Assignment

Task 1
App 1: Deep-Art:

Functionality:
Deep-Art is an AI-powered application that transforms ordinary images into artistic masterpieces by applying various artistic styles.
Users can upload their photos and choose from a selection of artistic styles, such as famous paintings or unique visual patterns.
The app leverages deep learning algorithms to analyze the input image and recreate it in the chosen artistic style, producing visually stunning results.
It provides a creative and accessible way for users to explore different artistic expressions through their own photos.

RunwayML:

Functionality:
RunwayML is a versatile platform that allows users to easily integrate and run generative AI models without extensive programming knowledge.
The app supports a wide range of pre-trained models spanning various domains, including art, music, text, and more.
Users can experiment with and apply these models to generate creative outputs, such as generating artwork, composing music, or even manipulating text.
It caters to both developers and artists, providing a user-friendly interface and a hub for accessing and deploying generative models.
RunwayML empowers users to explore the capabilities of generative AI across different creative disciplines.

High-Level Architecture Design: DeepArt

Components:

Frontend: User interacts with the DeepArt web interface to upload images and select artistic styles.
Backend Server: Receives requests from the frontend, handles image processing using deep learning models.
Deep Learning Model: Utilizes a pre-trained deep neural network for style transfer.
External Storage: Stores user-uploaded images and processed artworks.
Data Flow:

User uploads an image through the frontend.
The Frontend sends the image data to the Backend Server.

The Backend processes the image using the deep learning model to apply the chosen artistic style.
The processed artwork is sent back to the Frontend.
The Frontend displays the transformed image to the user.
High-Level Architecture Design: RunwayML

Components:

RunwayML Platform: Web-based platform where users access and run various pre-trained generative models.
Model Hub: Central repository of pre-trained models in various domains.
User Workspace: Area where users experiment with models and generate outputs.
RunwayML Client: A client-side application that communicates with the platform and manages model execution.
Generative Models: Diverse set of pre-trained models for different creative tasks.
Data Flow:

User interacts with the RunwayML Platform through the web interface.
The Platform communicates with the Model Hub to fetch information about available models.
User selects a model and starts the execution in their workspace.
The RunwayML Client sends data (text, images, etc.) to the selected model.
The Generative Model processes the input and generates creative outputs.
The resulting output is sent back to the RunwayML Client and displayed to the user.

API Endpoint Doc: DeepArt

Endpoint: /uploadImage

Request:
Method: POST
Parameters: Image file
Response:
Status: 200 OK
Data: URL to the processed artwork
Endpoint: /availableStyles

Request:
Method: GET
Response:
Status: 200 OK
Data: List of available artistic styles
API Endpoint Doc: RunwayML

Endpoint: /listModels

Request:
Method: GET
Response:
Status: 200 OK

Data: List of available generative models
Endpoint: /runModel

Request:
Method: POST
Parameters: Model ID, Input data (text, image, etc.)
Response:
Status: 200 OK
Data: Output generated by the selected model

Task 2:

Invoice Web App(Link)
•Developed a web application enabling users to generate, send, and download receipts, quotations etc. through seamless email integration resulting in a 20% reduction in manual
paperwork and workload.
•Designed a highly efficient records dashboard using React.js and MongoDB, showcasing all
details about the invoice Generated and resulting in an 28% increase in invoice accuracy.
•Store generated bills securely in a MongoDB database for easy retrieval and send them again
via email as well.
•Integrated Nodemailer for during development as well as same for production email delivery.
•Provided a secure user authentication system using JWT, combined with Google auth for user
convenience.
•Technical Skills-React.js, Node.js, Express.js, MongoDB,JWT (JSON Web Tokens),JsPdf,
Nodemailer, Typescript.

Hosting: Hostinger for the front-end nd for the backend, onrender.com was used.

Project link: https://invoice.simrantandon.live/landing