VIRTUAL 3D TOUR

Project Team

Dabeer Ul Haq Qureshi	21I-1518
Senan Faisal	21I-1546
Saad Ahsan	21I-2965

Session 2024-2025

Supervised by

Mr Zeshan Khan

Department of Computer Science

National University of Computer and Emerging Sciences
Islamabad, Pakistan

June, 2025

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

The tourism industry has witnessed a significant transformation with the integration of digital technologies that enhance user engagement and accessibility. One of the most impactful developments in this regard is the use of Virtual Reality (VR) and 3D visualizations to create immersive experiences that allow users to explore locations remotely. However, many tourist regions—especially in countries like Pakistan—remain underrepresented in digital spaces due to limited resources, poor accessibility, and a lack of technological integration.

Pakistan's northern areas, including regions like Hunza, Skardu, and Swat, are home to some of the world's most breathtaking landscapes, yet remain largely unexplored by both domestic and international tourists. The lack of comprehensive visual content and accessibility has created a significant gap in promoting these locations to a broader audience. At the same time, technological advancements in computer vision, machine learning, and neural rendering techniques like NeRF (Neural Radiance Fields) and Gaussian Splatting have opened new avenues for converting simple 2D images into rich 3D scenes.

This project—Virtual 3D Tour—addresses this gap by leveraging modern technologies to build a system capable of transforming tourist images into interactive, photorealistic 3D models. The aim is to create a user-friendly digital platform where users can explore iconic destinations virtually, using only an image, place name, or descriptive text prompt.

1.2 Motivation

Traditional methods of virtual tourism often require expensive equipment or manually constructed 3D scenes, making them inaccessible or unsustainable for large-scale implementation. Furthermore, tourists often have limited information or visual access to many remote areas before visiting, which can affect their travel decisions.

Our project is motivated by the idea of democratizing access to Pakistan's hidden scenic locations through a scalable and intelligent system that:

- Requires only a simple 2D image, place name, or text description as input.
- Uses deep learning models (e.g., ResNet) for image embedding and language models for understanding textual prompts.
- Generates and retrieves interactive 3D tours using Gaussian Splatting for high-quality rendering.

We aim to reduce the gap between a user's curiosity and their ability to visualize a location in 3D, enhancing both the tourism planning experience and cultural awareness.

1.3 Related Work and Technological Landscape

Several commercial and academic efforts have attempted to enhance virtual tourism. Google Street View offers panoramic photography of many areas, but it lacks interactivity and detailed 3D models. Projects using Structure from Motion (SfM) or photogrammetry often require multiple overlapping images and fail in low-texture or inconsistent lighting conditions.

Recent research in Neural Radiance Fields (NeRF) and Gaussian Splatting has revolutionized the field by enabling high-fidelity scene reconstruction from sparse views. Similarly, embedding-based retrieval using Convolutional Neural Networks (CNNs) and Sentence Transformers has enabled efficient matching between input queries and relevant media.

However, to our knowledge, there is no integrated system that:

- Accepts multi-modal inputs (image, place name, or prompt),
- Retrieves or generates 3D models in real-time,
- And presents them in an accessible mobile/web interface tailored for tourism use cases.

1.4 Objectives

The key objectives of this project are:

- To build a system that generates or retrieves 3D virtual scenes from 2D images, place names, or text prompts.
- To develop a web/mobile application that allows users to interact with these 3D models (zoom, rotate, pan).
- To construct a searchable embedding database of images and associated 3D models.
- To apply ResNet-based CNNs and text embedding models for effective image/text matching.
- To render immersive 3D scenes using Gaussian Splatting, offering high visual fidelity and interactivity.

1.5 Existing Solutions

Table 1.1: Comparison of Existing Solutions

System Name	System Overview	System Limitations
Google Street View	Provides panoramic street-level imagery allowing users to virtually navigate areas.	Lacks immersive 3D models; no user-uploaded image-based generation or interactivity.
Photogrammetry-Based Apps	Reconstructs 3D models from multiple overlapping images using SfM techniques.	Requires dense and consistent datasets; fails in low-light or texture-less environments.
Sketchfab/RealityCapture	Offers hosting and viewing of user-generated 3D models; supports some interactivity.	High-quality models require expensive capture setups; lacks multimodal input support.

1.6 Scope

The scope of this project includes:

- Collecting and processing 2D images of selected tourist destinations in Pakistan.
- Training deep learning models (CNNs for images, language models for text) to create embeddings.
- Creating a vector database linking embeddings with 3D model identifiers.

- Using Gaussian Splatting techniques to generate or retrieve 3D models for immersive visualization.

- Developing a responsive mobile/web application with features like 3D interaction, search, and multilingual support.

The project excludes the development of VR headset-based functionality, real-time photogrammetry pipelines, and integration with external mapping services (e.g., Google Maps). It also does not cover real-time user collaboration features or post-deployment content moderation systems.

1.7 Modules

1.7.1 Module 1: Data Acquisition and Processing

This module handles the collection and cleaning of images from selected tourist locations.

1. High-quality image collection.

2. Preprocessing and resizing for model readiness.

1.7.2 Module 2: Embedding and Retrieval

Responsible for converting inputs (images or text prompts) into embeddings and matching them with the database.

1. Image embedding using ResNet.

2. Text embedding using a Sentence Transformer or similar model.

3. Vector similarity matching using FAISS.

1.7.3 Module 3: 3D Model Generation and Storage

This module generates or retrieves the 3D model associated with the matched class.

1. Gaussian Splatting for realistic rendering.

2. Storage of models in standardized formats.

1.7.4 Module 4: User Interface (Web/Mobile)

Frontend interface for input, display, and interaction.

1. Input options: image upload, text prompt, or location selection.
2. 3D model viewer with zoom, pan, rotate.

1.7.5 Module 5: Information Display and Feedback

Displays contextual information and collects user feedback.

1. Description generation using LLM.
2. User ratings and feedback storage.

Chapter 2

Project Requirements

This section outlines the necessary requirements for the successful completion of the Virtual 3D Tour application. Project requirements are divided into two main categories: functional requirements, which describe the core operations of the system, and non-functional requirements, which specify performance, security, and usability standards.

2.1 Use-case/Event Response Table/Storyboarding

This section describes how users will interact with the Virtual 3D Tour application through various scenarios, including use cases and storyboarding. Each use case presents a specific interaction between the user and the system, outlining steps and expected responses. Key components include:

Use Case ID and Title

Actors (e.g., tourists, students, or general users)

Preconditions (e.g., internet access, app installation)

Main Flow of Events (e.g., user enters place name $\rightarrow$ 3D model is generated)

Postconditions (e.g., interactive 3D model displayed)

Exceptions (e.g., invalid input or failed model generation)

Storyboarding is used to visually depict these user interactions and ensure that the design supports a seamless and intuitive experience. For instance, a storyboard can show the step-by-step flow of a user entering a destination and exploring the generated 3D environment interactively.

Depending on the project phase and context, use cases, storyboards, or event-response tables may be used to effectively represent user-system interaction. Examples are provided in Appendix A.

2.2 Functional Requirements

The functional requirements describe specific behaviors and functions of the Virtual 3D Tour system. These are organized by key modules of the application.

2.2.1 Module 1: Input and Retrieval

This module handles user input and retrieval of relevant 3D data.

1. FR1: The system shall allow users to input a place name, an image, or a descriptive sentence.
2. FR2: The system shall retrieve a corresponding 3D model based on the user input using semantic search or image embeddings.
3. FR3: The system shall connect to a Firebase database to check for existing 3D models associated with user input.

2.2.2 Module 2: 3D Model Generation and Display

This module is responsible for model generation, rendering, and interaction.

1. FR1: If no matching 3D model exists, the system shall generate a 3D model using NerfStudio and Instant-NGP from the uploaded 360-degree images.
2. FR2: The system shall convert the generated 3D model into .ply or .obj format for rendering.
3. FR3: The system shall display the interactive 3D model on the app's 3D output screen using a viewer compatible with the selected format.

2.2.3 Module 3: User Interface and Experience

This module provides the front-end interface to users for a smooth experience.

1. FR1: The system shall present a welcome screen, an input screen, and a 3D output screen.

2. FR2: The system shall provide real-time feedback and loading indicators while the 3D model is being retrieved or generated.

3. FR3: The system shall allow users to zoom, rotate, and navigate the displayed 3D models interactively.

2.3 Non-Functional Requirements

Non-functional requirements define the quality attributes of the system.

2.3.1 Reliability

- REL-1: The system shall have a Mean Time Between Failures (MTBF) of at least 500 hours.
- REL-2: In case of a failed model retrieval, the system shall attempt fallback generation within 10 seconds.
- REL-3: All user inputs shall be validated to prevent failure from malformed or unsupported input types.

2.3.2 Usability

- USE-1: The system shall allow users to access the 3D tour within three clicks from the home screen.
- USE-2: The interface shall support accessibility standards including text-to-speech and high contrast options.
- USE-3: New users shall be able to generate and view a 3D model without needing instructions in under 2 minutes.

2.3.3 Performance

- PER-1: 95
- PER-2: 3D model generation from a 360-degree image set shall complete within 90 seconds on a standard Google Colab GPU runtime.
- PER-3: User interactions (zooming, rotation, panning) on the 3D output screen shall respond within 100 milliseconds.

2.3.4 Security

- SEC-1: All data transmission between the client and Firebase shall be encrypted using HTTPS.
- SEC-2: User-generated images and models shall be securely stored with access control based on user authentication.
- SEC-3: The system shall resist unauthorized access through rate limiting and input sanitization to prevent injection attacks.

Chapter 3

System Overview

This project is focused on building an intelligent and interactive **Virtual 3D Tour Application**. The system enables users to experience immersive 3D tours of real-world locations by uploading 360° images, entering place names, or providing textual descriptions. These inputs are processed to either retrieve a corresponding 3D model or generate one using advanced neural rendering techniques (e.g., NeRF, Instant-NGP).

The frontend is developed using Flutter, offering a responsive and user-friendly experience across platforms. The backend employs a combination of Firebase (for storage and metadata), machine learning pipelines (for 3D reconstruction or retrieval), and a database for mapping input queries to 3D assets.

3.1 Architectural Design

The architecture of the system follows a **Client-Server Model with a Modular Design Pattern**, ensuring scalability, maintainability, and reusability. The system is divided into distinct components based on their functionality, such as input processing, 3D model generation/retrieval, storage, and rendering.

Major Modules

- **Frontend (Flutter App)**: Welcome Screen, Input Screen (Name/Image/Description), 3D Output Viewer
- **Backend Services**: Query Handler, 3D Model Retrieval, 3D Model Generator
- **Database (Firebase)**: Maps place names and image embeddings to 3D models
- **Storage (Firebase Storage)**: Stores .ply/.obj files and associated metadata

Box and Line Diagram

Flutter App (User Interface) ⟷ Backend Server (API + Logic) ⟷ Firebase Services (DB + Storage)

Architecture Pattern

- **Presentation Layer**: Flutter UI
- **Logic Layer**: Query handling, model retrieval/generation logic
- **Data Layer**: Firebase database and storage

3.2 Design Models

The system uses an object-oriented design approach. The following models are applicable:

Object-Oriented Design Models

- **Activity Diagram**: Illustrates user flow from input to 3D output
- **Class Diagram**: Core classes like `UserInputHandler`, `ModelRetriever`, `FirebaseManager`, etc.
- **Class-level Sequence Diagram**: Shows interaction of components during a complete 3D tour session
- **State Transition Diagram**: States such as Idle, Processing, Generating Model, and Rendering Output

3.3 Data Design

The system manages the following types of data:

- **User Inputs**: Text descriptions, place names, and 360° images
- **3D Models**: Stored as .ply/.obj files in Firebase Storage
- **Metadata**: Includes location ID, tags, generation source, timestamps

- **Embeddings**: Stored and indexed via FAISS for image and text similarity-based retrieval

Firebase Firestore Example


```
{
  "locations": {
    "istanbul_aya_sofia": {
      "place_name": "Aya Sofya",
      "embedding": [0.19, 0.83, ...],
      "model_url": "gs://.../ayasofya.obj",
      "tags": ["historical", "turkey"]
    }
  }
}
```


3.4 Domain Model

The domain model provides a conceptual overview of the system's entities and their relationships.

Entities and Attributes

- **UserInput**: inputType, inputValue, timestamp
- **Tour3DModel**: modelID, fileType, storageURL, source, associatedPlace
- **Place**: placeID, name, location, tags, embedding
- **RequestLog**: requestID, inputType, matchedPlaceID, resultStatus, processingTime

Relationships

- A `UserInput` can lead to multiple `Tour3DModels` (via retrieval or generation)
- A `Place` is associated with one or more `Tour3DModels`
- Each `RequestLog` corresponds to one user query and one resulting model

Associations

- `Tour3DModel` $\leftrightarrow$ `Place`: One-to-many
- `UserInput` $\leftrightarrow$ `RequestLog`: One-to-one
- `RequestLog` $\rightarrow$ `Tour3DModel`: Many-to-one

Constructing this domain model helps clarify the business logic and guides both backend development and database schema design.

Chapter 4

Implementation and Testing

4.1 General Description

The Virtual 3D Tour project is designed to transform 2D images or textual descriptions of places into interactive 3D virtual environments that users can explore seamlessly through a web or mobile application. The core functionality includes accepting multiple input types (place name, images, or textual descriptions), processing the inputs to generate or retrieve accurate 3D models, and rendering these models in an immersive 3D viewer.

The system integrates computer vision, 3D reconstruction algorithms, and cloud storage solutions to provide a user-friendly and efficient 3D tour experience. The backend handles the heavy computational tasks such as 3D model generation and embedding-based retrieval, while the frontend offers intuitive interaction with the generated 3D content.

4.2 Algorithm Design

The project consists of several major modules:

- **Input Processing Module:** Parses the input (place name, image, or description), performs similarity search or triggers 3D model generation.
- **3D Model Generation Module:** Uses Neural Radiance Fields (NeRF) techniques to generate 3D models from 360-degree image datasets.
- **Retrieval Module:** Searches existing 3D models in the database using embedding-based similarity for image or text queries.
- **Rendering Module:** Displays the generated or retrieved 3D models in an interactive viewer on the frontend.

Core Algorithm (Pseudocode)

Algorithm 1 Virtual3DTour(input)

input_data place_name or image or description 3D_model for visualization input_data is place_name model_id ← searchDatabaseByPlaceName(input_data) model_id exists retrieve3DModel(model_id) "Model Not Found" input_data is image image_embedding ← generateImageEmbedding(input_data) model_id ← searchDatabaseByEmbedding(image_embedding) model_id exists retrieve3DModel(model_id) generate3DModelFromImages([input_data]) input_data is description text_embedding ← generateTextEmbedding(input_data) model_id ← searchDatabaseByEmbedding(text_embedding) model_id exists retrieve3DModel(model_id) "No matching model found"

Algorithm 2 generate3DModelFromImages(image_set)

preprocessImages(image_set) runNeRFTraining(image_set) exportModel(format=".ply" or ".obj") storeModelInDatabase() model_path

4.3 External APIs/SDKs

The project uses several third-party APIs and SDKs to accelerate development and provide essential functionalities. The following table summarizes the main external tools:

API and Version	**Description**	**Purpose of Usage**	**API Endpoint/Function/Class Used**
Google Cloud Storage	Cloud storage for 3D models	Store and retrieve generated 3D files	`storage.bucket().upload()`
Firebase Firestore	NoSQL database	Store metadata, embeddings, and mappings	`firestore.collection('models').doc(id)`
OpenAI CLIP	Image and text embedding model	Generate embeddings for similarity search	`clip.encode_image()`, `clip.encode_text()`
Instant Neural Graphics Primitives (Instant-NGP)	3D model reconstruction	Fast NeRF-based 3D reconstruction	Internal library functions (custom Colab code)
Three.js	JavaScript 3D rendering engine	Render 3D models in frontend	`THREE.Mesh()`, `THREE.PerspectiveCamera()`

Table 4.1: External APIs and SDKs Used

4.4 Testing Details

Once the system has been successfully developed, testing is performed to ensure that the system works as intended and meets the specified requirements. Testing helps identify any hidden errors before deployment.

4.4.1 Unit Testing

Unit testing is a critical phase in the software development process that involves testing individual components or modules of the system in isolation to ensure that each part functions correctly. The primary goal of unit testing is to validate that each unit of the software performs as designed and meets its specifications.

Each unit test is designed to test a specific function or method independently from other components, helping to identify issues directly related to the functionality being tested. Unit tests are often automated, allowing for quick execution and repeatability, enabling frequent running during integration or regression testing to catch bugs early.

A well-structured unit testing suite aims to achieve high test coverage, including positive cases, edge cases, and error conditions. Unit testing provides immediate feedback to developers about code quality, helping maintain reliability and making it easier for new developers to understand the system.

Test case ID	Test Objective	Precondition	Steps	Test data	Expected result	Post-condition	Actual Result	Pass/fail
TC001	Verify admin login with username and password	Admin should be registered with valid email and password before login.	Click on Login button Enter valid username and password	Email-id: abc@xyz.com Password: Xyz123	System displays Admin homepage	Admin should be kept logged in until logout.	As Expected,	Pass

Figure 4.1: Example for Unit Testing

Unit tests in this project cover:

- Embedding generation for images and text inputs.
- Database query functions for retrieving 3D models.
- 3D model export and storage modules.

Automated testing frameworks such as `pytest` (for Python backend) are used to run these tests on every code update.

4.4.2 Integration and System Testing

After unit testing, integration tests verify the interaction between modules, ensuring smooth data flow from input parsing through to 3D rendering. System testing includes:

- Validating retrieval of 3D models by place name, image, and description inputs.
- Confirming 3D model generation pipeline functions correctly when no matching model exists.
- Ensuring 3D models render properly in the frontend viewer.
- Performance testing to measure 3D model generation and retrieval latency.

User Acceptance Testing (UAT) is conducted with real users exploring virtual tours to collect feedback and improve usability before final deployment.

—

Chapter 5

Conclusions and Future Work

5.1 Conclusion

In this project, we successfully developed a virtual 3D tour application that enables users to explore real-world places through interactive 3D models. The system accepts various inputs — a place name, an image, or a descriptive prompt — and generates corresponding 3D representations using a combination of image-based 3D reconstruction (e.g., NeRF) and pre-trained generative models. These models are displayed in an interactive viewer, enhancing user experience and accessibility.

We integrated Firebase for efficient storage and retrieval of 3D assets and metadata, allowing quick access to previously generated models. Our backend pipeline includes image embedding generation, nearest-neighbor search, and fallback mechanisms for unseen inputs, ensuring robustness. The frontend was implemented using Flutter, providing a smooth, cross-platform interface with screens for input, results, and navigation.

Overall, the system demonstrates the feasibility and effectiveness of combining deep learning, image processing, and modern app development frameworks to create immersive virtual experiences.

5.2 Future Work

Although our application demonstrates strong performance in generating and visualizing 3D tours, several areas can be enhanced further:

- **Real-time 3D Model Generation:** Implementing real-time 3D generation with improved speed and accuracy using optimized pipelines such as Instant-NGP integrated with Nerfstudio on cloud or edge devices.

- **Improved Input Interpretation:** Enhancing natural language understanding for descriptive prompts using advanced transformer models and grounding with visual features.
- **Multimodal Search:** Adding multimodal embedding-based retrieval to allow better matching between text, image, and 3D assets.
- **Semantic Labeling and Navigation:** Integrating semantic segmentation and labeling in the 3D models for guided navigation or educational tours.
- **User Personalization:** Incorporating user preferences and history to offer personalized tour suggestions or highlight points of interest.
- **Support for VR Devices:** Expanding support to VR headsets for more immersive virtual travel experiences.
- **Collaborative or Social Tours:** Allowing multiple users to join and explore the same 3D environment in real time, adding a social dimension to the experience.

[4] [3] [2] [1] [5]

Bibliography

[1] Livio De Luca, Philippe Véron, and Michele Florenzano. 3d virtual tourism via web-based interactive visualization of cultural heritage. *Journal of Cultural Heritage*, 41:1–11, 2020.

[2] Hyeonggeun Kim, Donghwan Lee, and Chanwoo Park. Immersive virtual tour system using 360-degree images and videos. In *2019 IEEE International Conference on Consumer Electronics (ICCE)*, pages 1–2. IEEE, 2019.

[3] Bo Li, Min Xu, Yiqun Wang, et al. Real-time 3d scene reconstruction with consumer rgb-d cameras. *Sensors*, 16(5), 2016.

[4] Fabio Remondino and Alessandro Rizzi. Heritage recording and 3d modeling with photogrammetry and 3d scanning. *Remote Sensing*, 3(6):1104–1138, 2011.

[5] Matthew et al. Tancik. Nerfstudio: A modular framework for neural radiance fields. Online Documentation, 2022. https://docs.nerf.studio.

Appendices

Appendix A – Use Case Diagram

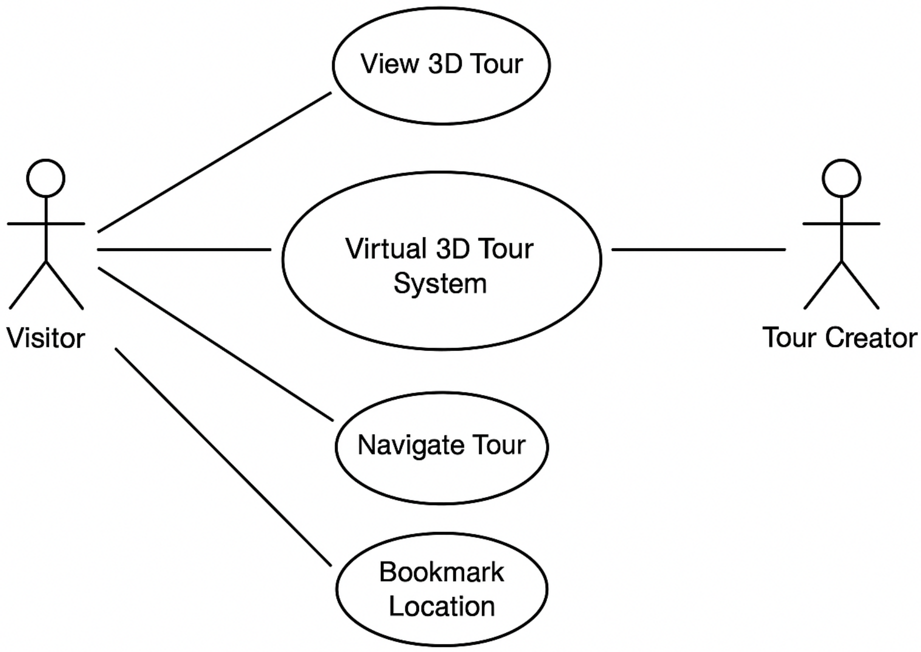

Figure 1: Use Case Diagram for theVirtual 3D Tour System

Figure 1: Use Case Diagram for the Virtual 3D Tour System

Appendix B – Domain Model

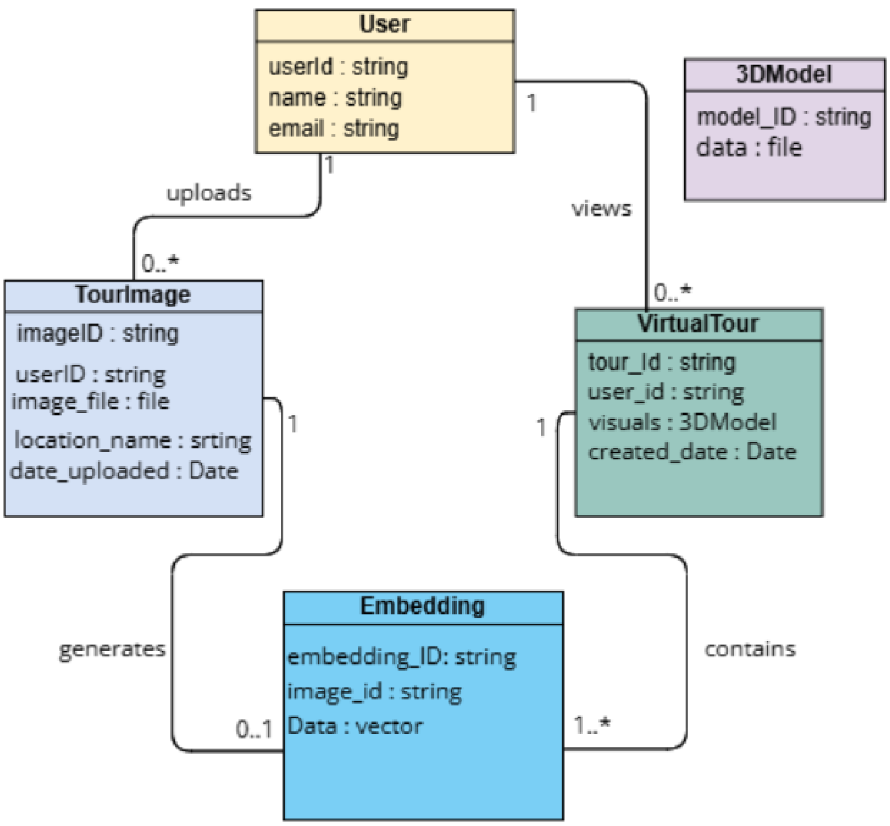

Figure 2: Domain Model for Virtual 3D Tour System

Appendix C – UML Diagrams

Activity Diagram

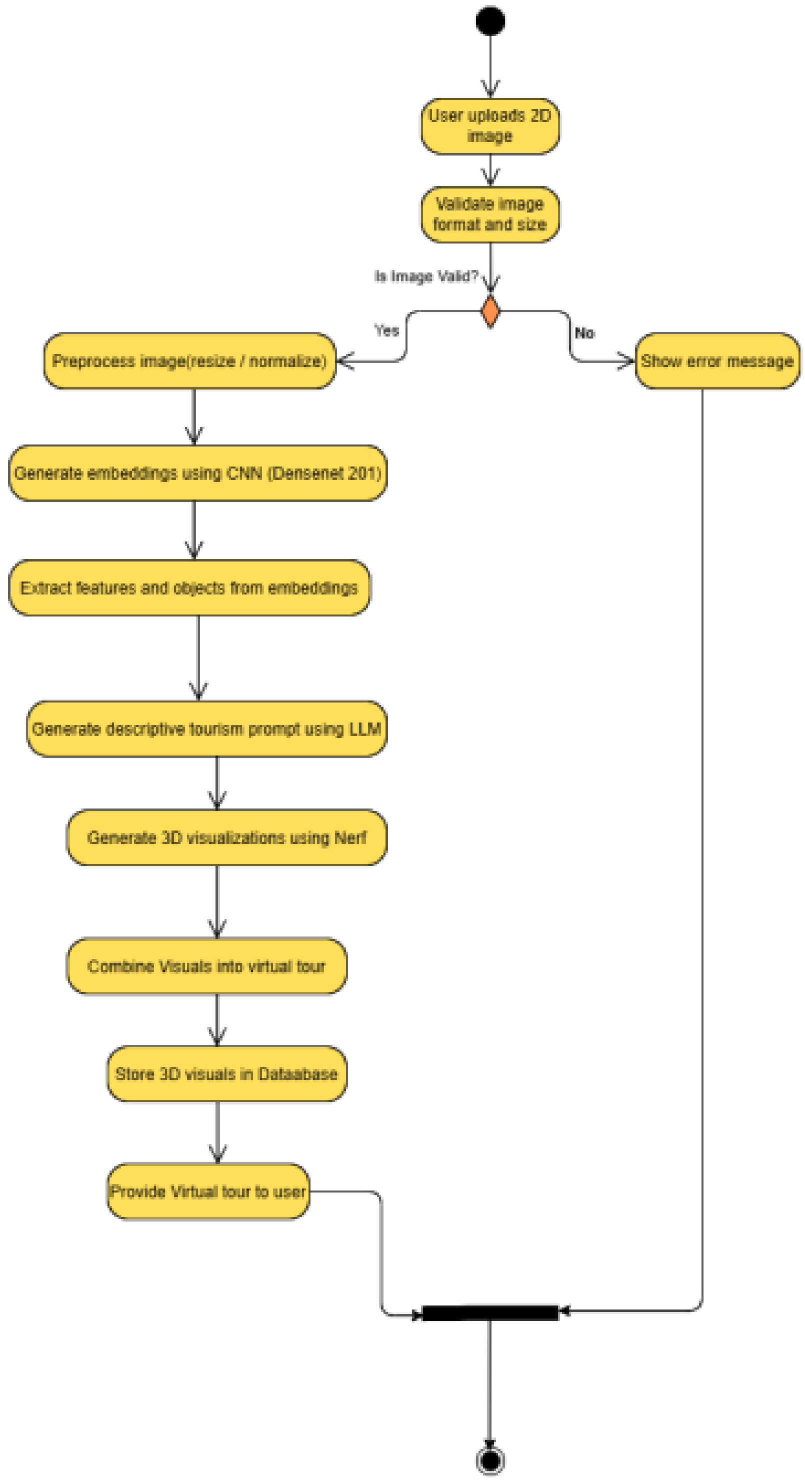

Figure 3: Activity Diagram for Virtual 3D Tour System

Class Diagram

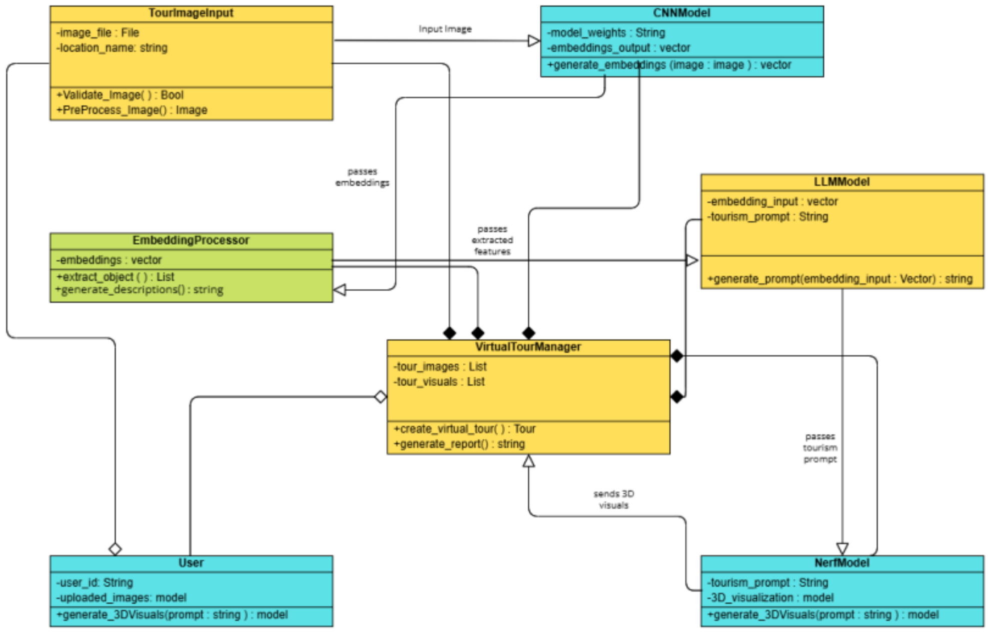

Figure 4: Class Diagram for Virtual 3D Tour System

Sequence Diagram

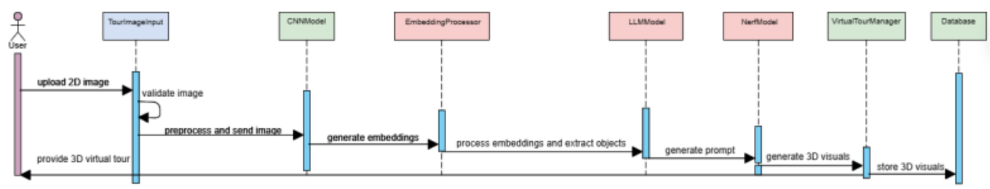

Figure 5: Sequence Diagram for Virtual 3D Tour Interaction

State Transition Diagram

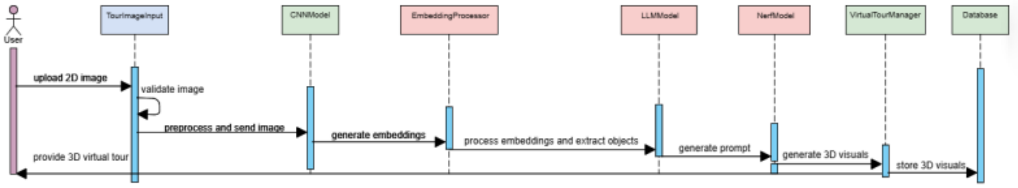

Figure 6: State Transition Diagram for Virtual 3D Tour System