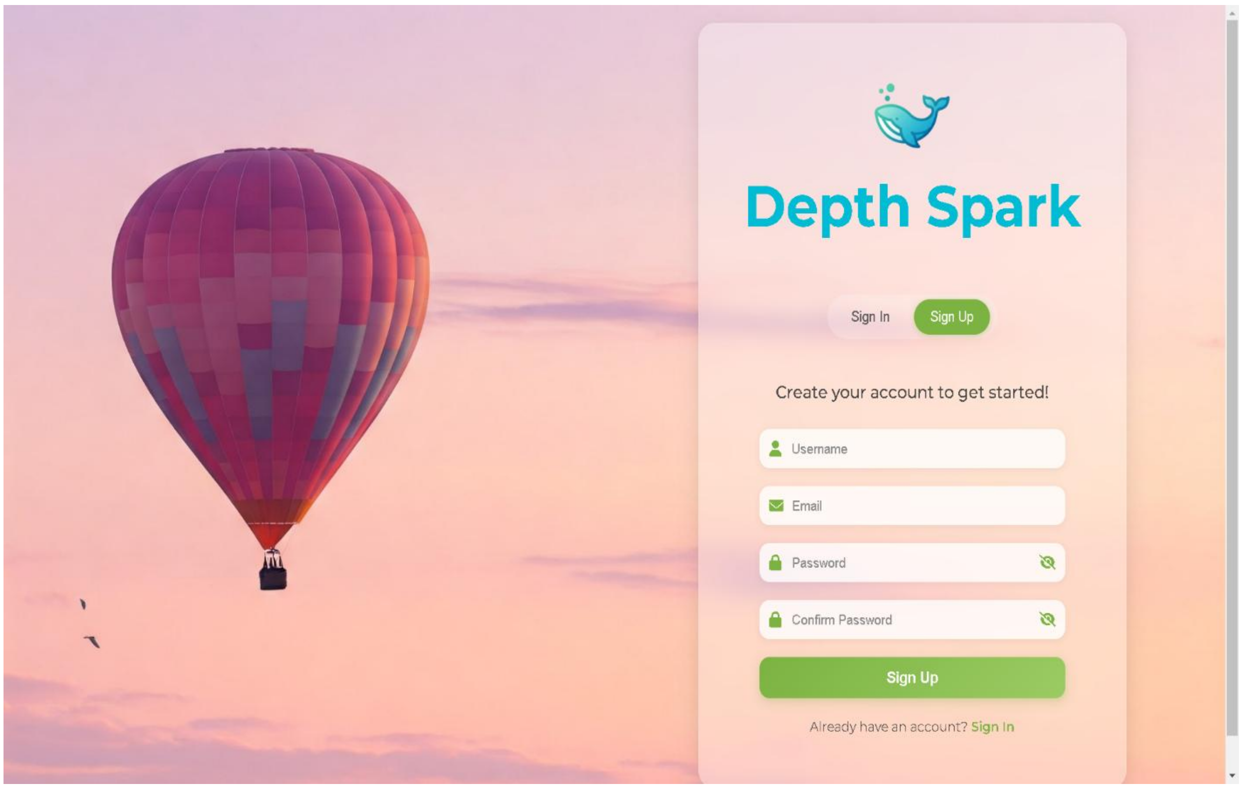
Your image is ready for magical 3D transformation!

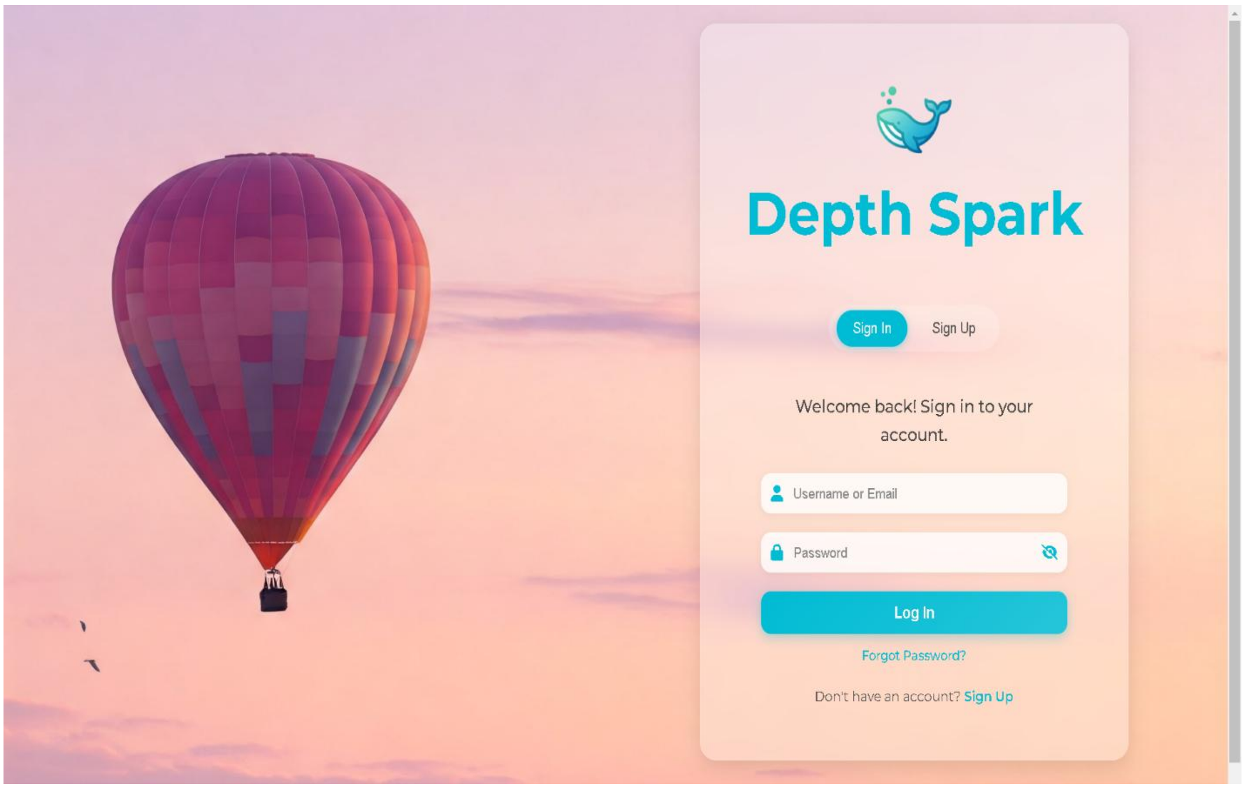


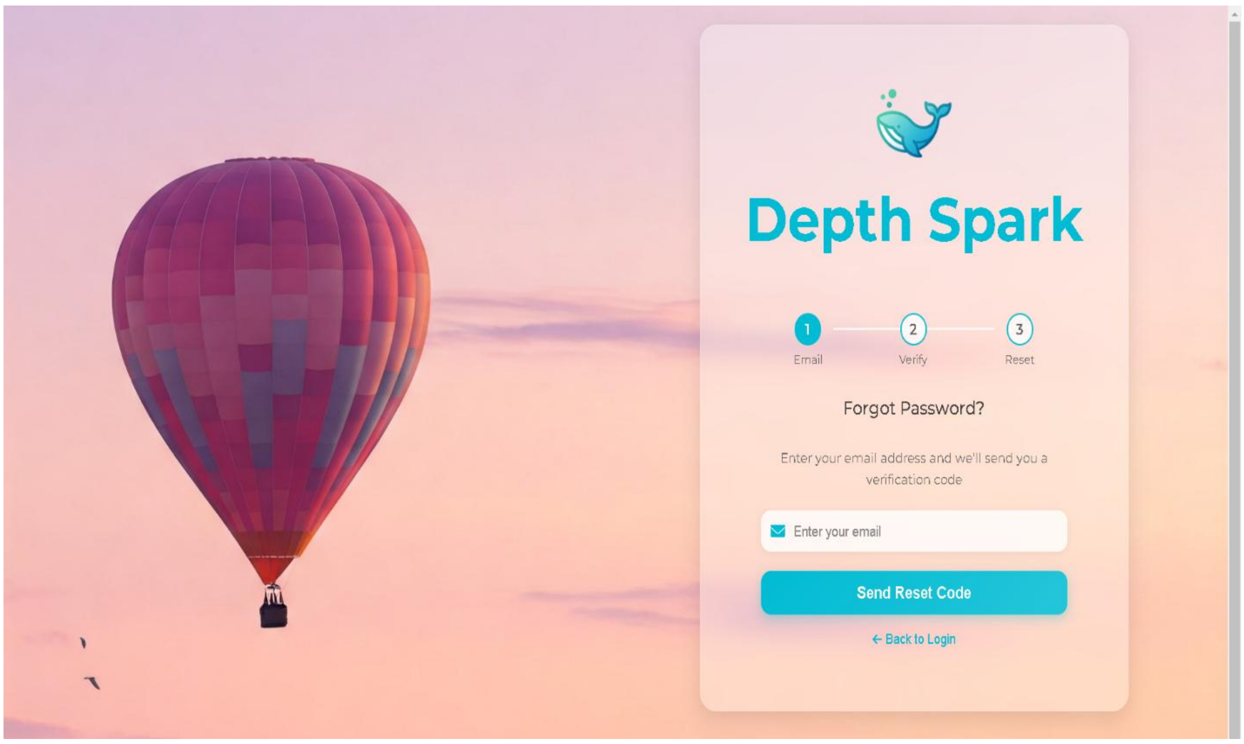
Conversion Progress

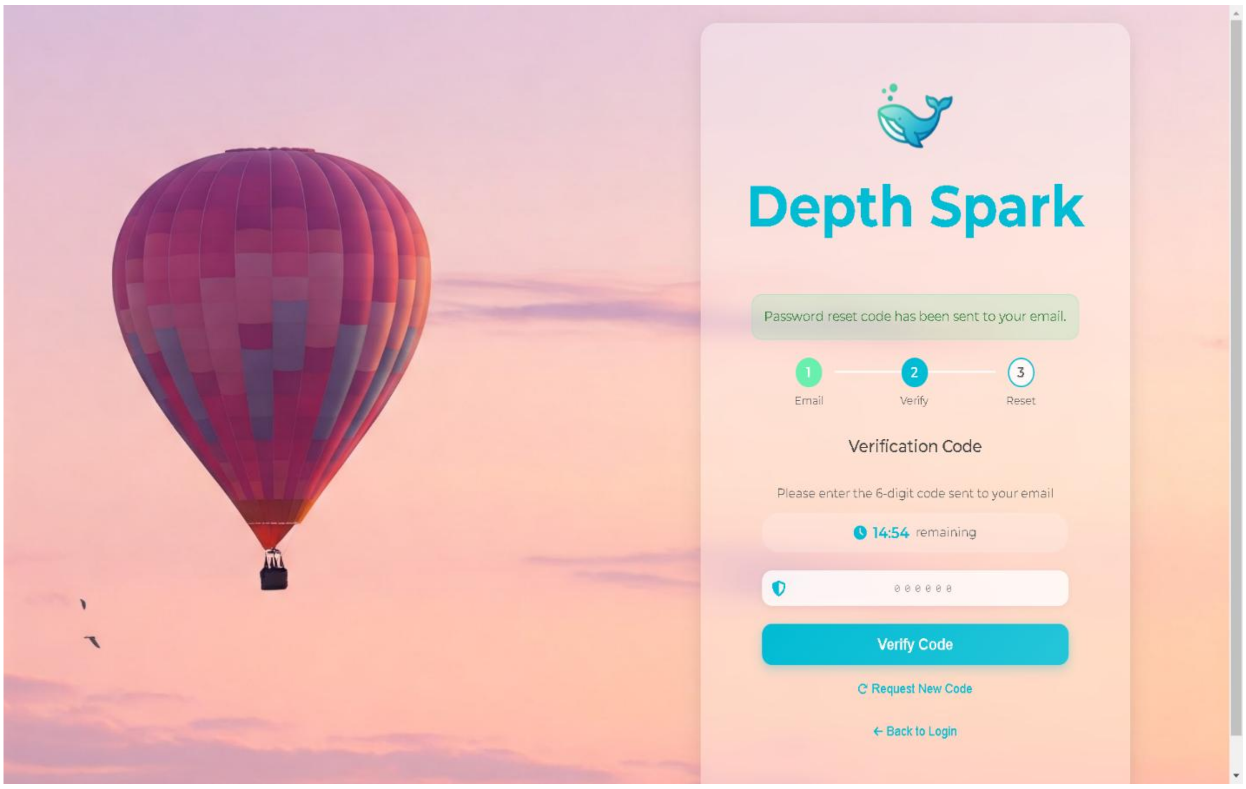
```
Starting conversion process...    15:37:05
Uploading image data...           15:37:05
Processing 3D conversion...       15:37:07
Rendering 3D output...            15:37:08
```


Depth Spark
Home
Features
How It Works
About
FAQ
Contact Us
uttam
Logout
Interactive 3D Model
Skybox: Default Gradient
Environment: Neutral
Download 3D Model


Depth Spark
Sign In
Sign Up
Create your account to get started!
Username
Email
Password
Confirm Password
Sign Up
Already have an account? Sign In


Depth Spark
Sign In
Sign Up
Welcome back! Sign in to your account.
Username or Email
Password
Log In
Forgot Password?
Don't have an account? Sign Up


Depth Spark
1
Email
2
Verify
3
Reset
Forgot Password?
Enter your email address and we'll send you a verification code
Enter your email
Send Reset Code
← Back to Login


Depth Spark

Password reset code has been sent to your email.

1 Email 2 Verify 3 Reset

Verification Code

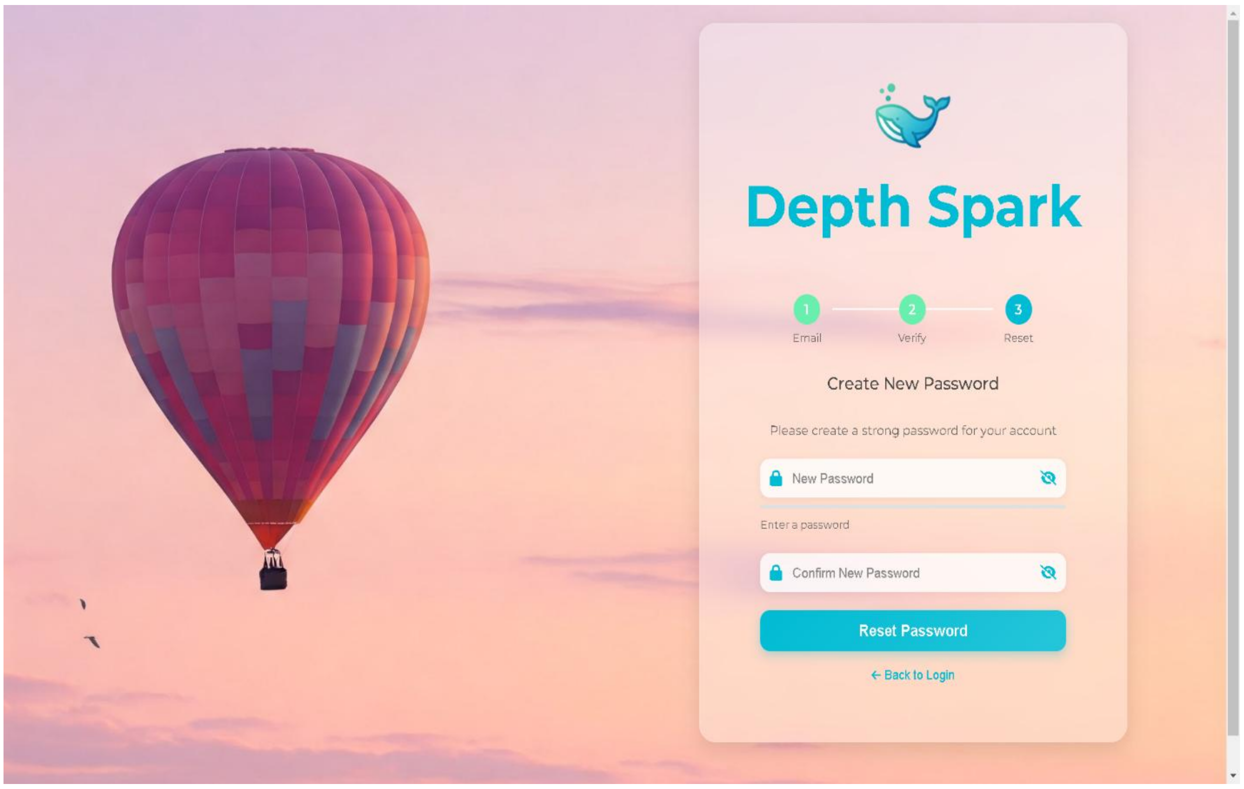
Please enter the 6-digit code sent to your email

14:54 remaining

Verify Code

Request New Code

Back to Login


Depth Spark

1 Email 2 Verify 3 Reset

Create New Password

Please create a strong password for your account

New Password

Enter a password

Confirm New Password

Reset Password

Back to Login

15.3 Code Implementation

app.py

```
from flask import Flask, render_template, url_for, request, redirect, flash, session, send_file,
jsonify, send_from_directory
from flask_sqlalchemy import SQLAlchemy
from flask_bcrypt import Bcrypt
from werkzeug.utils import secure_filename
import os
import requests
import uuid
from model import process_image_to_3d
import tempfile
import shutil
import magic  # For file type validation
from dotenv import load_dotenv
from flask_mail import Mail, Message
from datetime import datetime, timedelta
import random
import string
load_dotenv()

app = Flask(__name__)
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///database.db'
app.config['SECRET_KEY'] = 'secret_key'
app.config['UPLOAD_FOLDER'] = tempfile.gettempdir()
app.config['ALLOWED_EXTENSIONS'] = {'png', 'jpg', 'jpeg', 'gif'}
app.config['MODEL_FOLDER'] = os.path.join(app.static_folder, 'models')
app.config['MAIL_SERVER'] = 'smtp.gmail.com'
app.config['MAIL_PORT'] = 587
app.config['MAIL_USE_TLS'] = True
app.config['MAIL_USERNAME'] = os.getenv('MAIL_USERNAME')
app.config['MAIL_PASSWORD'] = os.getenv('MAIL_PASSWORD')

db = SQLAlchemy(app)
bcrypt = Bcrypt(app)
mail = Mail(app)

class User(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(50), nullable=False)
    email = db.Column(db.String(100), nullable=False)
    password = db.Column(db.String(100), nullable=False)
    reset_token = db.Column(db.String(6))
    reset_token_expiry = db.Column(db.DateTime)

    def __repr__(self):
        return 'User ' + str(self.id)

    def __init__(self, username, email, password):
        self.username = username
        self.email = email
        self.password = bcrypt.generate_password_hash(password).decode('utf-8')

    def check_password(self, password):
        return bcrypt.check_password_hash(self.password, password)

class Contact(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.String(100), nullable=False)
    email = db.Column(db.String(100), nullable=False)
    subject = db.Column(db.String(200), nullable=False)
    message = db.Column(db.Text, nullable=False)

    def __repr__(self):
        return f'Contact {self.id}'

with app.app_context():
    db.create_all()

def allowed_file(filename):
    return '.' in filename and filename.rsplit('.', 1)[1].lower() in
app.config['ALLOWED_EXTENSIONS']

def is_valid_gltf(file_path):
    """Check if the file is a valid GLTF/GLB by checking the magic number or extension."""
    try:
        mime = magic.Magic(mime=True)
        file_type = mime.from_file(file_path)
        return file_type in ['model/gltf-binary', 'application/octet-stream'] or
file_path.endswith('.glb')
    except Exception as e:
        print(f"Error validating GLTF file: {str(e)}")
        return False

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/features')
def features_section():
    return render_template('index.html')

@app.route('/how-it-works')
def how_it_works_section():
    return render_template('index.html')

@app.route('/about')
def about():
    return render_template('about.html')
```

```
@app.route('/faq')
def faq_section():
    return render_template('index.html')

@app.route('/contact-us')
def contact_us():
    return render_template('contact_us.html')

@app.route('/login', methods=['POST', 'GET'])
def login():
    if request.method == 'POST':
        email_or_username = request.form['email']
        password = request.form['password']
        user = User.query.filter_by(email=email_or_username).first() or
User.query.filter_by(username=email_or_username).first()

        if user and user.check_password(password):
            session['user'] = user.username
            flash('Login successful!', 'success')
            return redirect(url_for('index'))
        else:
            error = 'Invalid email or password'
            return render_template('login.html', error=error)
    return render_template('login.html')

@app.route('/signup', methods=['POST', 'GET'])
def signup():
    if request.method == 'POST':
        username = request.form['username']
        email = request.form['email']
        password = request.form['password']
        confirm_password = request.form['confirm_password']
        if password != confirm_password:
            return render_template('signup.html', message='Passwords do not match')
        user = User.query.filter_by(email=email).first() or
User.query.filter_by(username=username).first()
        if user:
            return render_template('signup.html', message='User already exists')
        new_user = User(username=username, email=email, password=password)
        db.session.add(new_user)
        db.session.commit()
        return redirect(url_for('login'))
    return render_template('signup.html')

@app.route('/convert', methods=['GET', 'POST'])
def convert():
    if 'user' not in session:
        return redirect(url_for('login'))
    if request.method == 'POST':
        image_url = request.form.get('image_url')
        file = request.files.get('file')
        if not image_url and not file:
            return jsonify({'error': 'Please provide an image URL or upload a file'}), 400

        try:
            if file and allowed_file(file.filename):
                filename = secure_filename(file.filename)
                temp_path = os.path.join(app.config['UPLOAD_FOLDER'], filename)
                file.save(temp_path)
                result = process_image_to_3d(temp_path, is_url=False)
                os.remove(temp_path)
            elif image_url:
                result = process_image_to_3d(image_url, is_url=True)
            else:
                return jsonify({'error': 'Invalid input'}), 400

            print('fal_client result:', result)  # Log the full result for debugging

            # Process the result for 3D model
            output = {}
            # Try multiple possible keys for the model, including nested model_mesh.url
            possible_keys = ['model_url', 'model', 'glb_url', 'output', 'file_url', 'gltf_url']
            model_url = None
            for key in possible_keys:
                model_url = result.get(key)
                if model_url and isinstance(model_url, str) and model_url.startswith(('http://',
'https://')):
                print(f"Found model URL in key '{key}': {model_url}")
                break
            # Check for nested model_mesh.url
            if not model_url and 'model_mesh' in result and isinstance(result['model_mesh'], dict):
                model_url = result['model_mesh'].get('url')
                if model_url and isinstance(model_url, str) and model_url.startswith(('http://',
'https://')):
                print(f"Found model URL in model_mesh.url: {model_url}")
                else:
                model_url = None

            if model_url:
                # Download the model to static/models
                model_filename = f"model_{uuid.uuid4()}.glb"
                model_path = os.path.join(app.config['MODEL_FOLDER'], model_filename)
                os.makedirs(app.config['MODEL_FOLDER'], exist_ok=True)

                try:
                response = requests.get(model_url, stream=True, timeout=10)
                if response.status_code == 200:
                    # Save to a temporary file first
                    temp_model_path = os.path.join(app.config['UPLOAD_FOLDER'],
f"temp_{model_filename}")
                    with open(temp_model_path, 'wb') as f:
                        for chunk in response.iter_content(chunk_size=8192):
                            if chunk:
                                f.write(chunk)

                    # Validate the file
                    if is_valid_gltf(temp_model_path):
                        shutil.move(temp_model_path, model_path)
                        output['model_url'] = url_for('static', filename=f'models/{model_filename}')
                        print(f"Successfully saved model to {model_path}")
                    else:
                        print(f"Invalid GLTF file downloaded from {model_url}")
                        os.remove(temp_model_path)
                        output['model_url'] = ''
                else:
                    print(f"Failed to download model from {model_url}: HTTP
{response.status_code}")
                    output['model_url'] = ''
                except Exception as e:
                print(f"Error downloading model from {model_url}: {str(e)}")
                output['model_url'] = ''
            else:
                # Check if result contains binary data or a file
                if isinstance(result, bytes) or (isinstance(result, dict) and 'data' in result and
isinstance(result['data'], bytes)):
                model_filename = f"model_{uuid.uuid4()}.glb"
                model_path = os.path.join(app.config['MODEL_FOLDER'], model_filename)
                os.makedirs(app.config['MODEL_FOLDER'], exist_ok=True)
                temp_model_path = os.path.join(app.config['UPLOAD_FOLDER'],
f"temp_{model_filename}")

                try:
                    if isinstance(result, bytes):
                        model_data = result
                    else:
                        model_data = result['data']

                    with open(temp_model_path, 'wb') as f:
                        f.write(model_data)

                    if is_valid_gltf(temp_model_path):
                        shutil.move(temp_model_path, model_path)
                        output['model_url'] = url_for('static', filename=f'models/{model_filename}')
                        print(f"Successfully saved model from binary data to {model_path}")
                    else:
                        print("Invalid GLTF data in fal_client result")
                        os.remove(temp_model_path)
                        output['model_url'] = ''
                except Exception as e:
                    print(f"Error saving model from binary data: {str(e)}")
                    output['model_url'] = ''
                else:
```

```
                print('No valid model URL or data found in fal_client result:', result)
                output['model_url'] = ''

            return jsonify({'success': True, 'result': output})

        except Exception as e:
            print(f"Error in /convert: {str(e)}")
            return jsonify({'error': str(e)}), 500

    return render_template('convert.html')

@app.route('/models/<path:filename>')
def serve_model(filename):
    return send_from_directory(app.config['MODEL_FOLDER'], filename)

@app.route('/logout')
def logout():
    session.pop('user', None)
    return redirect(url_for('index'))

@app.route('/submit-contact-form', methods=['POST'])
def submit_contact():
    if request.method == 'POST':
        name = request.form['name']
        email = request.form['email']
        subject = request.form['subject']
        message = request.form['message']

        try:
            contact = Contact(name=name,email=email, subject=subject, message=message)
            db.session.add(contact)
            db.session.commit()
            flash('Your message has been sent successfully!', 'success')
        except Exception as e:
            flash('An error occurred while sending your message.', 'error')
            db.session.rollback()

        return redirect(url_for('contact_us'))

@app.route('/forgot-password', methods=['GET', 'POST'])
def forgot_password():
    if request.method == 'POST':
        email = request.form['email']
        user = User.query.filter_by(email=email).first()

        if user:
            # Generate 6-digit OTP
            otp = ''.join(random.choices(string.digits, k=6))
            user.reset_token = otp
            user.reset_token_expiry = datetime.utcnow() + timedelta(minutes=15)
            # Send email with OTP
            msg = Message('Password Reset Request',
                         sender=app.config['MAIL_USERNAME'],
                         recipients=[email])
            msg.body = f'''To reset your password, use the following code:

{otp}

This code will expire in 15 minutes.

If you did not request a password reset, please ignore this email.
'''
            mail.send(msg)
            db.session.commit()

            flash('Password reset code has been sent to your email.', 'success')
            return render_template('verify_otp.html', email=email)
        else:
            flash('No account found with that email address.', 'error')

    return render_template('forgot_password.html')

@app.route('/verify-reset-token', methods=['POST'])
def verify_reset_token():
    email = request.form['email']
    otp = request.form['otp']

    user = User.query.filter_by(email=email).first()

    if not user:
        flash('Invalid email address.', 'error')
        return redirect(url_for('forgot_password'))

    if not user.reset_token or not user.reset_token_expiry:
        flash('No reset code was requested.', 'error')
        return redirect(url_for('forgot_password'))

    if datetime.utcnow() > user.reset_token_expiry:
        flash('Reset code has expired. Please request a new one.', 'error')
        return redirect(url_for('forgot_password'))

    if user.reset_token != otp:
        flash('Invalid reset code.', 'error')
        return render_template('verify_otp.html', email=email)

    return render_template('reset_password.html', email=email, token=otp)

@app.route('/reset-password', methods=['POST'])
def reset_password():
    email = request.form['email']
    token = request.form['token']
    password = request.form['password']
    confirm_password = request.form['confirm_password']

    if password != confirm_password:
        flash('Passwords do not match.', 'error')
        return render_template('reset_password.html', email=email, token=token)

    user = User.query.filter_by(email=email).first()

    if not user or not user.reset_token or user.reset_token != token:
        flash('Invalid reset request.', 'error')
        return redirect(url_for('forgot_password'))

    if datetime.utcnow() > user.reset_token_expiry:
        flash('Reset code has expired. Please request a new one.', 'error')
        return redirect(url_for('forgot_password'))

    # Update password
    user.password = bcrypt.generate_password_hash(password).decode('utf-8')
    user.reset_token = None
    user.reset_token_expiry = None
    db.session.commit()

    flash('Your password has been reset successfully. Please log in with your new password.',
'success')
    return redirect(url_for('login'))

if __name__ == '__main__':
    app.run(debug=True)
```

16. Conclusion

The development of Depth Spark marks an important achievement in democratizing access to 3D technology. By offering a free, easy-to-use, and AI-powered web application, Depth Spark bridges the gap between simple 2D imagery and the rapidly expanding world of 3D content creation. The project successfully addresses the initial problem statement: the high cost, technical complexity, and limited accessibility of professional 3D model generation tools for individuals, educators, small businesses, and creative professionals.

Throughout the project lifecycle, from problem analysis and system design to implementation and testing, each phase was carefully executed with a clear focus on performance, usability, security, and scalability. The chosen technology stack — consisting of Python, Flask, SQLAlchemy, HTML/CSS/JavaScript, Model Viewer, and the Falcon AI API — proved to be effective in achieving the functional and non-functional requirements of the system. The modular architecture, comprising well-defined components such as the User Interface Module, Image Handler, AI-Based 3D Generator, 3D Model Exporter, Viewer, Queue Manager, and Authentication System, ensured that development remained organized, manageable, and future-ready.

The final system output met all critical objectives. Users can effortlessly upload images, track their progress in real-time, view AI-generated 3D models interactively, and download high-quality models for a variety of applications. Performance testing validated the platform's ability to handle concurrent users efficiently, while security testing ensured that user data and external API communications remain fully protected.

One of the standout achievements of Depth Spark is its ability to maintain a highly responsive user experience without requiring expensive hardware or advanced technical skills. This accessibility opens new opportunities for different fields, including e-commerce (for virtual product displays), education (for interactive learning materials), marketing (for enhanced digital campaigns), and entertainment (for 3D asset generation).

However, the project also revealed new possibilities for future enhancement. With batch uploads, cloud storage, advanced model editing tools, AR/VR integration, and API-as-a-service models on the horizon, Depth Spark has the potential to evolve even further. By continuously embracing technological innovations and user feedback, the platform can establish itself as a major player in the domain of free and accessible 3D content creation.

In conclusion, Depth Spark is more than just a software project — it is a step toward making powerful 3D technologies accessible to a wider audience, removing the barriers of cost and complexity. The project lays a strong foundation for future expansion and provides a meaningful contribution to the growing ecosystem of open digital tools. With its solid base, clear vision, and continued commitment to innovation, Depth Spark is well-positioned to grow, adapt, and impact how users create and interact with 3D content in the coming years.

16.1 Summary and Reflections

The Depth Spark project has been a comprehensive and insightful journey, covering the complete software development lifecycle — from the initial identification of a real-world problem to the final, user-centered solution. The project successfully delivered a fully operational 2D-to-3D model conversion web application that fulfills the original objectives of simplicity, accessibility, security, and free availability.

Through the systematic approach taken during the analysis, design, development, and testing phases, Depth Spark emerged as a strong proof of concept demonstrating that advanced technologies like AI-based 3D modeling can be made accessible without requiring extensive resources or specialized knowledge. Each module of the system — including the User Interface, Image Handler, AI-Based 3D Generator, 3D Viewer, Queue Manager, and Authentication System — was carefully crafted to perform its specific role effectively while maintaining smooth integration with the overall architecture.

Reflecting on the project's progression, several key insights and lessons have emerged. One of the most important realizations was the critical role of modular design in managing complexity. By breaking down the system into distinct modules with clear responsibilities, it became easier to develop, debug, and maintain the platform. Another important learning was the significance of early testing and continuous integration. By regularly testing individual components and their interactions, the development team was able to catch and resolve issues early, saving significant time during the final stages.

The project also highlighted the importance of performance optimization and user-centric design. Regular load testing and performance monitoring ensured that Depth Spark remained fast and stable, while feedback-driven UI improvements made the platform intuitive and easy to navigate for users of all backgrounds.

Additionally, working on Depth Spark emphasized the need for strong security practices even in seemingly simple web applications. Implementing secure authentication, session management, and encrypted API communication proved vital in building user trust and ensuring the integrity of the platform.

From a personal and team development perspective, this project offered valuable hands-on experience with real-world tools such as Flask, SQLAlchemy, Model Viewer, and cloud services. It reinforced the practical importance of collaboration, time management, documentation, and adaptability during technical problem-solving.

Overall, Depth Spark stands as a successful implementation of both technical skills and project management principles. It also serves as a strong foundation for future enhancements, where the platform can expand its features, integrate with AR/VR environments, and offer even greater value to users worldwide.

17. Future Enhancements

Depth Spark successfully achieves its primary goal of providing a free, accessible, and simple 2D-to-3D model conversion platform. However, as technology evolves and user expectations grow, there are several opportunities to expand the platform's features and improve its applications even further. This chapter explores potential future enhancements for Depth Spark that could significantly extend its functionality, performance, and reach.

Building on its current strengths, Depth Spark can evolve into a more comprehensive 3D content creation and management tool by introducing new features, enhancing existing modules, and exploring integration with emerging technologies like Augmented Reality (AR), Virtual Reality (VR), and the Metaverse.

17.1 Feature Expansion and Applications

There are several planned and possible future enhancements for Depth Spark that would increase its value to users and open up new application areas. Some of the key areas for feature expansion and broader application are discussed below.

One of the most immediate enhancements would be the addition of Batch Uploading and Processing. Currently, users can upload and convert one image at a time. In the future, a batch processing feature could allow multiple images to be uploaded simultaneously, processed in parallel, and generated as multiple 3D models in one session. This would greatly improve efficiency for users working with large datasets, such as e-commerce platforms with extensive product catalogs.

Another important upgrade would be the development of Advanced Customization Tools for the generated 3D models. Users could be given the ability to adjust model details after AI generation, such as:

- Fine-tuning the depth map manually.
- Changing colors, textures, or surface finishes.

Customization would not only improve user satisfaction but also expand the creative possibilities for marketing, education, and design users.

Depth Spark also has strong potential for Augmented Reality (AR) and Virtual Reality (VR) Integration. By allowing users to export their models in AR-ready formats or providing direct AR previews within the web app, users could place their 3D objects in real-world environments through smartphones or AR headsets. Similarly, VR integration could allow full-scale model viewing and interaction inside virtual spaces, broadening the platform’s relevance for architecture, gaming, and immersive education.

Cloud Storage and User Model Libraries are another planned area of development. In the current system, users download models directly after generation. A future version could introduce cloud-based accounts where users can save their past uploads and models, organize them into folders, and access them anytime from different devices.
This feature would make Depth Spark more useful for regular users who work with multiple models over time.

Expanding Supported Input and Output Formats is also under consideration. While the current system supports common 2D image formats and outputs GLB files, future enhancements could allow:

- Input from other types of media such as short video clips or depth sensor images.
- Output to other 3D file formats like OBJ, FBX, and USDZ to support a wider range of platforms.

To further support commercial applications, API-as-a-Service could be introduced. This would allow businesses and developers to integrate Depth Spark's 2D-to-3D model generation functionality directly into their own apps, websites, or workflows through a secure API, creating possibilities for partnerships and monetization.

Additionally, improvements in AI Model Accuracy and Speed would always remain a priority. As AI research progresses, integrating more advanced machine learning models could reduce generation times even further while improving the realism and detail of 3D outputs.

Finally, enhancing Accessibility Features — such as voice-guided navigation, multi-language support, and simplified mobile-first interfaces — would allow Depth Spark to reach even broader audiences globally.

Conclusion
Depth Spark, while already a strong and functional system, has significant room for future growth and enhancement. By implementing batch processing, model customization, AR/VR integration, cloud storage, expanded file format support, commercial APIs, and continuous AI improvements, Depth Spark can evolve into a full-fledged 3D content platform capable of serving a wide range of users across industries.

Bibliography

1. Bhattacharjee, S. (n.d.). Computer graphics. Indian Institute of Technology Guwahati. Retrieved from https://iitg.ac.in/samit/Computer%20Graphics.pdf

2. Flask Documentation. (2024). Flask web development, one drop at a time. Retrieved from https://flask.palletsprojects.com

3. Falcon AI Platform. (2024). 2D to 3D model conversion API services. Retrieved from https://fal.ai/

4. Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT Press. Retrieved from https://www.deeplearningbook.org/

5. Khronos Group. (2024). glTF 2.0 specification. Retrieved from https://www.khronos.org/gltf/

6. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems, 25, 1097–1105. Retrieved from https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks

7. Let's Encrypt. (2024). Free SSL/TLS certificates. Retrieved from https://letsencrypt.org/

8. Marcotte, E. (2011). Responsive web design. A Book Apart. Retrieved from https://abookapart.com/products/responsive-web-design

9. Model Viewer Documentation. (2024). Retrieved from https://modelviewer.dev

10. Python Official Documentation. (2024). Python programming language resources. Retrieved from https://docs.python.org/3/

11. SQLAlchemy Documentation. (· 2024). The database toolkit for Python. Retrieved from https://www.sqlalchemy.org/

12. W3C Web Standards. (2024). World Wide Web Consortium - Web technologies. Retrieved from https://www.w3.org/