HELWAN UNIVERSITY
Faculty of Computers and Artificial Intelligence
Computer Science Department

TRY & BUY

A graduation project dissertation by:

Abdelrahman Magdy Abdelhamid....................................... 20210522

Abdallah Ragab Abdelkader.. 20210545

Shawqy Hussien Shawqy Sayed.. 20210472

Nadia Talaat Hasan Mohamed.. 20210988

Nada Mohamed Abdelrahman... 20211000

Amira Hasan Sobhi Hegazi…... 20210185

Submitted in partial fulfilment of the requirements for the degree of Bachelor of Science in Computers & Artificial Intelligence, at the Computer Science Department, the Faculty of Computers & Artificial Intelligence, Helwan University

Supervised by:

[Dr. Ahmed Hesham Mostafa]

June 2025

Acknowledgement

As We complete our academic journey, we first thank Allah for His guidance and blessings. Our heartfelt gratitude extends to **our families** whose unwavering support sustained us through countless challenges and late-night coding sessions.

We sincerely thank **the faculty at FCAI, Helwan University,** whose dedication shaped our intellectual growth.

Special appreciation goes to our supervisor, **Dr. Ahmed Hisham**, whose mentorship and technical expertise guided our research ambitions.

To the entire **FCAI-HU community**—fellow students, staff, and support teams—thank you for creating an environment where innovation flourished.

This four-year odyssey has equipped us with not just technical skills, but the resilience and critical thinking needed for our professional futures.

We move forward carrying the knowledge and values gained at Helwan University with profound gratitude. The evolving landscape of technology. As we embark on the next chapter of our professional lives, we carry forward the values, knowledge, and experiences gained at Helwan University with profound gratitude and pride.

Contents

Abstract

The rapid growth of e-commerce has revolutionized the shopping experience, particularly in the fashion industry. Despite the advancements in online shopping platforms, a significant challenge persists customers are unable to try on clothing before making a purchase. This lack of a personalized fitting experience often leads to uncertainty, poor product selection, high return rates, and decreased customer satisfaction.

To address this issue, this project proposes an innovative AI-powered virtual try-on system, specifically designed for fashion e-commerce platforms. The system allows users to upload their own images and see how different clothing items look on them in real time, offering a realistic and interactive fitting experience.

By integrating innovative computer vision and machine learning technologies, the system is capable of accurately overlaying garments onto the user's image, ensuring a seamless and highly realistic fit with minimal manual intervention. This system not only enhances the user's decision-making process by providing a clearer visual representation of the clothes, but it also reduces the uncertainty associated with online shopping.

Additionally, the platform incorporates essential e-commerce features like any other platform such as product browsing, cart management, and secure payment processing, making the entire shopping experience more engaging and user-friendly.

The goal of this project is to reduce product return rates, improve customer confidence, and increase sales by offering a personalized shopping experience that closely mirrors in-store fitting, but with the convenience of online shopping. This project also highlights the transformative potential of artificial intelligence in solving real-world challenges in the fashion sector, making it a valuable contribution to both technological innovation and the future of e-commerce.

Chapter 1: Introduction

This chapter introduces our AI-powered virtual try-on solution for fashion e-commerce, addressing the inability of customers to visualize garments on themselves before purchasing. We outline how this technology enhances purchase confidence, reduces returns, and creates a personalized shopping experience through real-time garment visualization.

1.1 overview

The increasing reliance on e-commerce platforms has reshaped consumer behavior, especially in the fashion and clothing industry. However, one major drawback remains unresolved: the inability of users to try on garments before purchasing. This limitation significantly affects user satisfaction, return rates, and purchase confidence. To address this challenge, the proposed project introduces an innovative web-based AI-powered Try-On System, specifically tailored for clothing ecommerce.

This system allows users to virtually try on clothes using an uploaded image. By leveraging computer vision and machine learning techniques, the AI component overlays garments onto the user's body image in real time, providing an interactive and realistic simulation of how each item would appear. In addition to the Try-On feature, the platform incorporates standard e-commerce functionalities including user registration, product browsing, searching and filtering, cart management, checkout, and secure payment.

The system is designed with a user-centered interface and provides a dedicated admin dashboard for managing users, products, and orders efficiently. The primary focus is to enhance user engagement, minimize returns, and provide a competitive advantage over conventional ecommerce systems by personalizing the shopping experience through AI integration.

1.2 Purpose

The purpose of this project is to enhance the online shopping experience by introducing an AI-powered virtual try-on system tailored for fashion e-commerce.

In recent years, online shopping has grown substantially due to its convenience and accessibility. However, a critical concern—especially in fashion—remains unresolved: "Will this product look good on me?" Traditional platforms rely on static images and size charts, which do not offer customers a realistic idea of how products would look on their own bodies.

This uncertainty often leads to:

- hesitation or lack of confidence in making purchases,
- high return rates due to poor fit or unmet expectations, and
- decreased user engagement and satisfaction.

To solve this, the project integrates computer vision and machine learning technologies to allow users to visualize clothing items on themselves in real time. This approach empowers consumers to make more informed decisions, increases interaction with the platform, and boosts trust in product selection.

From both technical and academic perspectives, the project demonstrates a meaningful application of artificial intelligence to solve real-world problems in the e-commerce industry.

1.3 Problem Definition

Despite the growth of online retail, most fashion e-commerce platforms still lack personalized fitting experiences. Customers are often required to rely on static product images and general size charts, which fail to provide an accurate representation of how garments would appear on their individual body types.

This disconnects between product presentation and user perception results in several key issues:

1. Inaccurate size or style selection
2. Customer dissatisfaction and frustration
3. Increased return and refund requests
4. Decline in consumer trust in online shopping

The need for a more intelligent and personalized solution is particularly evident in areas where physical store access is limited or where contactless shopping is preferred for convenience or health-related reasons.

This project seeks to address this gap by introducing an AI-powered virtual try-on system that enables users to visualize clothing items on themselves in real-time. In addition to enhancing the decision-making process, the system is designed with secure, responsive, and maintainable architecture to ensure seamless integration into modern e-commerce platforms.

1.4 Solution Description

Developing an effective virtual try-on system involves overcoming several technical challenges, particularly in ensuring the accuracy and consistency of clothing and user images. One of the key issues lies in collecting high-quality images or models of clothing items. Variations in lighting, angles, and fabric textures can significantly affect the visual realism and alignment of garments during the try-on process.

To address this, standardized photography techniques are recommended. Clothing items should be photographed under consistent lighting conditions, with fixed camera angles and neutral backgrounds to ensure uniformity across the dataset. In more advanced implementations, 3D scanning technology can be employed to generate highly detailed and realistic models of garments, enabling more accurate fitting simulations.

On the user side, capturing accurate images is equally critical. Users differ in body shapes, poses, and camera environments. To ensure compatibility, the system should include an automated image-capture assistant or set of prompts that guide users to upload front-facing photos under well-lit conditions with minimal obstructions. This improves the reliability of clothing overlay and enhances the realism of the virtual try-on experience.

These image collection and standardization processes form the foundation for delivering a personalized and accurate AI-powered fitting system within the broader e-commerce platform.

1.5 Results

The implementation of the proposed AI-powered Try-On System is expected to produce several key results that contribute to the overall success and practicality of the platform.

First, the system is designed to utilize high-quality and standardized images or 3D models of clothing items, ensuring visual consistency and accurate garment representation. Additionally, by guiding users to upload front-facing, well-lit photos, the system achieves precise user data suitable for virtual try-on without extensive manual preprocessing.

A significant technical achievement is the seamless integration of garment overlays onto user images, allowing real-time visualization that closely resembles how the clothing would fit in reality. This enhances user interaction and satisfaction by delivering a realistic and interactive shopping experience.

Collectively, these results improve purchase confidence, reduce product return rates, and support a more engaging and personalized e-commerce environment.

1.6 Similar Systems

Several virtual try-on systems have emerged in recent years, aiming to enhance the online shopping experience by allowing users to preview products before purchasing. Below are some notable examples

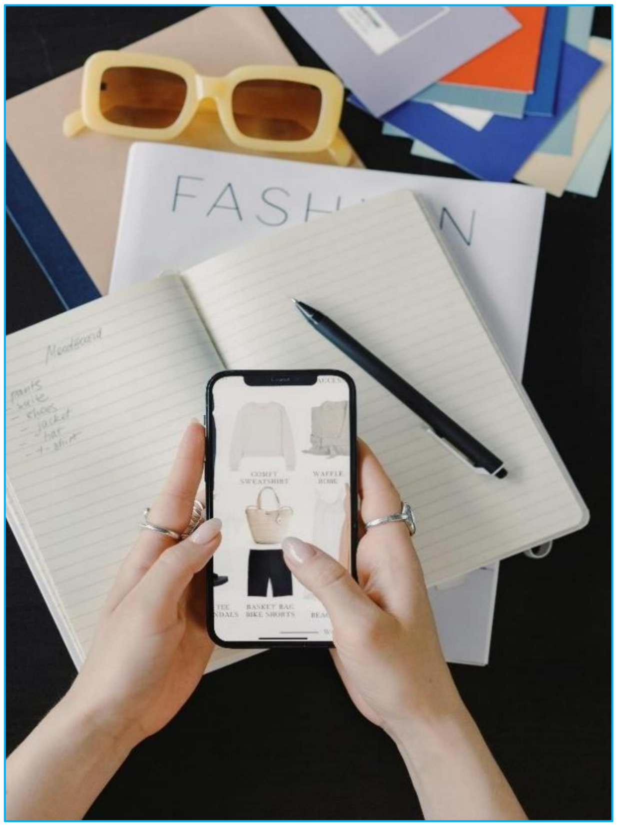
System Name	Description	Category / Input Type	Features
Zyler	Zyler uses advanced AI to create virtual fitting rooms for online shoppers. Users can upload a photo and try on various clothing items in real-time.	Clothing / Photo Upload	Customizable try-on options, realistic fitting, and integration with online retail platforms.
Ray-Ban	Ray-Ban's app allows users to virtually try on different eyewear styles using augmented reality.	Eyewear / Live Camera	AR eyewear try on, frame customization, and direct purchase options
YouCam Makeup	YouCam Makeup offers a virtual try-on experience for cosmetics. Users can apply makeup products to their photos or live camera feeds to see how different products look.	Makeup Photo / Camera	Real-time makeup application, recommendations, beauty consultations VIRTUSIZE and product virtual
ModiFace	ModiFace provides virtual try-on solutions for both cosmetics and eyewear. Users can see how products look on their own images or live camera feeds.	Cosmetics / Live Camera	Virtual try-on for makeup and eyewear, facial recognition, and realtime AR experience.
Virtusize	Virtusize helps users visualize how clothes will fit by comparing them with items they already own. It integrates with online stores to offer virtual fitting solutions.	Clothing / Size Comparison	Fit comparison, virtual size guide, and integration with e-commerce platforms.

1.7 Tools and Platforms Used:

The development and deployment of the AI-powered Try-On System involved the use of various tools and platforms. These applications were selected for their compatibility, performance, and support for AI integration:

- VS Code – Integrated development environment (IDE) used for writing and managing code.
- HTML, CSS, JS – For building responsive and user-friendly frontend interfaces.
- Xamp(MySQL) – For database management.
- Git + GitHub – For version control and collaboration.
- Postman – To test and debug RESTful API endpoints.
- Python – Primary programming language for AI modules and backend logic.
- PyTorch – To build and train the machine learning models used for garment fitting.
- Kaggle
- Gradio

Chapter 2: IDM-VTON Model for Virtual Try-On



This chapter examines the IDM-VTON diffusion model architecture that combines TryonNet, IP-Adapter, and GarmentNet components to achieve superior garment fidelity and pose generalization. The model preserves fine-grained garment details while generating photorealistic try-on results, outperforming previous GAN and diffusion-based approaches across multiple evaluation metrics.

2.1 Background

- Virtual Try-On (VTON) is a critical technology for e-commerce, reducing return rates by allowing users to visualize garments digitally.
- Traditional GAN-based methods suffer from artifacts and poor generalization. Diffusion models (e.g., Stable Diffusion) offer photorealistic results but struggle with garment fidelity.

2.2 Objective

Develop **IDM-VTON**, a diffusion-based model that:

- Preserves fine-grained garment details (textures, logos).
- Generalizes to in-the-wild poses and backgrounds.
- Supports customization with minimal user data.

2.3 Key Contributions

1. **Dual garment encoding**: Combines high-level (CLIP) and low-level (UNet) features.
2. **Customization**: Fine-tunes with single user-garment pairs.
3. **Detailed captions**: Improves semantic alignment via text prompts.

2.4 Problem Statement

a. Challenges in VTON

Challenge	GAN-Based Methods	Prior Diffusion Models
Garment fidelity	Low	Moderate
Pose generalization	Poor	Limited
Real-world adaptability	No	No

b. Research Questions

1. How to encode garments without losing details?
2. Can diffusion models outperform GANs in fidelity?
3. How to adapt to unseen poses/garments?

2.5 Methodology

a. Diffusion Models Primer

- Forward/reverse process with noise

b. IDM-VTON Pipeline

1) **Input**: Person image (x_p), garment (x_g), mask (m), DensePose (x_pose).

2) **Garment Encoding**:
   - **IP-Adapter**: CLIP ViT-H/14 for semantics.
   - **GarmentNet**: SDXL UNet for low-level features.
3) **Fusion**: Cross-attention (high-level) + self-attention (low-level).

2.6 Model Architecture

a. TryonNet

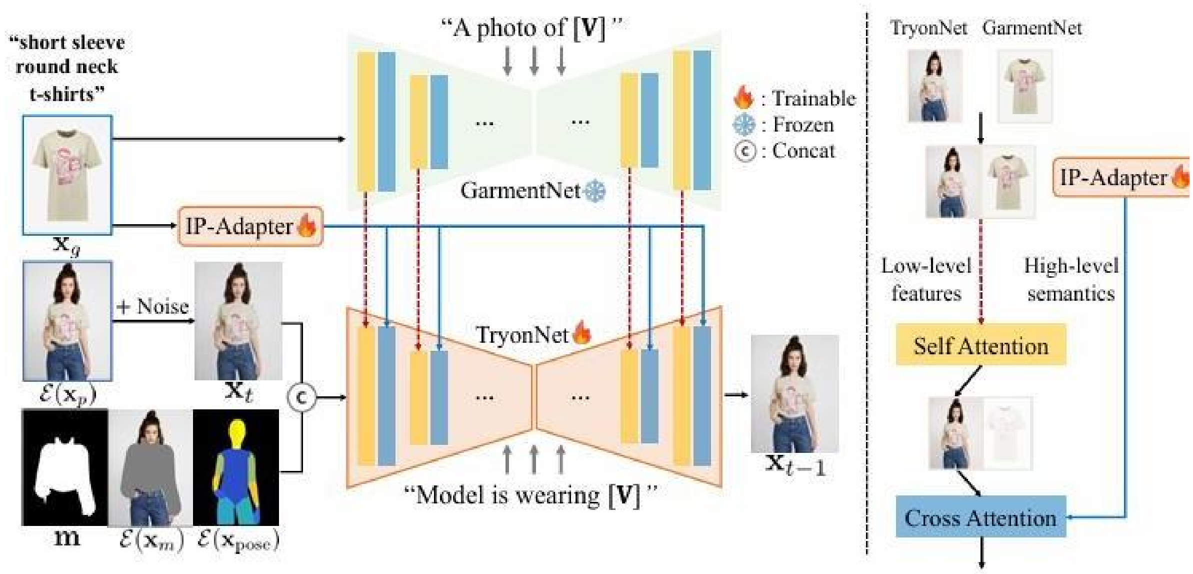
For the base UNet model, we consider latent diffusion model , where the diffusion generative modeling is conducted in latent space of variational autoencoder E and the output is passed to decoder D to generate an image. As of the input for our base UNet, we concatenate four components, the latent of person image, the (resized) mask m that removes the garment on the person the latent of masked-out person image, the latent of the Densepose xpose of a person image, i.e., Epxposeq. Then, the latents are aligned within the channel axis, where we expand the convolutional layer of UNet to 13 channels initialized with zero weights. Unlike previous works on virtual try-on with diffusion models we leverage Stable Diffusion XL (SDXL) inpainting model.

b. Image prompt adapter

To condition the high-level semantics of a garment image, we leverage image prompt adapter (IP-Adapter) . To encode the garment image, we use frozen CLIP image encoder to extract feature and fine-tune feature projection layers and cross-attention layers, which are initialized with pretrained IP-Adapters. Note that we pass textual prompts of garments, and the cross-attention is computed.

c. GarmentNet

While we already condition the garment image using IP-Adapter, it falls short in preserving the fine-grained details of garments when it has complicated patterns or graphic prints, This is because the CLIP image encoder lacks extracting the low-level features of a garment. To tackle this issue, we propose to utilize an additional UNet encoder to encode fine-details of garment images. Given the latent of a garment image Epxgq, we pass through the (frozen) pretrained UNet encoder to obtain the intermediate representation, and concatenate with the intermediate representation



Now, we present our method for designing diffusion models for virtual try-on. Let us denote xp be the image of a person, and xg be the image of a garment. Our primary goal is to generate an image xtr that visualizes a person from xp wearing a garment in image xg. It is common practice to cast virtual try-on as an exemplar based image inpainting problem , which aims to fill the masked image with a reference image. Here, it is important to extract relevant information of garment, and add conditional control to the diffusion model.

To this end, our model is composed of three components, the base UNet (TryonNet) that processes the masked person image with its pose information, the image prompt adapter (IP-Adapter) that extracts the high-level semantics of the garment, the garment UNet feature encoder (GarmentNet) that extracts the low-level features of the garment. The features from GarmentNet are fused within self-attention layer of TryonNet, and then it is processed with the features of IP-Adapter through cross-attention layer. We provide an overview of our method, and provide detailed explanation of each component as follows.

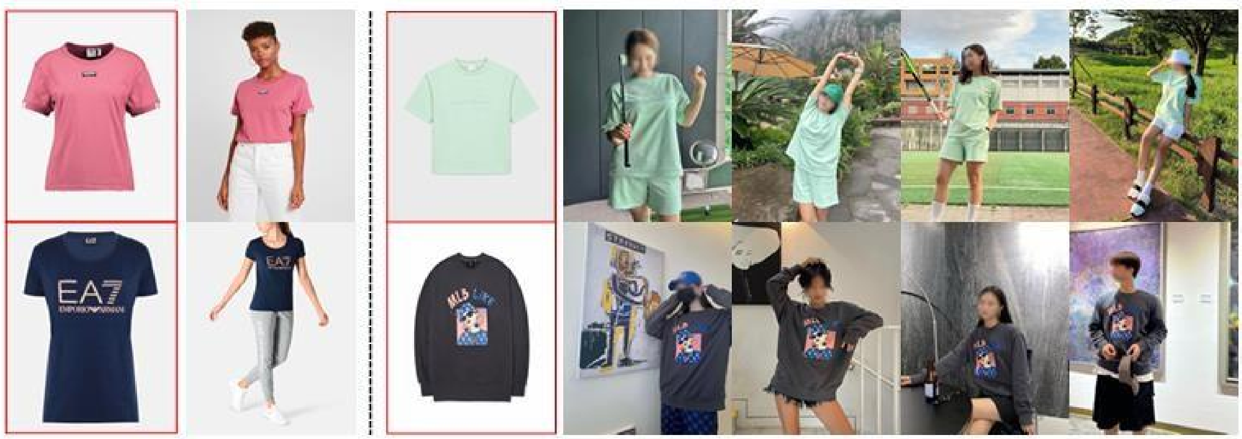
Dataset	VITON-HD				DressCode			
Method	LPIPS ↓	SSIM ↑	FID ↓	CLIP-I ↑	LPIPS ↓	SSIM ↑	FID ↓	CLIP-I ↑
GAN-based methods								
HR-VITON [27]	0.115	0.883	9.70	0.832	0.112	0.910	21.42	0.771
GP-VTON [54]	0.105	**0.898**	6.43	0.874	0.484	0.780	55.21	0.628
Diffusion-based methods								
LaDI-VTON [31]	0.156	0.872	8.85	0.834	0.149	0.905	16.54	0.803
DCI-VTON [9]	0.166	0.856	8.73	0.840	0.162	0.893	17.63	0.777
StableVITON [23]	0.133	0.885	6.52	0.871	0.107	0.910	14.37	0.866
IDM–VTON (ours)	**0.102**	0.870	**6.29**	**0.883**	**0.062**	**0.920**	**8.64**	**0.904**

Quantitative results on VITON-HD dataset. We evaluate the recon struction scores (e.g., LPIPS, SSIM) for low-level similarity, CLIP image similarity score (CLIP-I) for high-level semantic similarity, and FID score for image fidelity.

We compare with GAN-based virtual try-on methods (e.g., HR-VTON and GP-VTON), and Diffusion-based virtual try-on methods (e.g., LaDI-VTON, DCI-VTON, StableVITON). Bold denotes the best score for each metric.

Qualitative results on VITON-HD and DressCode dataset. We show generated virtual try-on images using IDM–VTON (ours) compared with other methods on (a) VITON-HD , and (b) DressCode (upper body) test datasets.

We see that IDM–VTON outperforms others in generating authentic images and preserving fine-grained details of garment. Best viewed in zoomed, color monitor.



(a) Public Dataset (b) In-the-wild Dataset

Method	LPIPS ↓	SSIM ↑	CLIP-I ↑
HR-VITON [27]	0.330	0.741	0.701
LaDI-VTON [31]	0.303	0.768	0.819
DCI-VTON [9]	0.283	0.735	0.752
Stable-VITON [23]	0.260	0.736	0.836
IDM–VTON (ours)	0.164	0.795	0.901

Quantitative results on In-the-Wild dataset. We compare IDM–VTON (ours) with other methods on In-the-Wild dataset to assess the generalization capabilities. We report LPIPS, SSIM and CLIP image similarity scores. For IDM–VTON and StableVITON , we further customize models using a single pair of person-garment images. We see that IDM–VTON outperforms other methods, and customized IDM–VTON.

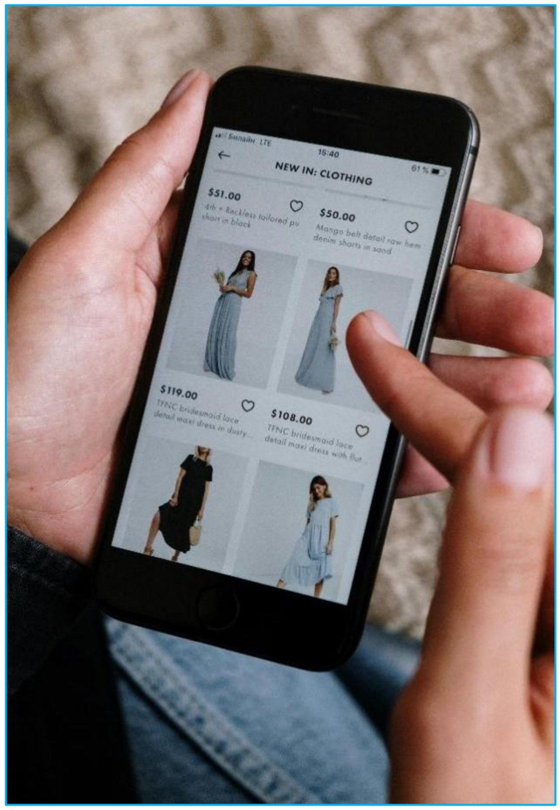
Effect of detailed caption for garment. We compare the generated virtual try-on images using naïve captions (left) and detailed captions (right). The model trained with detailed captions generates images that are more consistent with the garment. Best viewed in zoomed, color monitor.

GarmentNet	LPIPS ↓	SSIM ↑	CLIP-I ↑
✗	0.121	0.849	0.846
✓	**0.102**	**0.870**	**0.883**

2.7 Results and Limitations

- IDM–VTON struggles to preserve human attributes on masked regions such as tattoos or skin moles. One can design a method for conditioning those human attributes of when generating try-on images.
- Model act only on Upper Body .

Chapter 3: Advanced Virtual Try-On Implementation



This chapter outlines the Dress Code dataset and the implementation strategies employed to fine-tune the IDM-VTON model for virtual try-on tasks. We describe the dataset's structure, our fine-tuning approach across multiple garment categories, and the evaluation results. The implementation was optimized to work within Kaggle's computational limits while achieving robust performance.

A. Dataset

1) DressCode Dataset Overview

The DressCode dataset, introduced by Morelli et al. in their ECCV 2022 paper [1], represents a significant advancement in virtual try-on research. It is the first high-resolution, multi-category dataset specifically designed to address the limitations of previous virtual try-on benchmarks that primarily focused on upper-body garments.

2) Dataset Structure

The DressCode dataset contains 53,792 garments and 107,584 high-resolution images (1024×768) with size 76.13GB , making it more than three times larger than previously available datasets for image-based virtual try-on. The dataset is carefully organized into three distinct garment categories:

- Upper Body: Includes t-shirts, shirts, and tops (approximately 17,930 garments)
- Lower Body: Features pants, shorts, and skirts (approximately 17,930 garments)
- Dresses: Contains full-body dresses (approximately 17,930 garments)

Table 7. Number of train and test pairs for each category of the Dress Code dataset.

	Images	Training Pairs	Test Pairs
Upper-body Clothes	30,726	13,563	1,800
Lower-body Clothes	17,902	7,151	1,800
Dresses	58,956	27,678	1,800
All	107,584	48,392	5,400

Each garment is paired with two high-quality images: a front-view, full-body reference model wearing the garment and an isolated garment image on a clean background. This pairing enables both paired and unpaired virtual try-on scenarios, allowing for more versatile research applications.

4 D. Morelli *et al.*

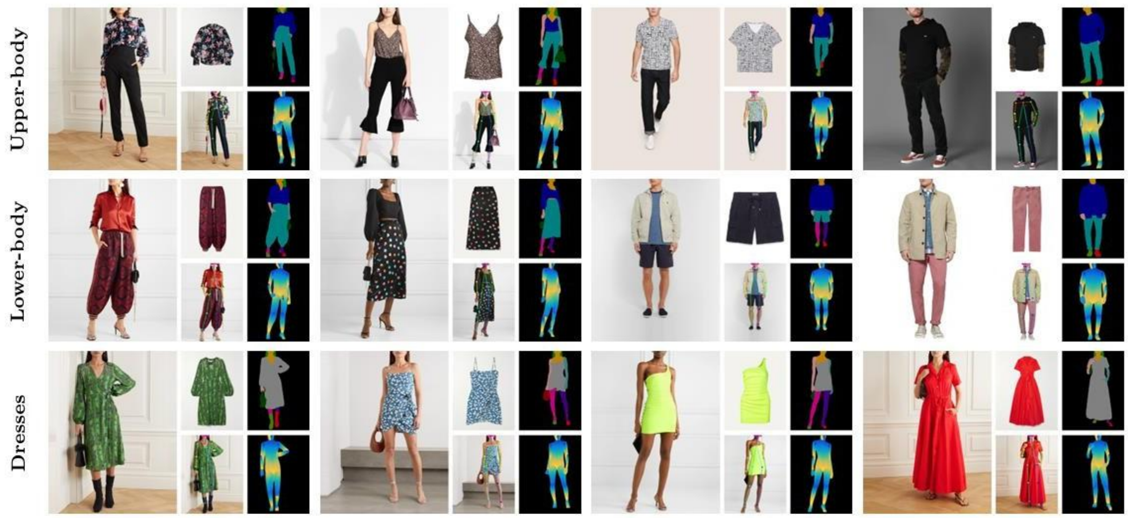


Fig. 2. Sample image pairs from the Dress Code dataset with pose keypoints, dense poses, and segmentation masks of human bodies.

Table 1. Comparison between Dress Code and the most widely used datasets for virtual try-on and other related tasks.

Dataset	Public	Multi-Category	# Images	# Garments	Resolution
O-VITON [30]	✗	✓	52,000	-	512 × 256
TryOnGAN [23]	✗	✓	105,000	-	512 × 512
Revery AI [24]	✗	✓	642,000	321,000	512 × 512
Zalando [43]	✗	✓	1,520,000	1,140,000	1024 × 768
VITON-HD [4]	✓	✗	27,358	13,679	1024 × 768
FashionOn [16]	✓	✗	32,685	10,895	288 × 192
DeepFashion [27]	✓	✗	33,849	11,283	288 × 192
MVP [6]	✓	✗	49,211	13,524	256 × 192
FashionTryOn [47]	✓	✗	86,142	28,714	256 × 192
LookBook [44]	✓	✓	84,748	9,732	256 × 192
VITON [13]	✓	✗	32,506	16,253	256 × 192
Dress Code	✓	✓	107,584	53,792	1024 × 768

For our implementation, we strategically selected a subset of 3,000 samples per category for training and 35 samples per category for testing.

This sampling approach balanced computational feasibility with sufficient data representation for effective model training.

The dataset follows a meticulously organized hierarchical structure:

Main Folder: Contains category-specific subfolders and annotation files:

- train_pairs.txt: Training pairs (model image | garment image)
- test_pairs_paired.txt: Paired test set annotations
- test_pairs_unpaired.txt: Unpaired test set annotations
- multigarment_test_triplets.txt: Triplets for multi-garment testing
- Category Folders (e.g., dresses, upper_body, lower_body): Each includes:
- images: Model (_0.jpg) and garment (_1.jpg) images
- key points: JSON files (_2.json) with 18 body joint locations
- skeletons: Pose visualization images (_3.jpg)
- label maps: Segmentation maps (_4.png)
- dense: DensePose UV maps (_5.npz) and label images (_5.jpg)
- image denspose :(_0.jpg)
- dc_caption.txt(contains captions for each garment type)

This comprehensive organization facilitates precise alignment and processing for virtual try-on tasks, enabling researchers to develop more sophisticated algorithms for garment transfer across diverse clothing categories.

3) Image Representations and Annotations

The DressCode dataset stands out for its rich and detailed annotations that significantly enhance model training and evaluation:

- Label Maps: 18 semantic categories (e.g., background: 0, upper clothes: 4, dress: 7) for pixel-wise segmentation, enabling precise garment localization and boundary identification
- Key points: 18 body joints (e.g., shoulders, elbows, knees) in JSON format, with missing points marked as [-1, -1, 0.0, -1], facilitating accurate pose estimation and alignment

- Skeletons: Visual pose representations derived from key points, providing a clear visualization of the human pose
- DensePose: UV maps and label images for 3D body mapping, offering dense correspondence between image pixels and 3D surface points
- Captions: Text descriptions (e.g., "red floral dress") from dc_caption.txt, enabling text-guided virtual try-on applications

These comprehensive annotations make DressCode particularly valuable for developing advanced virtual try-on systems that can handle multiple garment categories while maintaining high visual fidelity and realistic garment deformation.

4) Dataset Processing Challenges

Working with the Dress Code dataset presented several significant challenges:

- **Large Dataset Size**: The complete dataset was extremely large (76.13GB), making it difficult to upload to Kaggle.
- **Initial Solution Attempt**: We first tried using Wasabi Cloud's free tier for storage but encountered difficulties transferring the data to Kaggle.
- **Final Solution**: We worked directly with the dataset owners, who provided specialized links allowing us to download the dataset directly to Kaggle's environment.
- **Usage Restrictions**: The dataset required submitting a formal request form and signing an agreement restricting its use to educational purposes only, with no commercial applications or redistribution permitted.
- **Memory Constraints**: The high-resolution images (1024×768) and extensive annotations placed considerable demands on the Kaggle P100 GPU's 16GB memory, requiring efficient data handling strategies
- **Mask Creation**: Generating accurate agnostic masks (areas to be replaced by new garments) required intricate processing of label maps

and key points, including complex operations like dilation, body part parsing, and neckline detection.

We addressed these challenges through careful subset sampling, comprehensive memory optimization techniques, and robust preprocessing pipelines, as detailed in subsequent sections.

We enhanced the IDM-VTON model through a category-specific fine-tuning approach, memory-efficient techniques, and thorough evaluation.

B. Model Implementation and Fine-Tuning

1) UNet Architecture and Fine-Tuning Strategy

UNet Architecture in IDM-VTON

The IDM-VTON model utilizes a sophisticated UNet architecture as its core generative component. Based on the Stable Diffusion XL (SDXL) inpainting model, this UNet serves as the primary mechanism for synthesizing high-quality virtual try-on images. According to the IDM-VTON paper [1], the model employs a multi-component architecture where the UNet (called TryonNet) works in conjunction with specialized modules to preserve garment details while generating authentic try-on results

The UNet architecture consists of:

- An encoder path that progressively downsamples feature maps
- A decoder path that gradually restores spatial resolution
- Skip connections that preserve spatial information across different scales
- Attention mechanisms (both self-attention and cross-attention) that enable long-range dependencies

2) Multi-Category Fine-Tuning Approach

Our key contribution is the extension of the IDM-VTON model to handle multiple garment categories through category-specific fine-tuning. While the original model was primarily designed for upper-body garments, we systematically fine-tuned separate models for upper_body, lower_body, and dresses categories from the DressCode dataset. This multi-category approach enables a single model architecture to successfully handle diverse garment types with their unique characteristics - from t-shirts and blouses to pants, skirts, and full dresses. Each category presents distinct challenges: upper-body garments require attention to sleeve details and necklines, lower-body garments need proper alignment with the waist and legs, and dresses must maintain visual coherence across the entire body.

Our implementation focuses exclusively on fine-tuning the UNet component of the IDM-VTON model while keeping all other components frozen. The UNet is the core generative component responsible for synthesizing the try-on result, making it the most critical element for improving visual quality.

Importance of UNet Fine-Tuning

Our implementation focuses exclusively on fine-tuning the **UNet component** of the IDM-VTON model while keeping all other components frozen. The UNet is the core generative component responsible for synthesizing the try-on result, making it the most critical element for improving visual quality.

Key aspects of our training approach include:

- **Selective Parameter Updating**: Only the UNet parameters were unfrozen and updated during training, significantly reducing the number of trainable parameters and memory requirements.
- **Focused Learning**: The UNet is responsible for the actual denoising process in the diffusion model, making it the most critical component for improving visual quality, garment draping, and boundary handling.
- **Preservation of Pretrained Knowledge**: Other model components (VAE, text encoders, image encoder) remained frozen in evaluation mode, preserving their pretrained knowledge while allowing the UNet to adapt to the specific characteristics of the DressCode dataset.

This focused approach allowed us to efficiently improve try-on quality without the computational cost of fine-tuning the entire model. By targeting only the UNet, we achieved better garment draping, texture preservation, and boundary handling while maintaining the overall stability of the diffusion process.

3) Training and Inference Strategy

We fine-tuned the IDM-VTON model on **3,000 samples per category** (upper body, lower body, dresses) to capture category-specific features. Each category was fine-tuned separately using distinct notebooks, with **14 epochs completed per category**. To ensure that model act well in each garment.

Our training approach incorporated several techniques to optimize the UNet's performance:

- **Gradient Checkpointing**: We enabled gradient checkpointing for the UNet, a technique that trades computation for memory by recomputing intermediate activations during the backward pass instead of storing them.

- **Progressive Resolution Strategy:**

  We implemented a multi-stage resolution scaling approach (detailed in Section X.X) to balance memory usage and output quality.

- **Early Stopping:** Training stopped if validation loss plateaued for 3 epochs (patience=3).

- **Checkpoints:** Saved per epoch to HuggingFace Hub (e.g., checkpoint_epoch_14.pt for dresses and Lower and Upper**)**

4) Training Progression Visualization

These are examples during training on validation dataset that express about how model learning in epochs.

Resolution Progression Through Training Epochs

Epochs	Scale Factor	Resolution (Height × Width)	Percentage of Full Resolution
0-1	3.0	168 × 128	~33%
1-3	2.2	232 × 176	~45%
3-5	1.7	304 × 224	~59%
5+	1.0	512 × 384	100%

Importance of Progressive Resolution Scaling

This approach offers several critical benefits for model training:

1. **Memory Efficiency**: Starting with lower resolutions dramatically reduces memory requirements in early epochs, allowing training on GPUs with limited VRAM (like Kaggle's P100 with 16GB)
2. **Training Stability**: Lower resolutions provide a more stable learning environment in early epochs, helping the model learn coarse features before fine details
3. **Faster Convergence**: The model can quickly learn overall garment structure and placement at lower resolutions before focusing on details
4. **Curriculum Learning**: This implements a form of curriculum learning where the task gradually increases in difficulty, improving overall performance
5. **Computational Speed**: Early epochs complete much faster, providing quicker feedback on training progress

This technique was essential for successfully training on Kaggle's constrained environment while still achieving high-quality results in the final model.

Here are some examples during training showing model learning throughout training on different categories:

1. Dresses

Early epochs

Figure 1

Figure 2

Middle epochs

Figure 3

Figure 4

Late epochs

Figure 5

Figure 6

2. Upper body

Early epochs

Figure 7

Figure 8

Middle epochs

Figure 9

Figure 10

Late epochs

Figure 11

Figure 12

3. Lower body

Early epochs

Figure 13

Figure 14

Middle epochs

Figure 15

Figure 16

Late epochs

Figure 17

Figure 18

5) Memory Optimization Techniques

To operate within Kaggle's 16GB GPU memory:

- **VAE Slicing/Tiling**: Processed large images in chunks.
- **Attention Slicing**: Reduced memory usage in attention layers.
- **Mixed Precision**: Used FP16 to halve memory usage.

These techniques cut memory consumption by approximately 40%.

C. Experimental Results

1) Qualitative Evaluation

Loss Functions:

We trained the UNet using a combination of perceptual losses:

- LPIPS for structural similarity
- SSIM for image quality
- L1 loss for pixel-level accuracy
- CLIP image similarity (weighted 2.0×) for semantic consistency
- Evaluation Metrics

Our fine-tuned UNet achieved impressive results across multiple metrics:

Clothing Category Performance Metrics

Category	LPIPS (↓)	SSIM (↑)	CLIP-IS (↑)	FID (↓)
Upper Body	0.02828	0.9496	0.9804	4.022
Lower Body	0.03110	0.9480	0.9818	3.706
Dresses	0.06687	0.8907	0.9770	4.488

Legend:

- ↓ = Lower is better
- ↑ = Higher is better

These metrics demonstrate the effectiveness of our UNet fine-tuning approach, with particularly strong performance in the upper and lower body categories.

Metric Definitions:

- **LPIPS (Learned Perceptual Image Patch Similarity)**: Measures perceptual similarity between images using deep features, with lower values indicating better similarity.
- **SSIM (Structural Similarity Index)**: Evaluates the perceived quality of images based on structural information, with higher values indicating better quality.
- **CLIP-IS (CLIP Image Similarity)**: Measures semantic similarity between generated and target images using CLIP embeddings, with higher values indicating better semantic consistency.
- **FID (Fréchet Inception Distance)**: Evaluates the quality and diversity of generated images compared to real images, with lower values indicating better performance.

By focusing our fine-tuning efforts specifically on the UNet component with these specialized techniques, we were able to achieve high-quality virtual try-on results while maintaining computational efficiency within Kaggle's constraints

2) Quantitative Performance Analysis

To visually assess our model's performance, we created side-by-side comparisons between original images and our model-generated try-on results. As shown in Figure X, our model successfully preserves intricate garment details while realistically adapting to the target person's pose and body shape.

Side-by-side comparison images showing:

1. **Upper body:** Comparison of original garment (left) and model-generated try-on result (right) for upper body category

Figure 1

Figure 2

Figure 3

2. Dresses: Comparison of original garment (left) and model-generated try-on result (right) for dresses category

Figure 4

Figure 5

Figure 6

Figure 7

3. **Lower Body:** Comparison of original garment (left) and model-generated try-on result (right) for lower body category

Figure 8

Figure 9

Figure 10

Figure 11

Figure 12

These visual comparisons demonstrate our model's ability to handle different garment types, maintain texture fidelity, and properly align garments to diverse body poses. Particularly notable is the model's performance on complex patterns and graphics, which previous approaches struggled to preserve.

D. Implementation Challenges and Solutions

Hardware Constraints

The Kaggle P100 GPU's limitations are:

- **Solutions**: Reduced inference steps (15 for training), dynamically scaled resolutions, and frequently cleared CUDA cache and reduce Batch size.

E. Deployment

3.5.1 Environment Setup

Deploying the **IDM-VTON** model for virtual try-on requires a user-friendly interface and scalable infrastructure. This guide covers:

- **Kaggle**: Hosting the model and datasets.
- **Gradio**: Building a web demo for real-time try-on.

Why Kaggle + Gradio?

- **Kaggle**: Free GPU access (T4/P100), pre-installed libraries (PyTorch, Diffusers).
- **Gradio**: Simple Python API for web UIs, supports image uploads/outputs.

Steps:

- Setting up the environment
  - downloading needed libraries
  - do our imports
- Uploading the model
- Setting the model in inference mode
- Create our API functions
- Create our API
- Integrate it with our interface (web page in our case)

Chapter 4: Diagrams

Diagrams play a vital role in the software development process by visually representing different aspects of a system's structure, behavior, and data flow. They enhance understanding among team members, stakeholders, and developers by simplifying complex systems into manageable visual components. In our project, we utilized a variety of diagrams, each serving a specific purpose in illustrating the system's design

1. Functional C Non Functional
2. Usecase Diagram
3. Flowchart
4. Activity Diagram
5. Sequence Diagram
6. Block Diagram
7. Class Diagram
8. Entity Relation Diagram (ERD)
9. Entity Diagram
10. Collaboration Diagram

a. Requirement

a. Functional Requirement

These describe **what** the system should do—the core features and behaviors visible in the diagram.

Type	Functional Requirement
User Registration	The system should allow new users to register by providing necessary details.
User Login	Users (customers and admins) shall be able to log in with valid credentials
Password Recovery	Users shall be able to retrieve forgotten passwords via a password recovery process.
Product Search	Customers shall be able to search for products.
Cart Management	• Customers shall be able to place orders. • Customers shall be able to cancel orders. • Customers shall be able to view their past orders.
Account Management	Customers shall be able to edit their account information.
Product Management (Admin)	Admins shall be able to: ○ Add new products. ○ View products. ○ Update product details. ○ Delete products.
Logout	Users shall be able to log out securely.
Try-On Feature	Customers shall have the option to virtually try on products (e.g., in fashion apps).

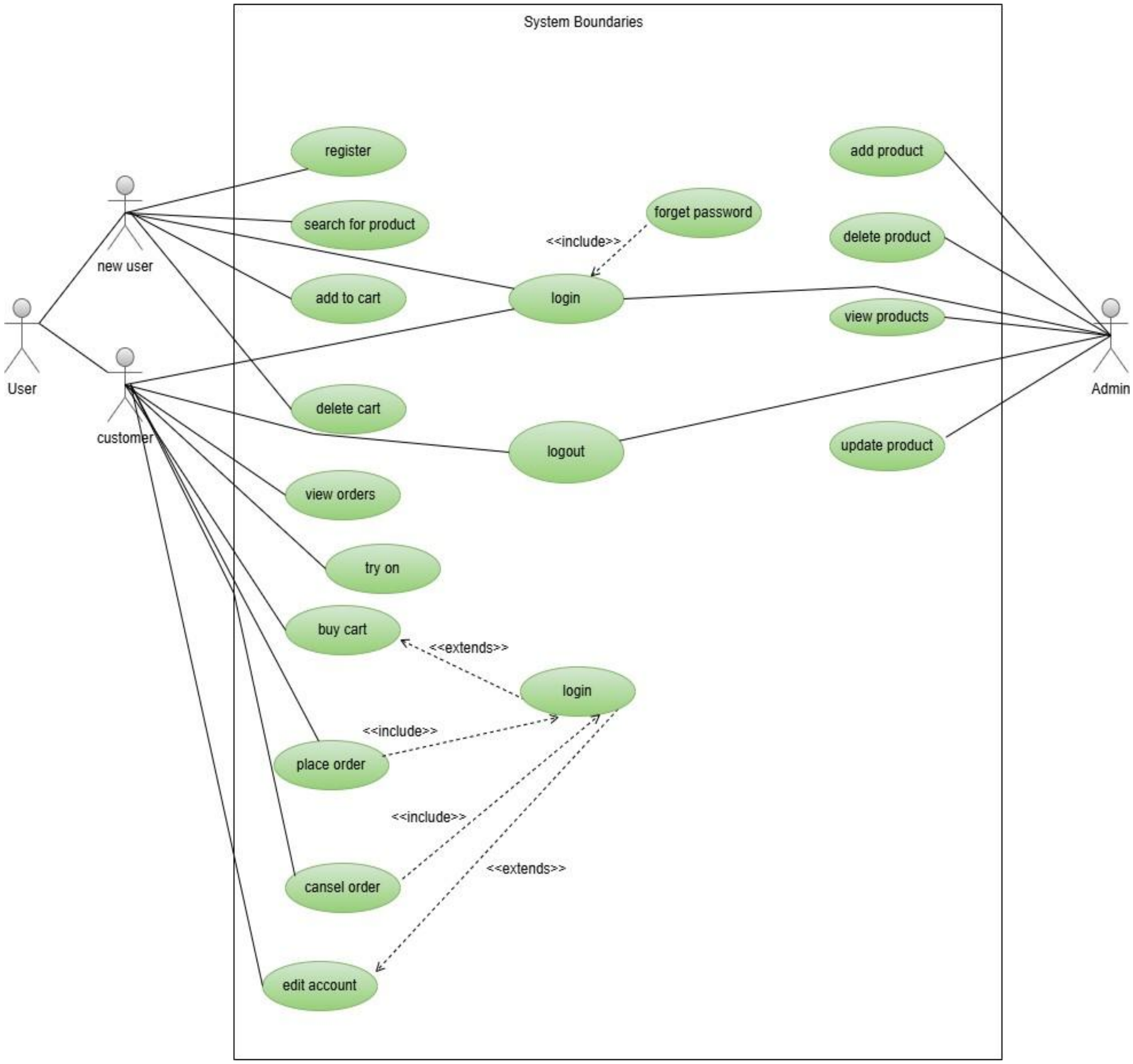
b. Non-Functional Requirement

These describe **how** the system should perform—quality attributes that are not explicitly shown in the diagram but are critical for a well-functioning system.

Type	Non-Functional Requirement
Performance	The system should respond to user actions within 2 seconds.
Scalability	It should support at least 10,000 concurrent users.
Usability	The user interface should be easy to use and intuitive & easy to navigate for both customers and admins.
Security	1-Passwords must be encrypted and stored securely. 2-Role-based access control shall be enforced (e.g., only admins can manage products).
Authorization	Only authorized users should access admin features.
Availability	System availability must be at least 99.9% of the time (uptime).
Maintainability	The codebase should be modular to allow easy updates and bug fixes.
Compatibility	The system should be compatible with major web browsers (Chrome, Firefox, Safari, Edge).
Data Integrity	The system should ensure data consistency and prevent data loss during operations like order placement or cart deletion.

b. Usecase

Identifies the system's actors and their interactions with different functionalities, providing a high-level overview of system requirements.



Category	Details
System Name	Try On system (try & buy)
Actors	- **New User**: Not yet registered - **Customer**: Registered user - **Admin**: System manager
Use Cases (New User)	- Register - Search for Product - Add to Cart - Login
Use Cases (Customer)	- Login - Add/Delete Cart - Buy Cart - Place Order - Cancel Order - View Orders - Edit Account - Try On - Logout
Use Cases (Admin)	- Add Product - View Products - Update Product - Delete Product
Include Relationships	- Forget Password includes Login - Place Order includes Login - Cancel Order includes Login

a. Register

Actor	User
Description	Create a new account
Precondition	Not logged in
Postcondition	1-Account created 2-Login
Main successes scenario (the flow steps)	1- Enter your information 2- Validate the information 3- Create account 4- Save to database
Extension	1- Invalid date 2- Show error

b. Login

Actor	User
Description	Authenticate user
Precondition	Registered
Postcondition	User logged in Go to profile
Main successes scenario (the flow steps)	1- Enter (email, password) 2- Check credentials 3- If valid 4- Go to the profile
Extension	1- Wrong credentials 2- Show errors

c. Try On

Actor	User
Description	Try product (visually)
Precondition	Logged in
Postcondition	The items will display on user
Main successes scenario (the flow steps)	1- Logged in 2- Show products 3- Enter try on 4- Enter your photo 5- Display the photo after processing
Extension	Not good photo

d. Search

Actor	User
Description	Find products
Precondition	logged in
Postcondition	Display products
Main successes scenario (the flow steps)	1- Logged in 2- Search for product 3- Display the product that match the search
Extension	None

e. Add To Cart

Actor	User
Description	Add product to cart
Precondition	logged in
Postcondition	Product added to cart
Main successes scenario (the flow steps)	1- Logged in 2- Search for product 3- Add product to cart 4- Update cart
Extension	1- Out of stock 2- Show error

f. Update Profile

Actor	User
Description	Update profile info
Precondition	Logged in
Postcondition	Profile updated
Main successes scenario (the flow steps)	1- Logged in 2- Go to edit profile 3- Enter your new data 4- Enter to update 5- Save data to database
Extension	Invalid input

g. Buy Cart

Actor	User
Description	Purchase all products(buy cart)
Precondition	Logged in Product in cart
Postcondition	Order placed
Main successes scenario (the flow steps)	1- Logged in 2- Show products 3- Enter product to (add to cart) 4- Finish shopping 5- Go to cart 6- Enter buy order 7- Process payment
Extension	Payment fails

h. Place Order

Actor	User
Description	Confirm purchase
Precondition	Logged in Cart has items
Postcondition	Order confirmed
Main successes scenario (the flow steps)	1- Logged in 2- Go to cart 3- Enter place order 4- Process payment
Extension	Payment fails

i. Cancel Order

Actor	User
Description	Cancel placed order
Precondition	Logged in
Postcondition	Order canceled
Main successes scenario (the flow steps)	1- Logged in 2- Go to cart 3- Enter cancel order 4- Update status
Extension	Already shipped

j. Delete Cart

Actor	User
Description	Remove all product from cart
Precondition	1- Logged in 2- Have cart 3- Product in cart
Postcondition	Cart is empty
Main successes scenario (the flow steps)	1- Logged in 2- Go to cart 3- Enter delete the cart 4- Clear cart
Extension	None

k. View Orders

Actor	User
Description	View past orders
Precondition	Logged in
Postcondition	Orders display
Main successes scenario (the flow steps)	1- Logged in 2- Go to profile, cart 3- Show orders
Extension	None

l. Reset Password

Actor	User
Description	Reset password
Precondition	None
Postcondition	Password reset link sent
Main successes scenario (the flow steps)	1- Enter (email) 2- Go to reset password page 3- Enter the new password 4- System send link
Extension	1- Email not found 2- Show errors

m. Update Product (Admin)

Actor	Admin
Description	Update products
Precondition	Logged in as Admin
Postcondition	Products updated
Main successes scenario (the flow steps)	1- Logged in 2- Select product 3- Enter edit product 4- Enter the new data 5- Enter edit
Extension	Validation fails

n. Add Product (Admin)

Actor	Admin
Description	Add new products to catalog
Precondition	Logged in as Admin
Postcondition	Products added
Main successes scenario (the flow steps)	1- Logged in 2- Add product button 3- Enter the details of product 4- Enter add
Extension	Invalid data

o. Delete Product (Admin)

Actor	Admin
Description	Remove products
Precondition	Logged in as Admin
Postcondition	Products deleted
Main successes scenario (the flow steps)	1- Logged in 2- Select product 3- Enter delete 4- Enter save
Extension	Product in order history

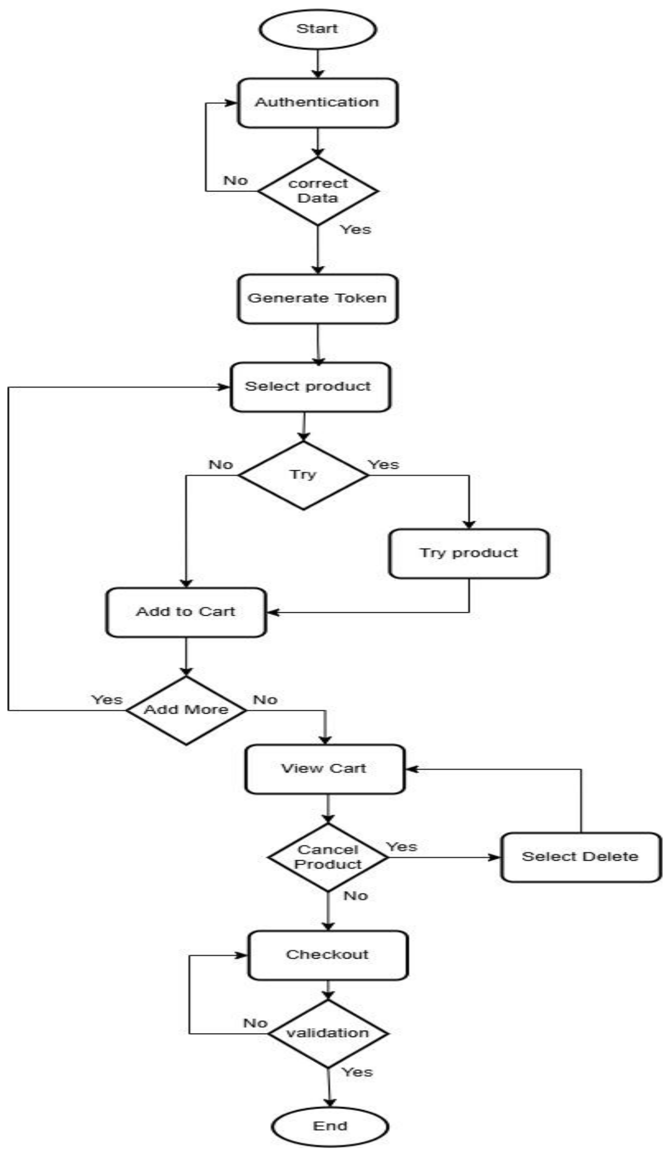
p. View Product (Admin)

Actor	Admin
Description	View products
Precondition	Logged in as Admin
Postcondition	Products displayed
Main successes scenario (the flow steps)	1- Logged in 2- Select category 3- Show product list
Extension	None

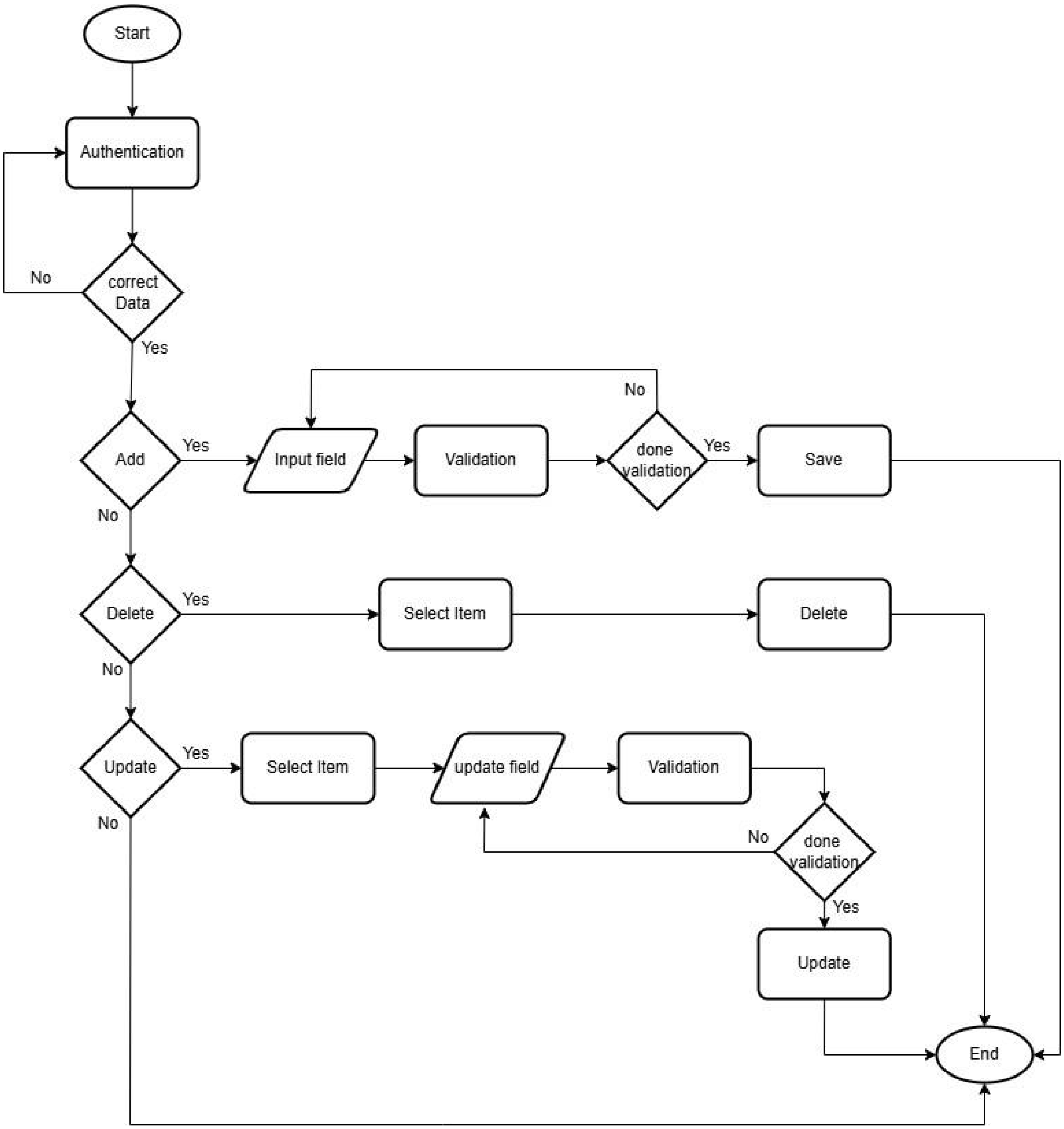
c. Flowchart

A flowchart is a visual representation of a process or workflow, using shapes connected by lines to depict the steps in a sequence. It's a tool used to document, analyze, and improve processes by clearly ou-tlining each step and its relationship to others.

Flowchart For User



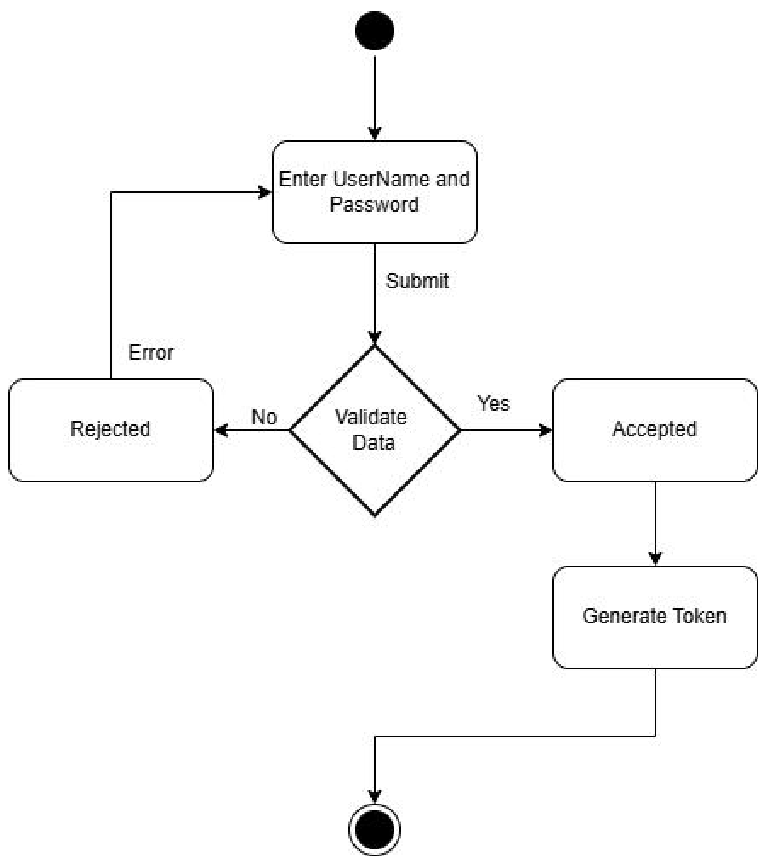
Flowchart For Admin



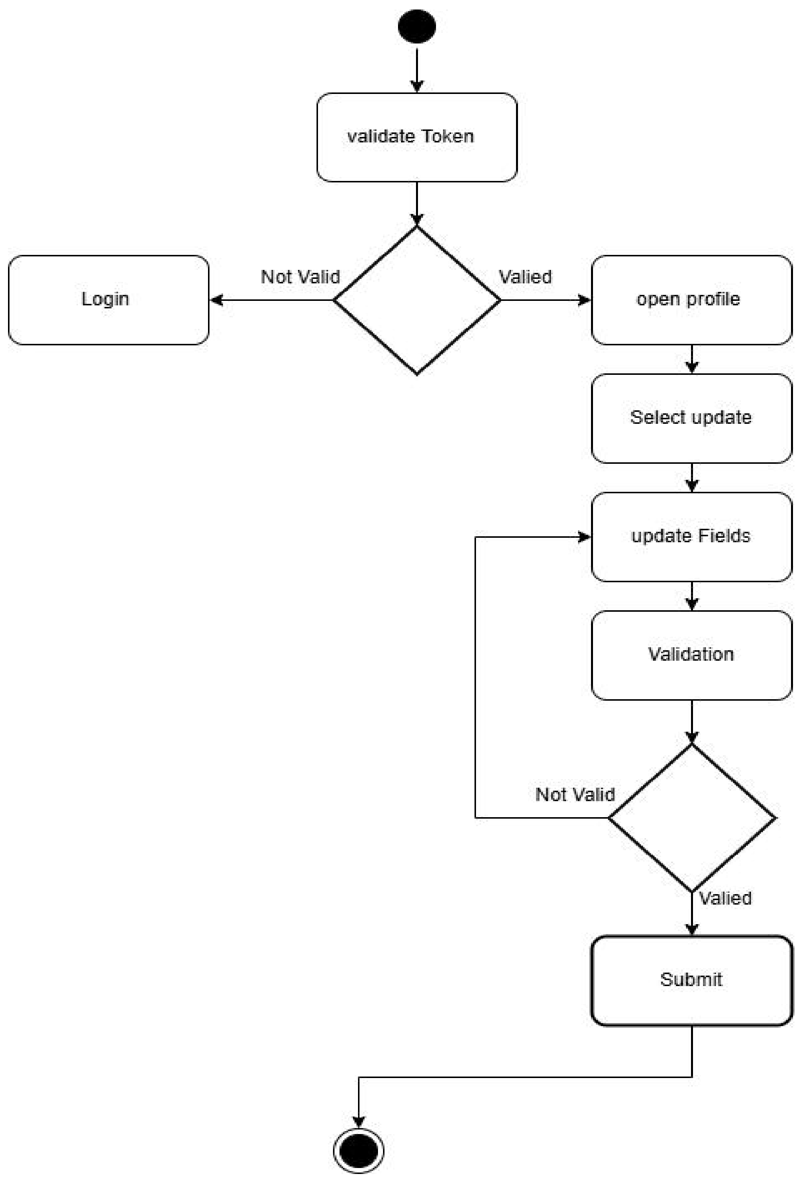
d. Activity Diagram

An activity diagram, in the context of Unified Modeling Language (UML), is a type of behavior diagram that visually represents the flow of actions or operations within a system or process. It's like a flowchart but with more detail and focus on the order and concurrency of activities.

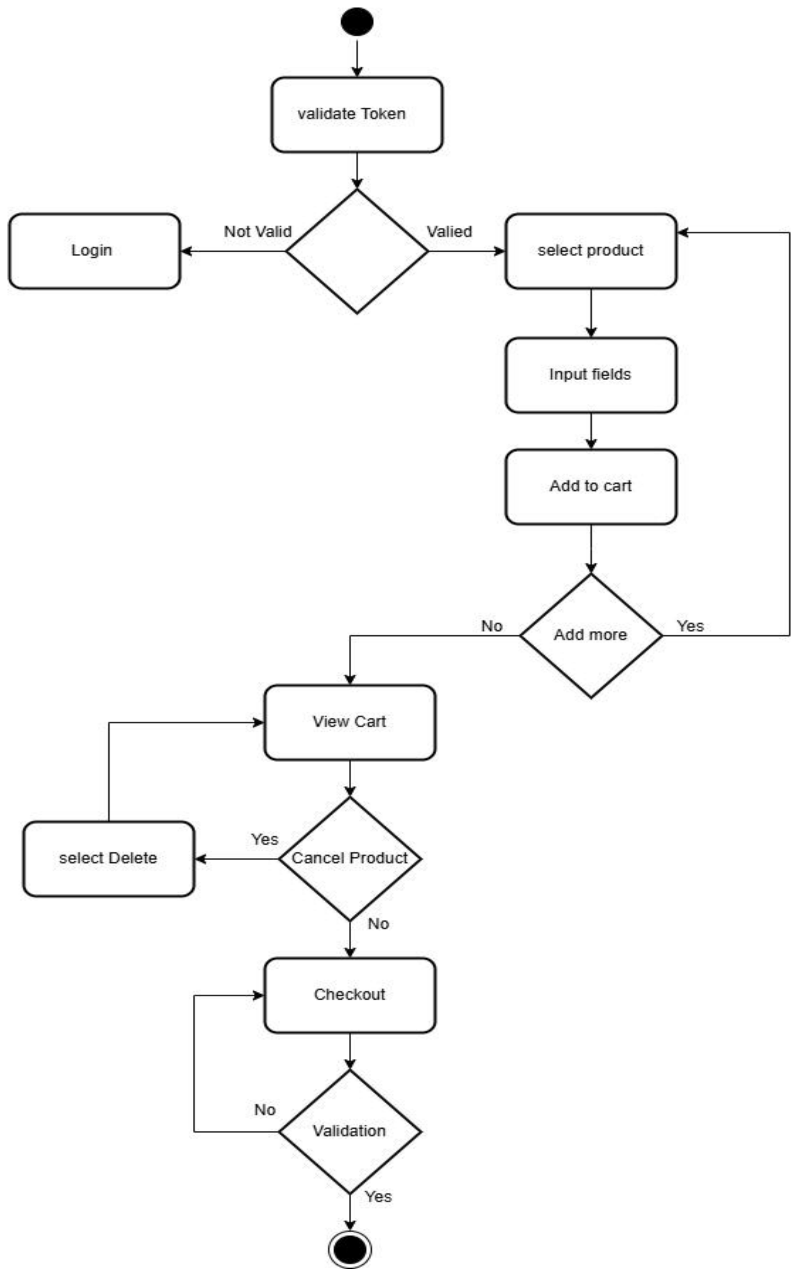
Activity Diagram for Login



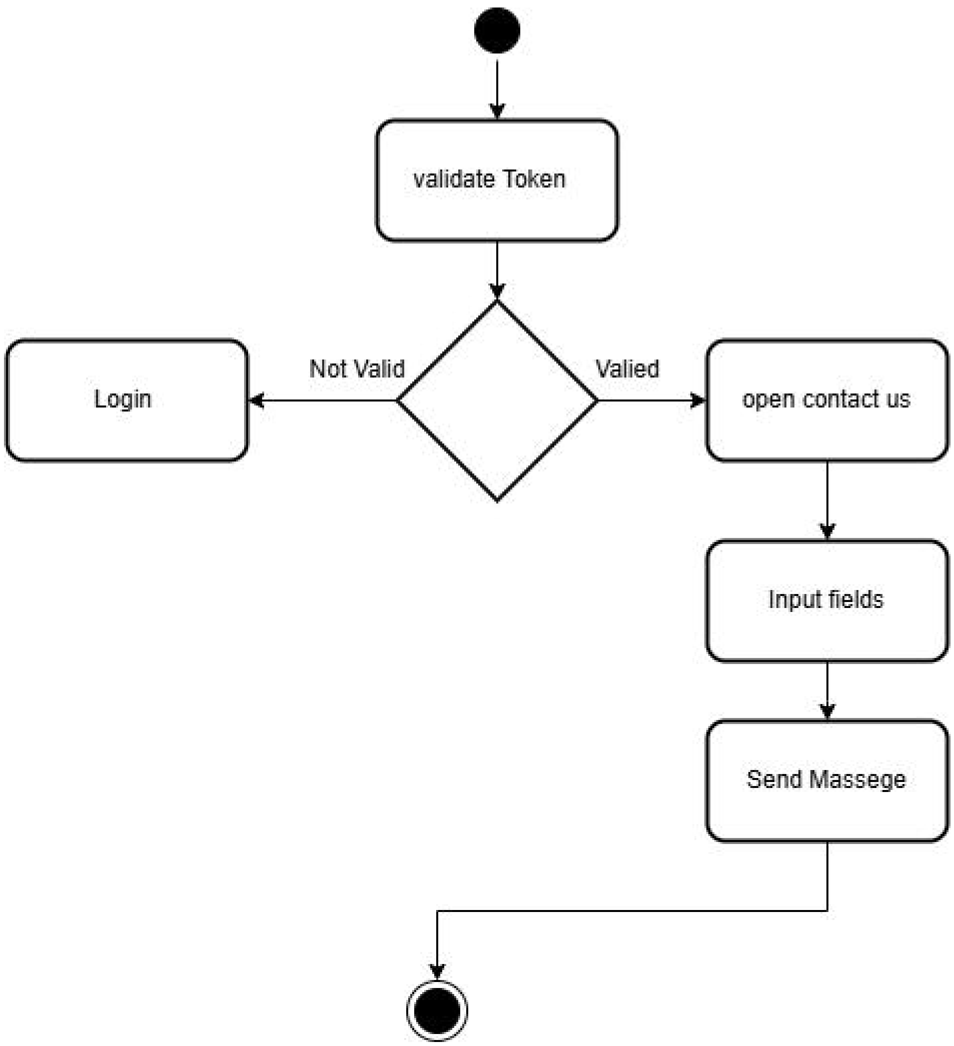
Activity Diagram for Update Profile



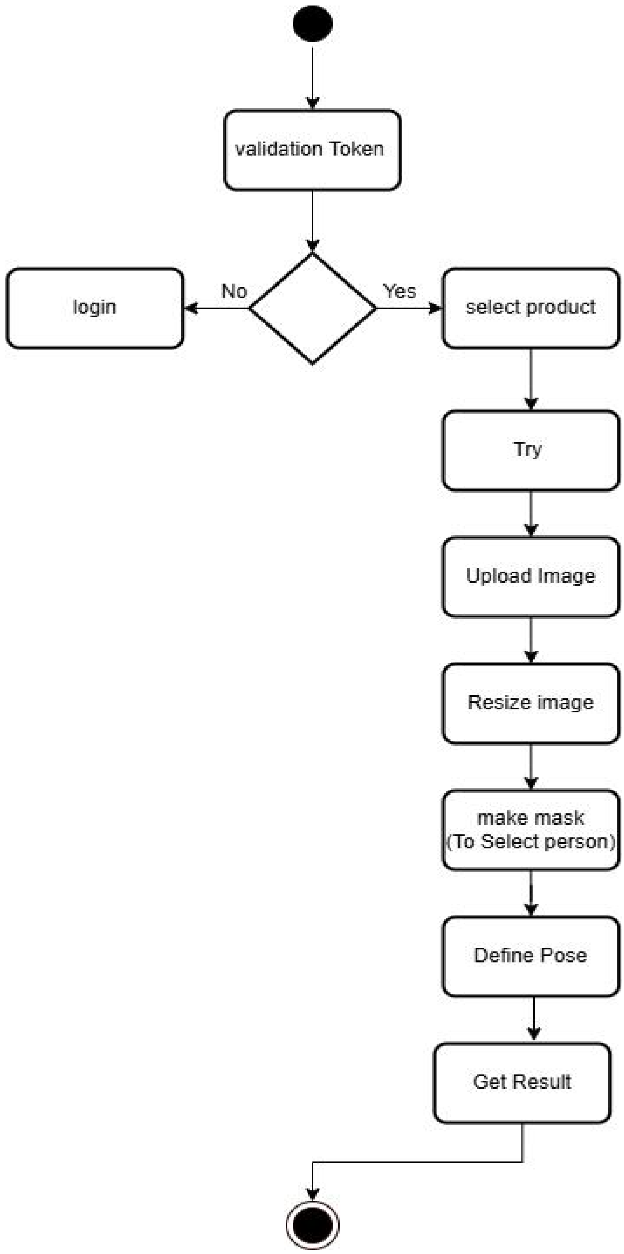
Activity Diagram for Buy Product



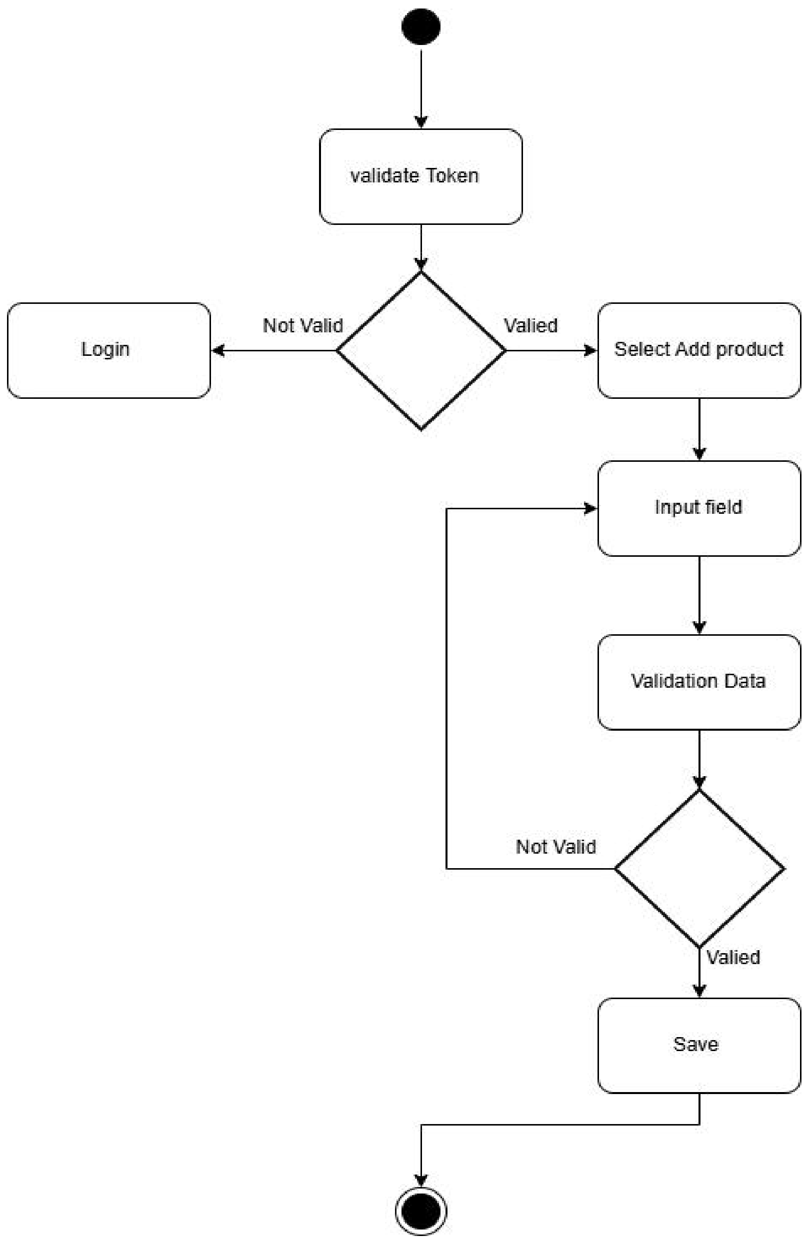
Activity Diagram for Contact-Us



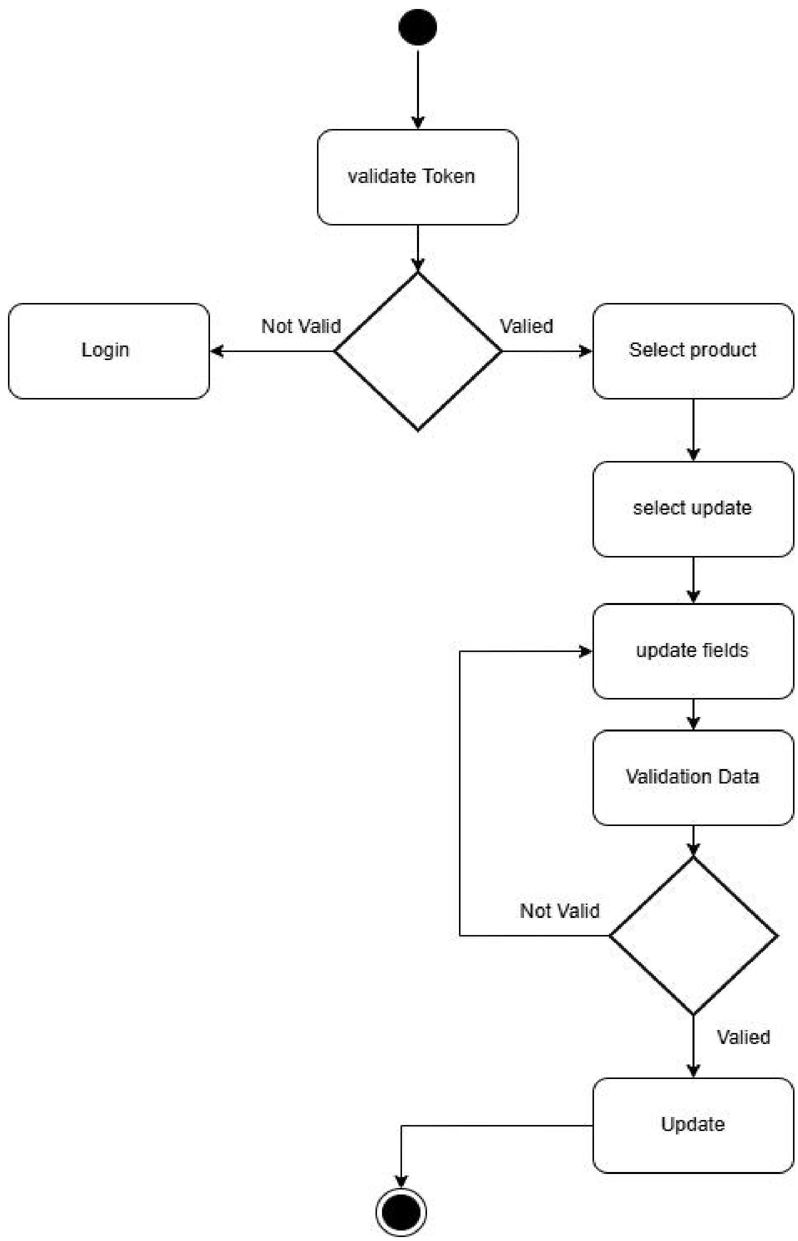
Activity Diagram for Try-On



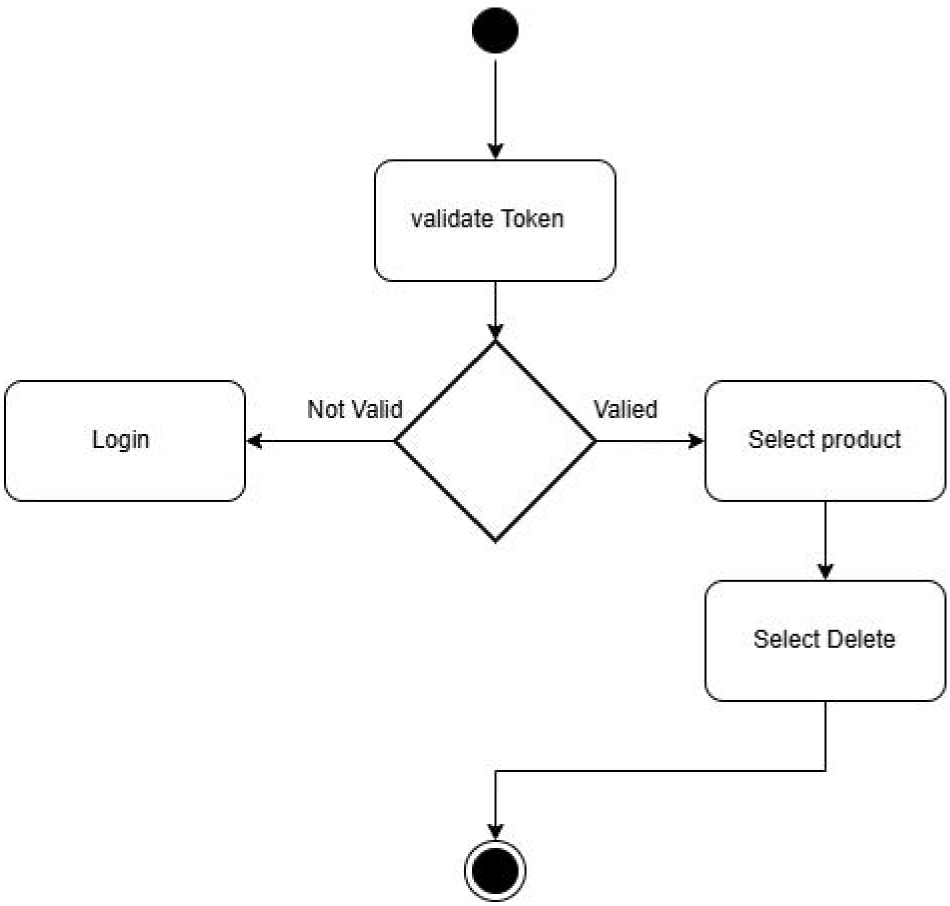
Activity Diagram for Add Product by Admin



Activity Diagram for Update Product by Admin



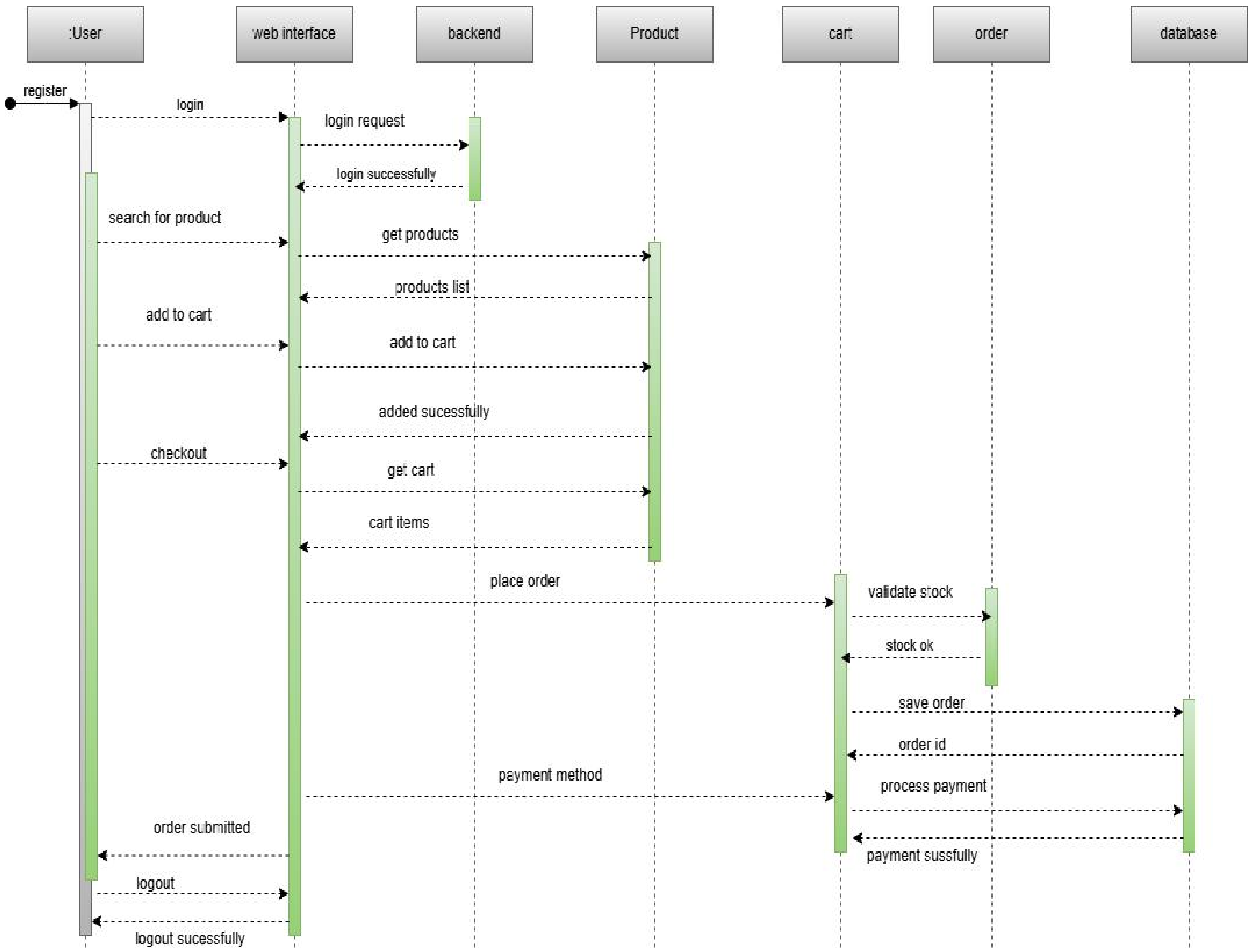
Activity Diagram for Delete Product by Admin



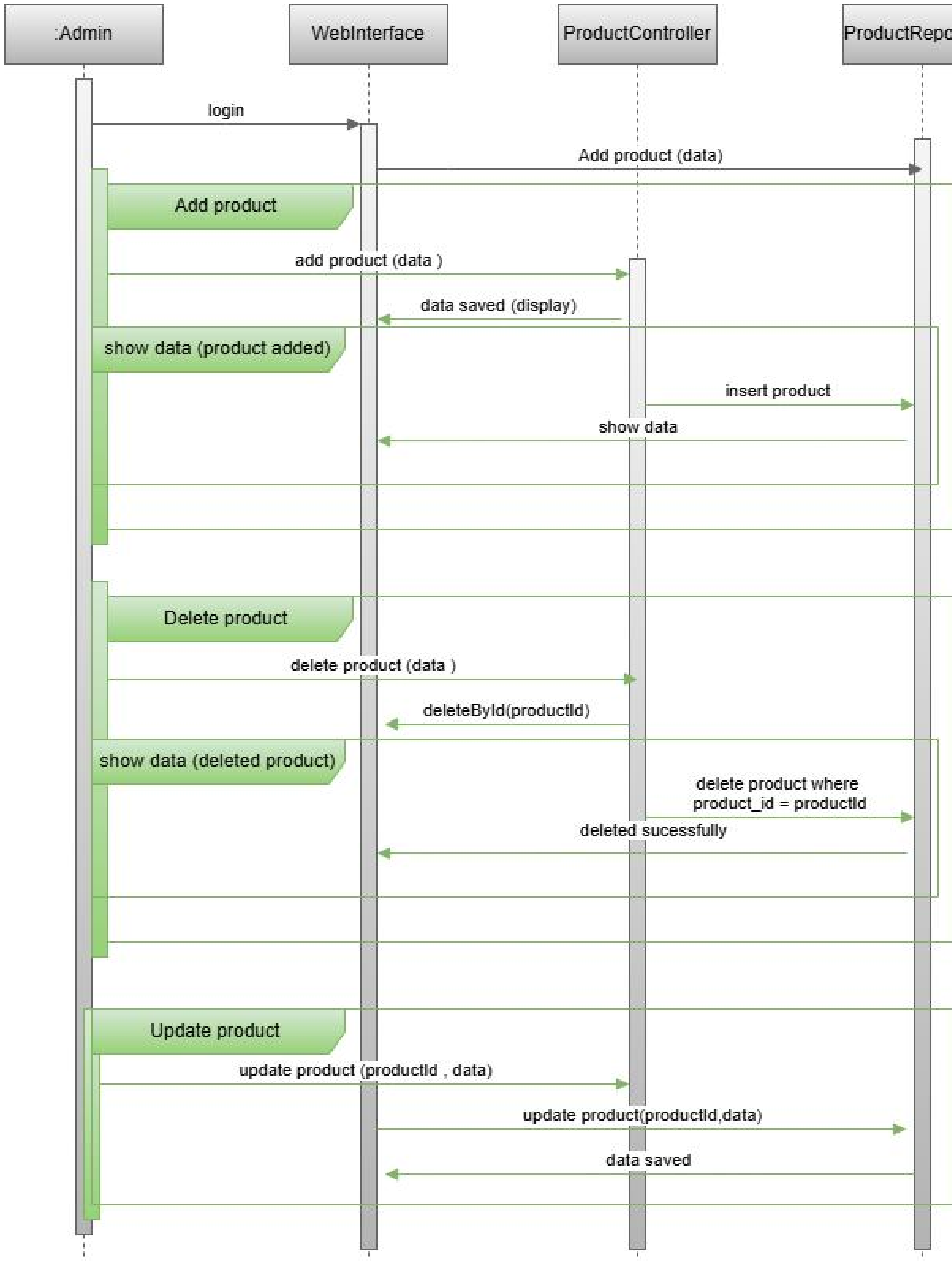
e. Sequence Diagram

Describes the time-ordered interaction between objects to carry out specific functionalities.

- User



- Admin

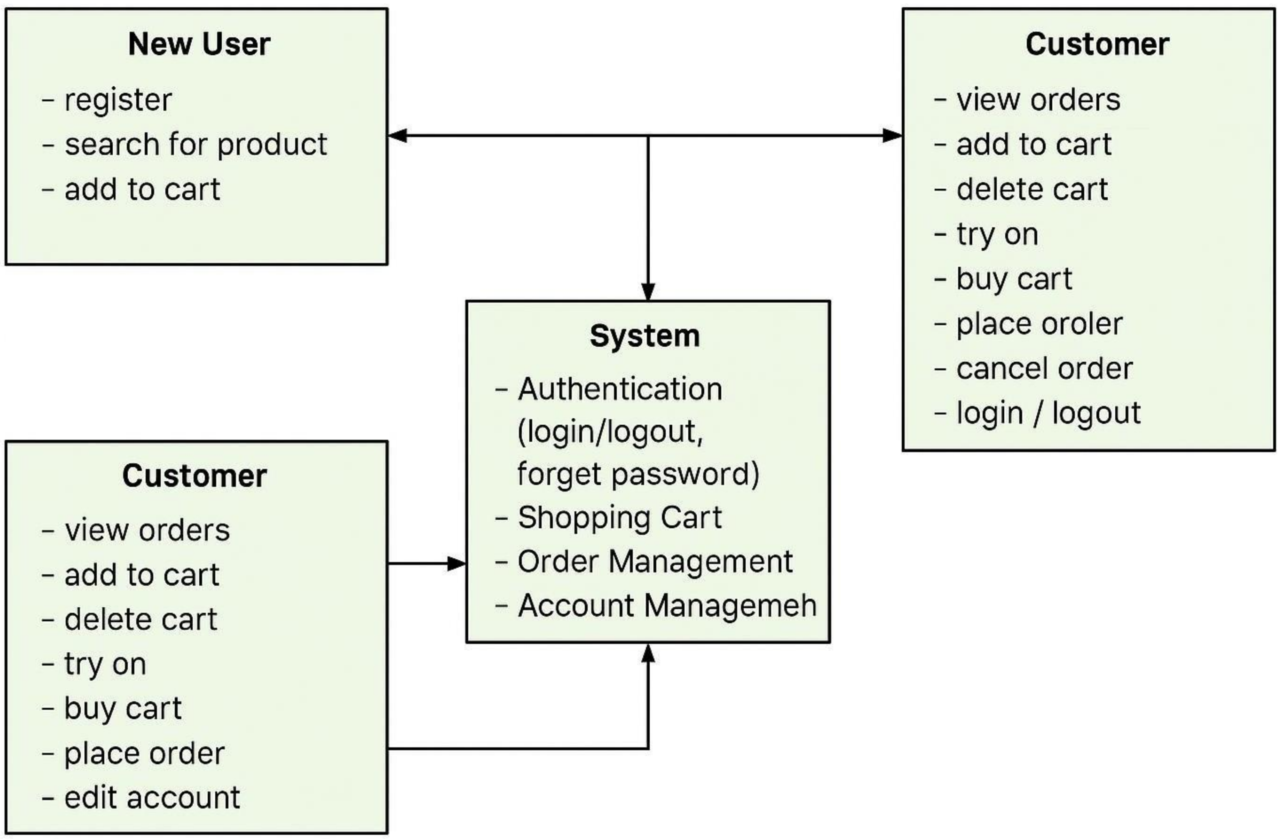


:Admin
WebInterface
ProductController
ProductRepository
login
Add product (data)
Add product
add product (data)
data saved (display)
show data (product added)
insert product
show data
Delete product
delete product (data)
deleteById(productId)
show data (deleted product)
delete product where
product_id = productId
deleted sucessfully
Update product
update product (productId , data)
update product(productId,data)
data saved

f. Block Diagram

Offers a simplified view of the major system components and how they interact.

a. User



New User

A user who has not yet registered in the system.

Allowed actions:

Register: Create a new account.

Search for Product: Browse available items.

Add to Cart: Begin shopping and save items to a temporary cart.

Customer

A registered user who can fully interact with the system.

Allowed actions:

View Orders: See a history of past purchases.

Add to Cart: Continue shopping.

Delete Cart: Remove items from the cart.

Try On: Use the virtual try-on feature.

Buy Cart: Proceed to purchase items in the cart.

Place Order: Finalize an order and confirm purchase.

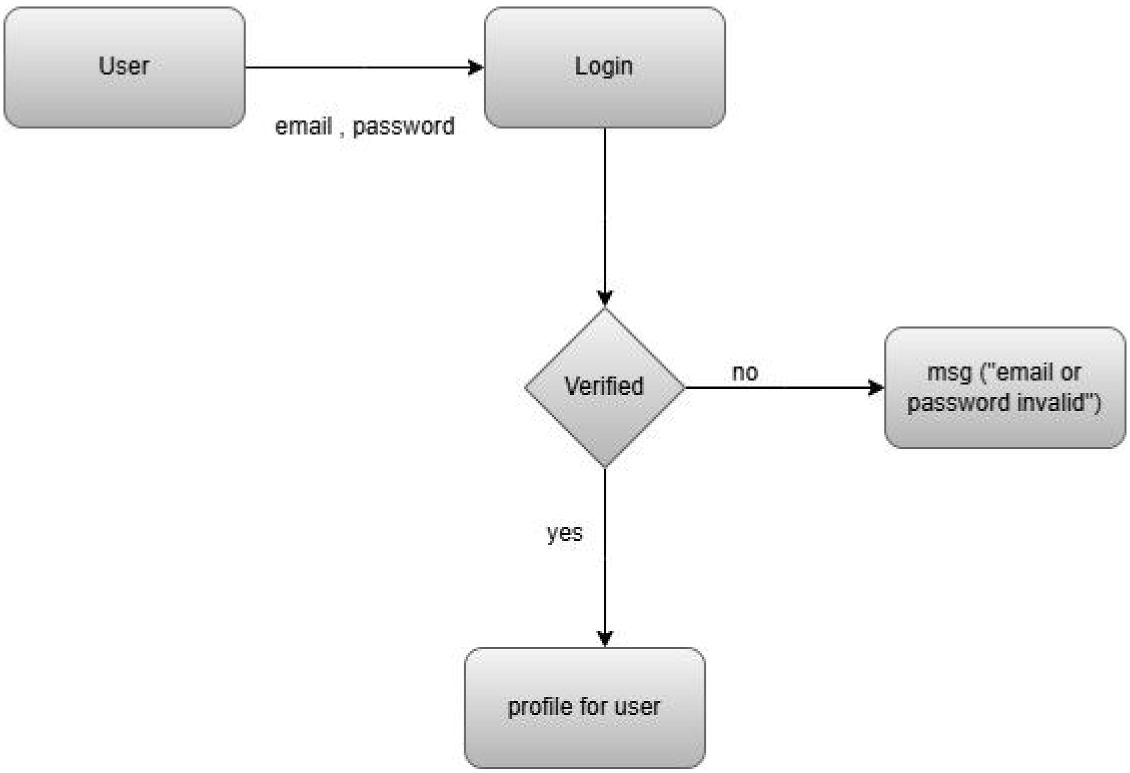
Cancel Order: Withdraw an existing order.

Edit Account: Update personal details.

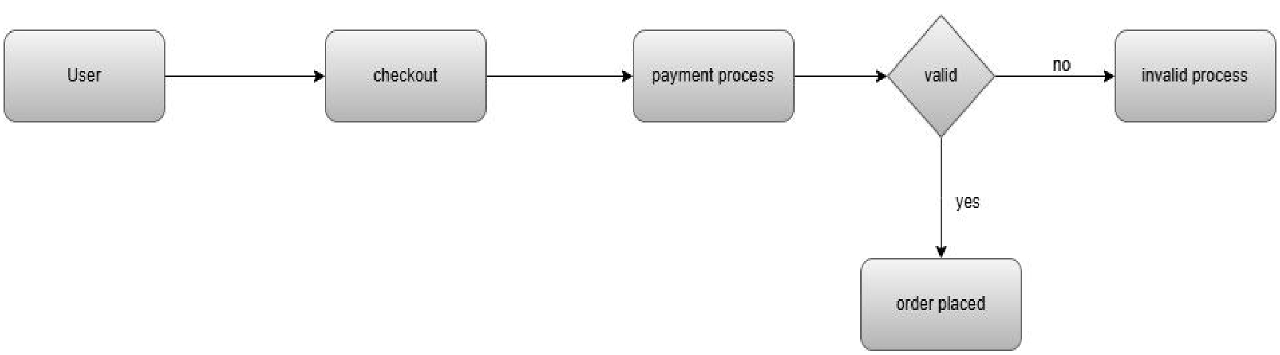
Login / Logout: Secure access to the system.

- *Register*

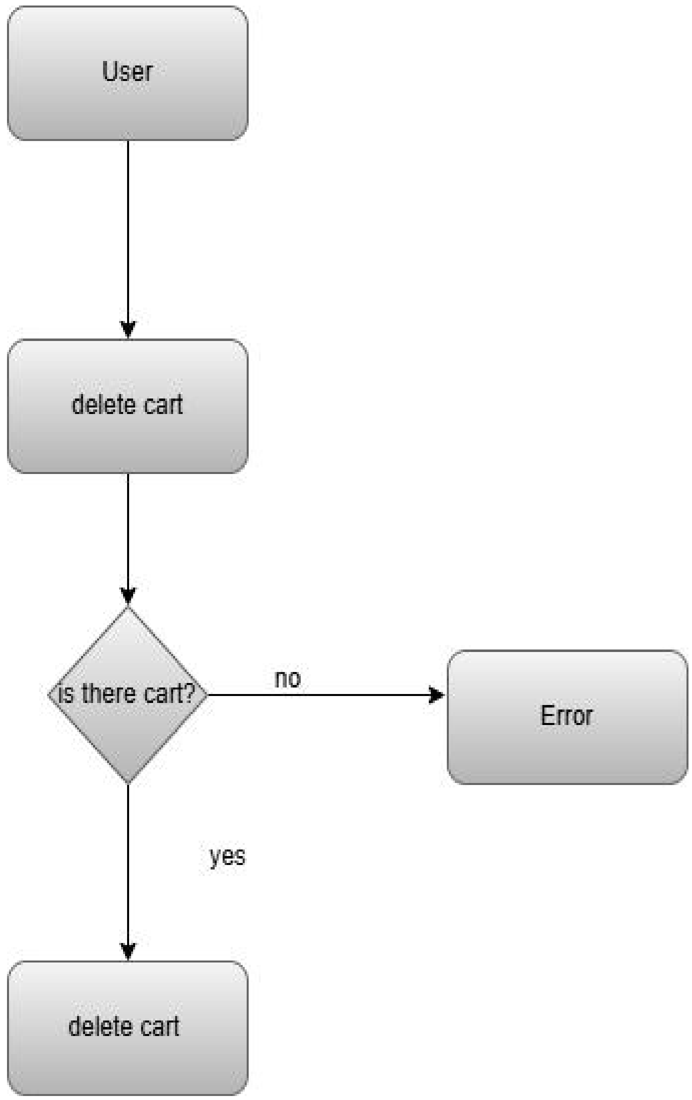
- *Login*



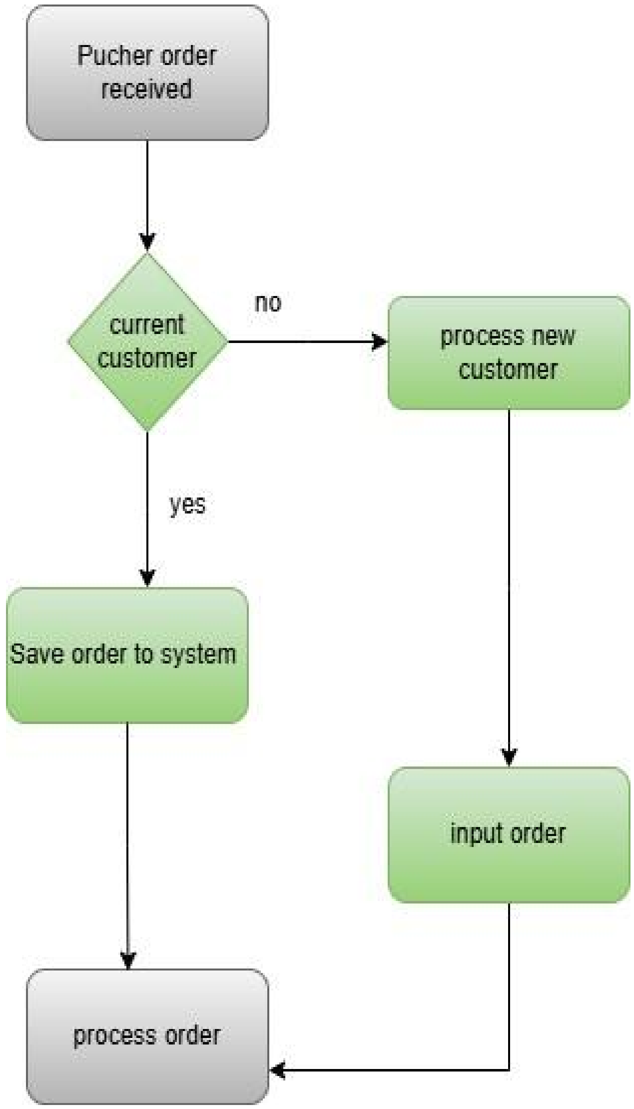
- *Checkout*



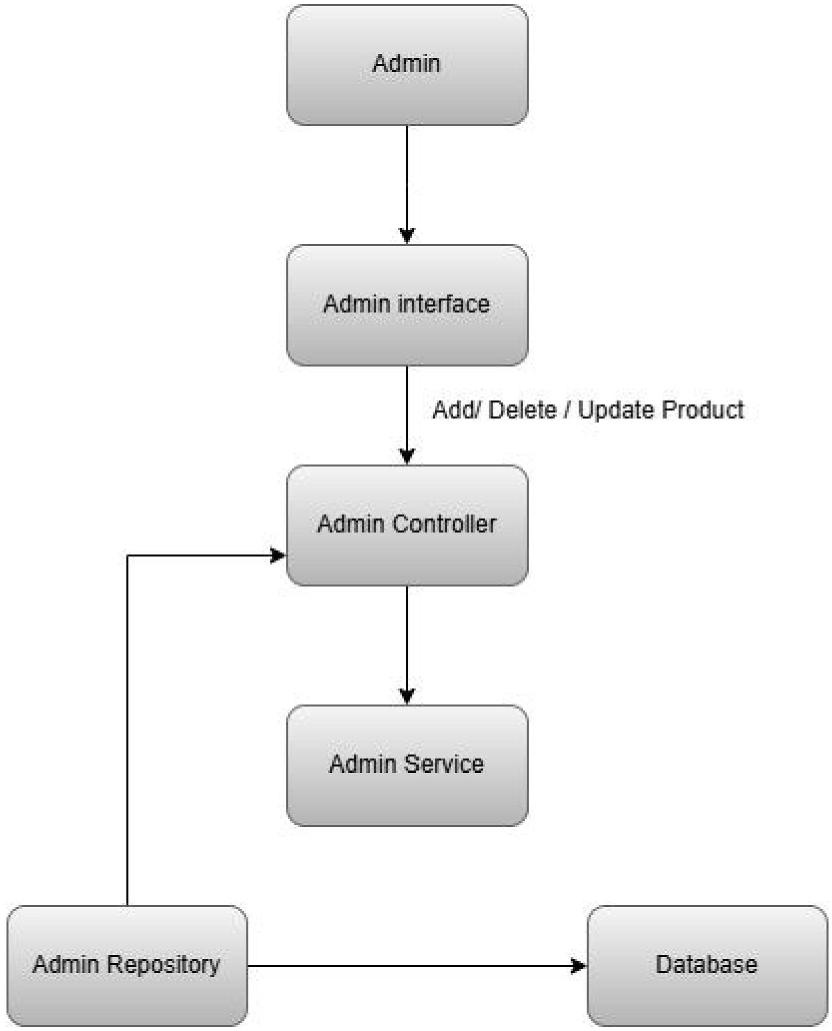
- *Delete Cart*



- *Place Order*


Pucher order received
current customer
no
process new customer
yes
Save order to system
input order
process order

b. Admin

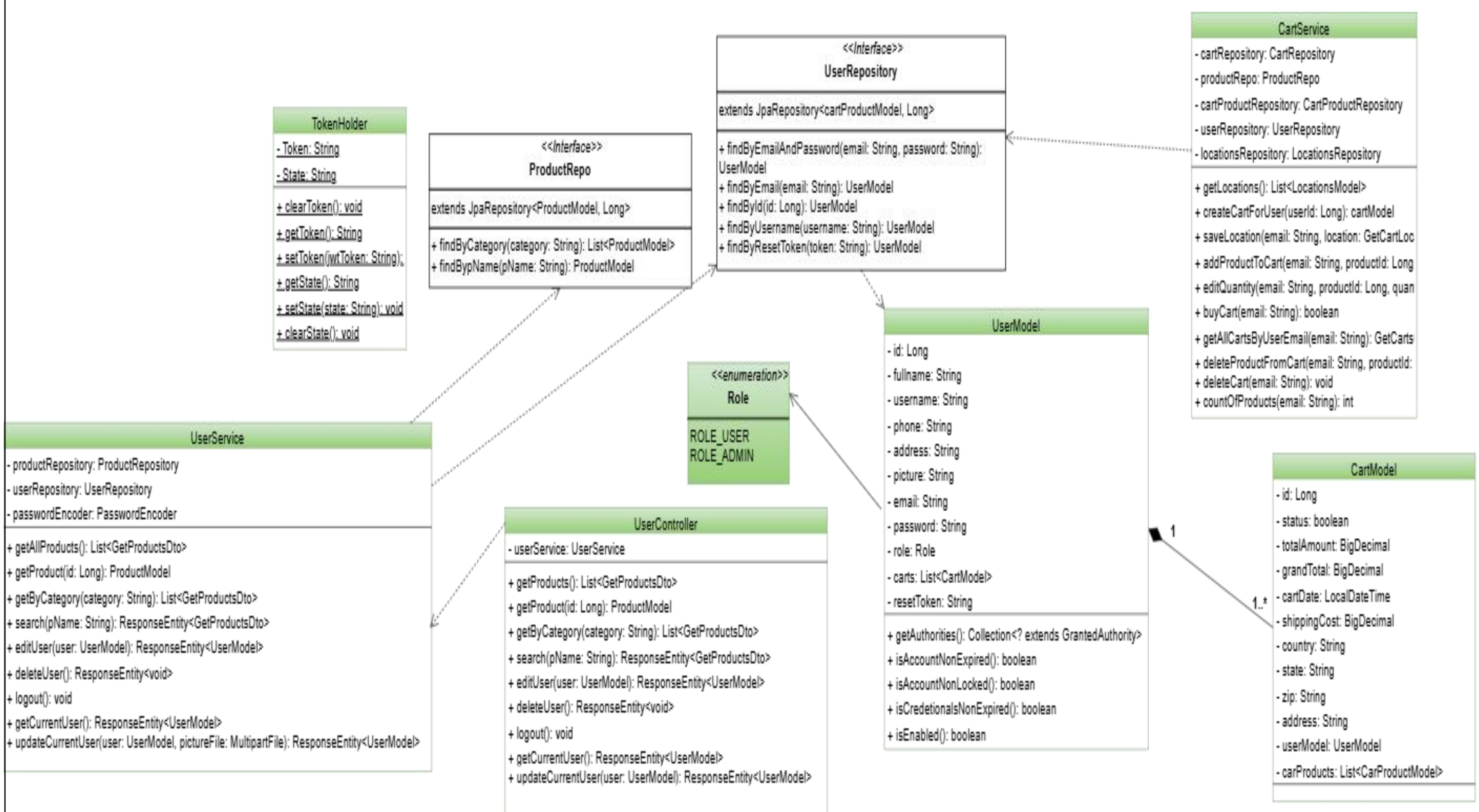


Admin
Admin interface
Add/ Delete / Update Product
Admin Controller
Admin Service
Admin Repository
Database

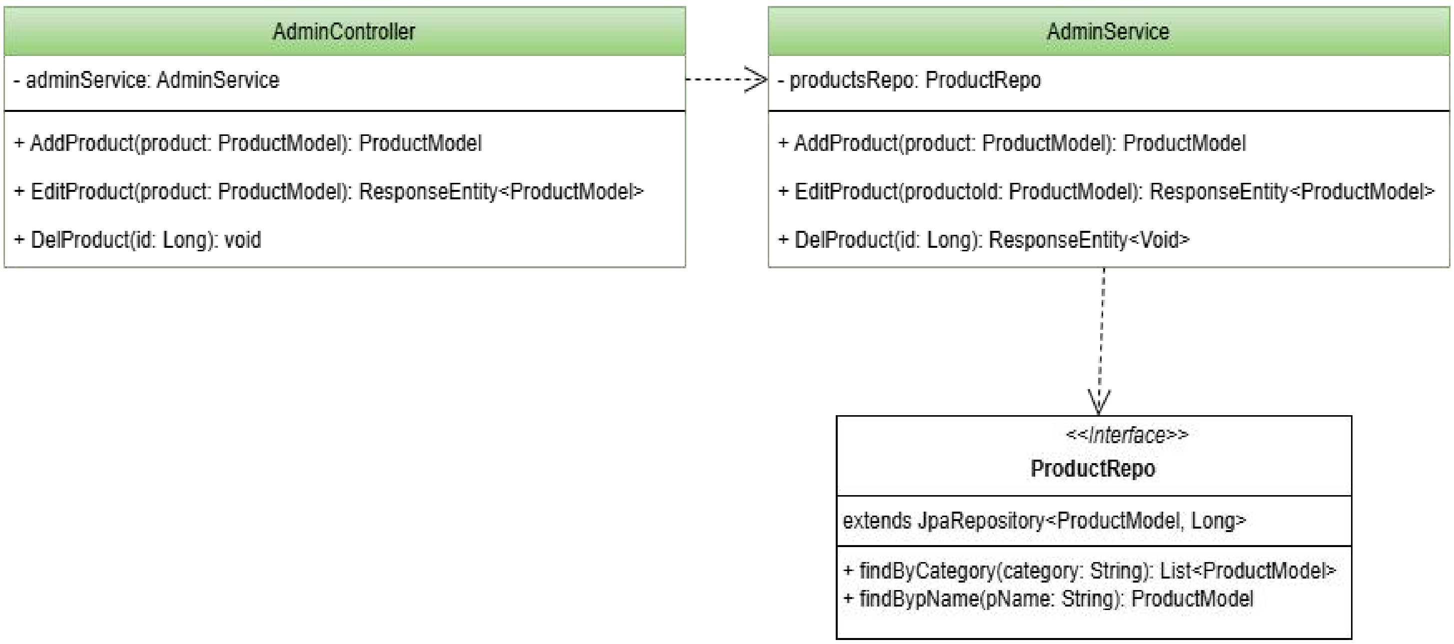
g. Class Diagram

Models the system's classes, their attributes, methods, and the relationships between them, used for object-oriented design.

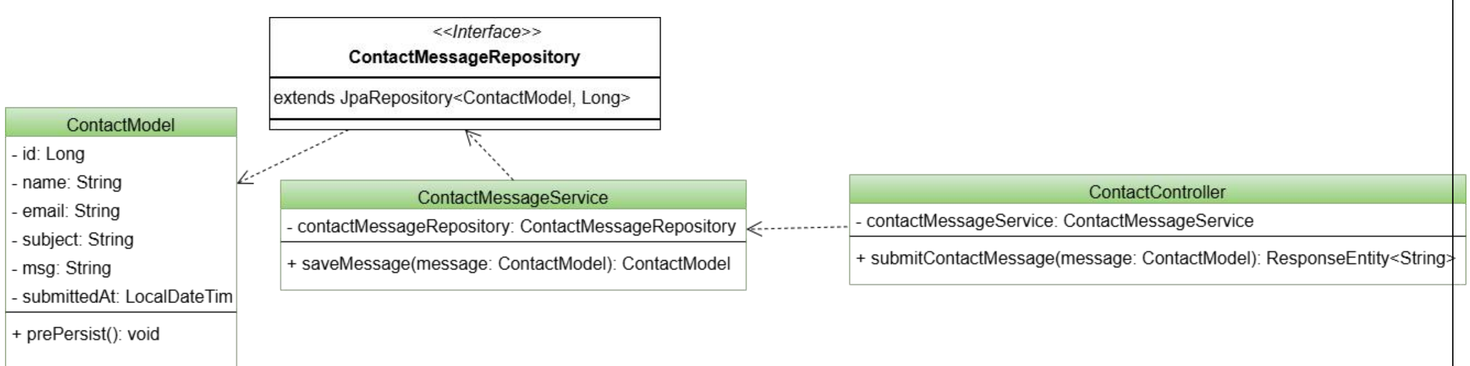
1. User



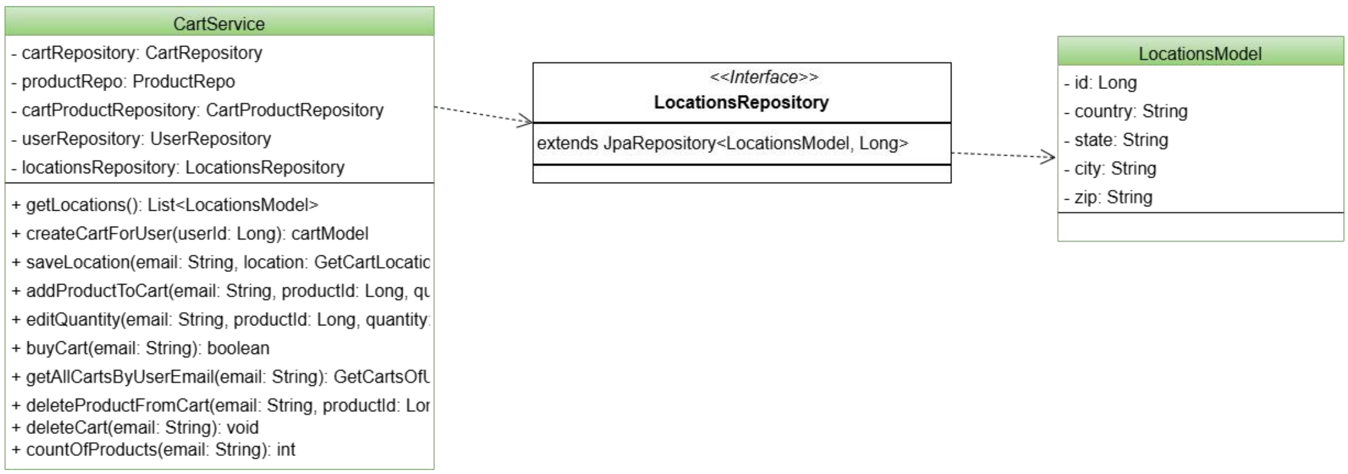
2. Admin



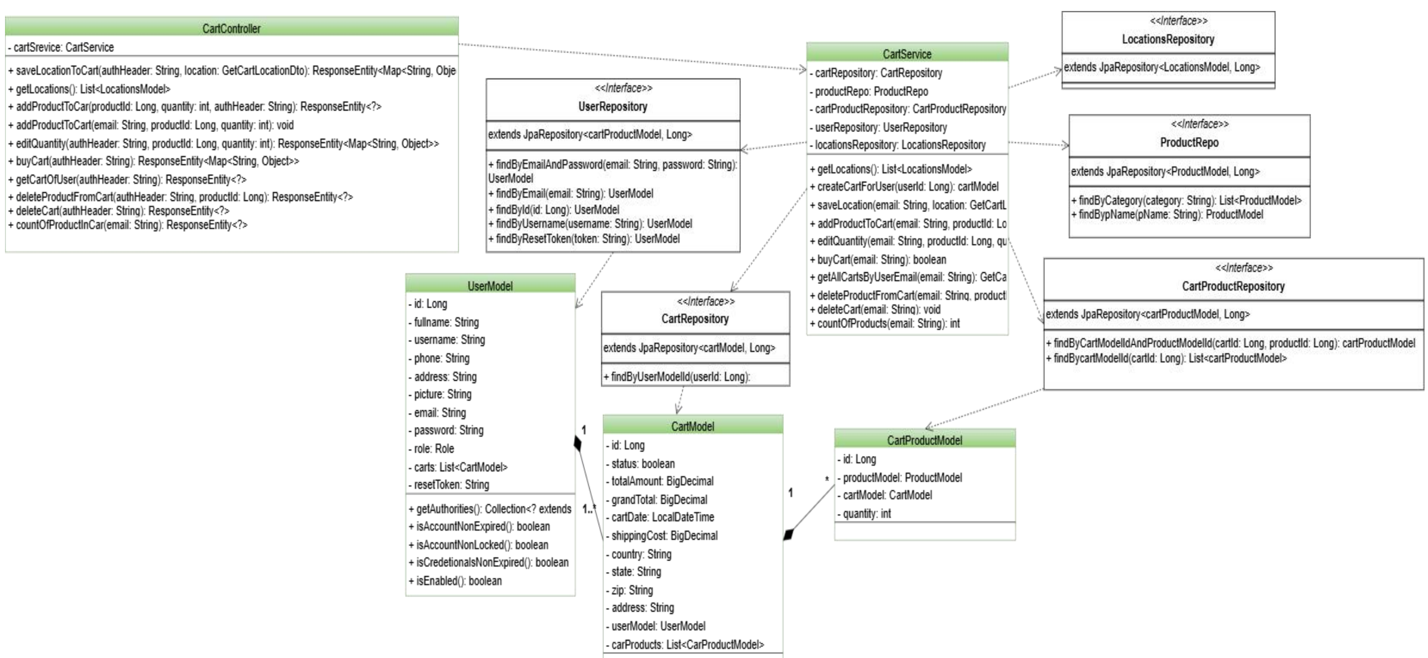
3. Contact



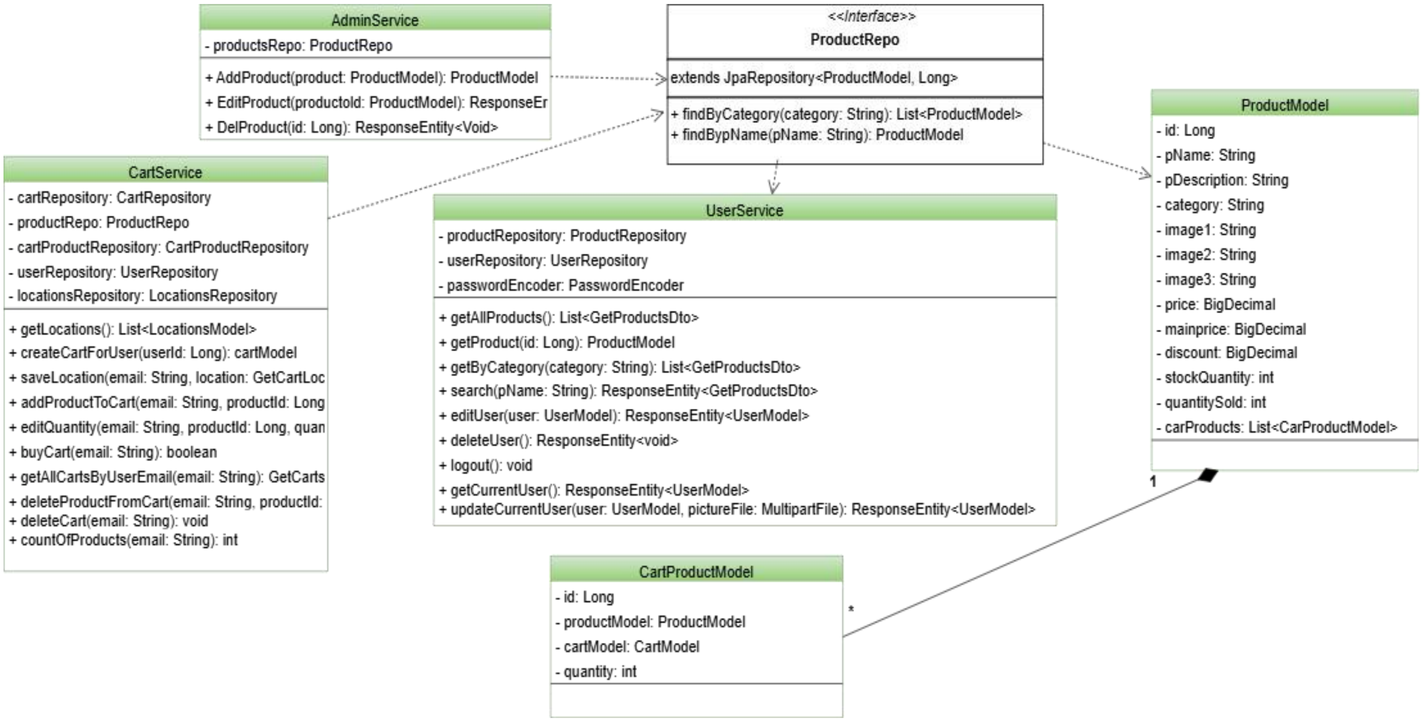
4. Location



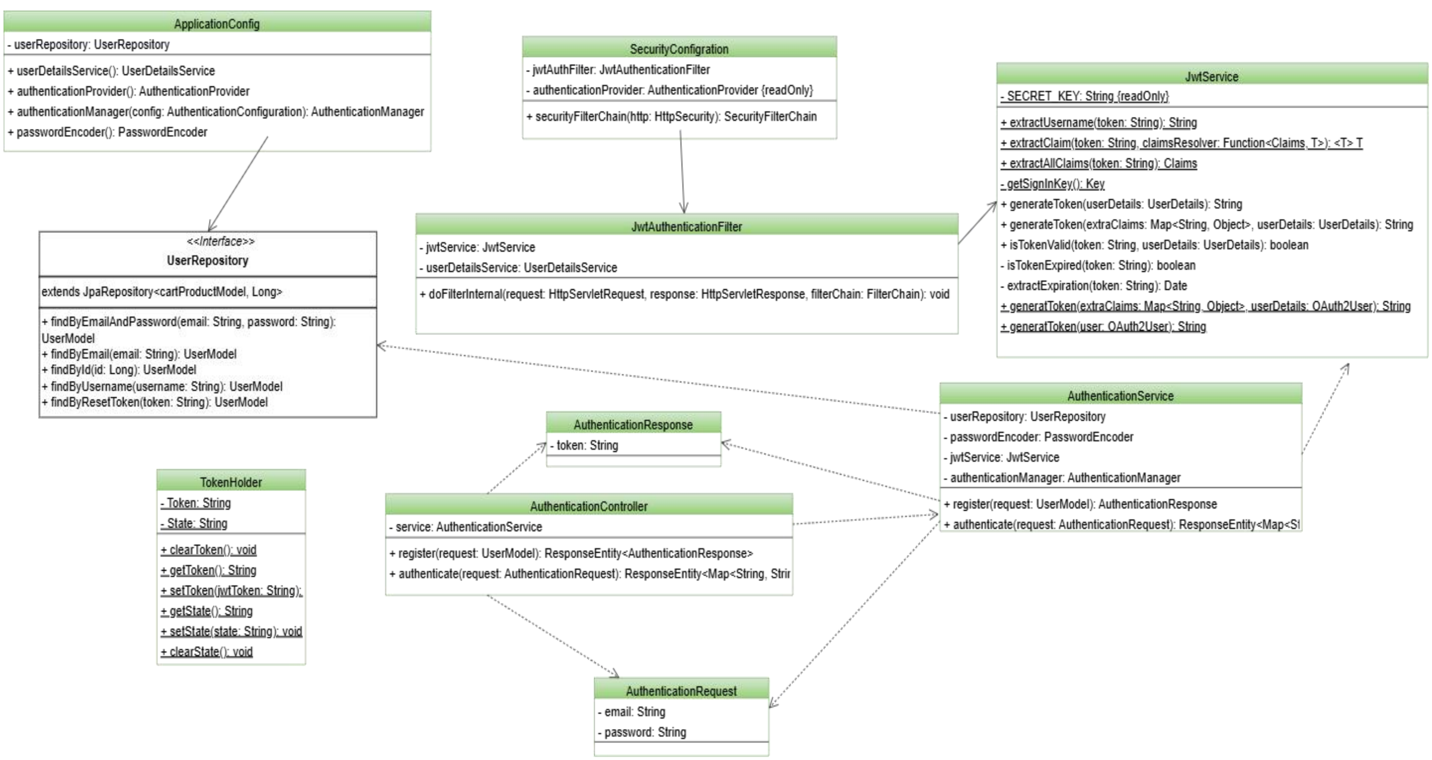
5. Cart



6. Product

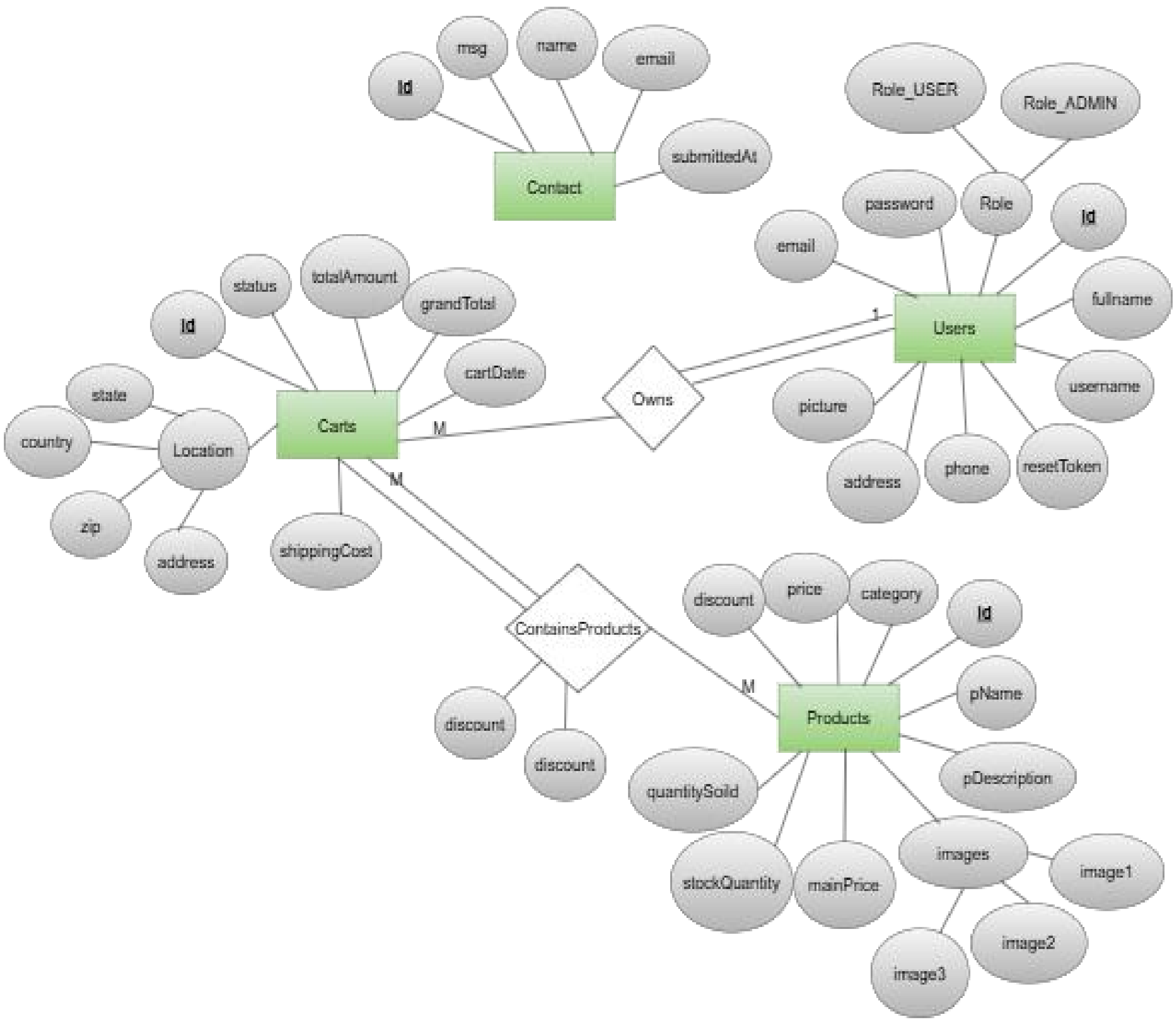


Security



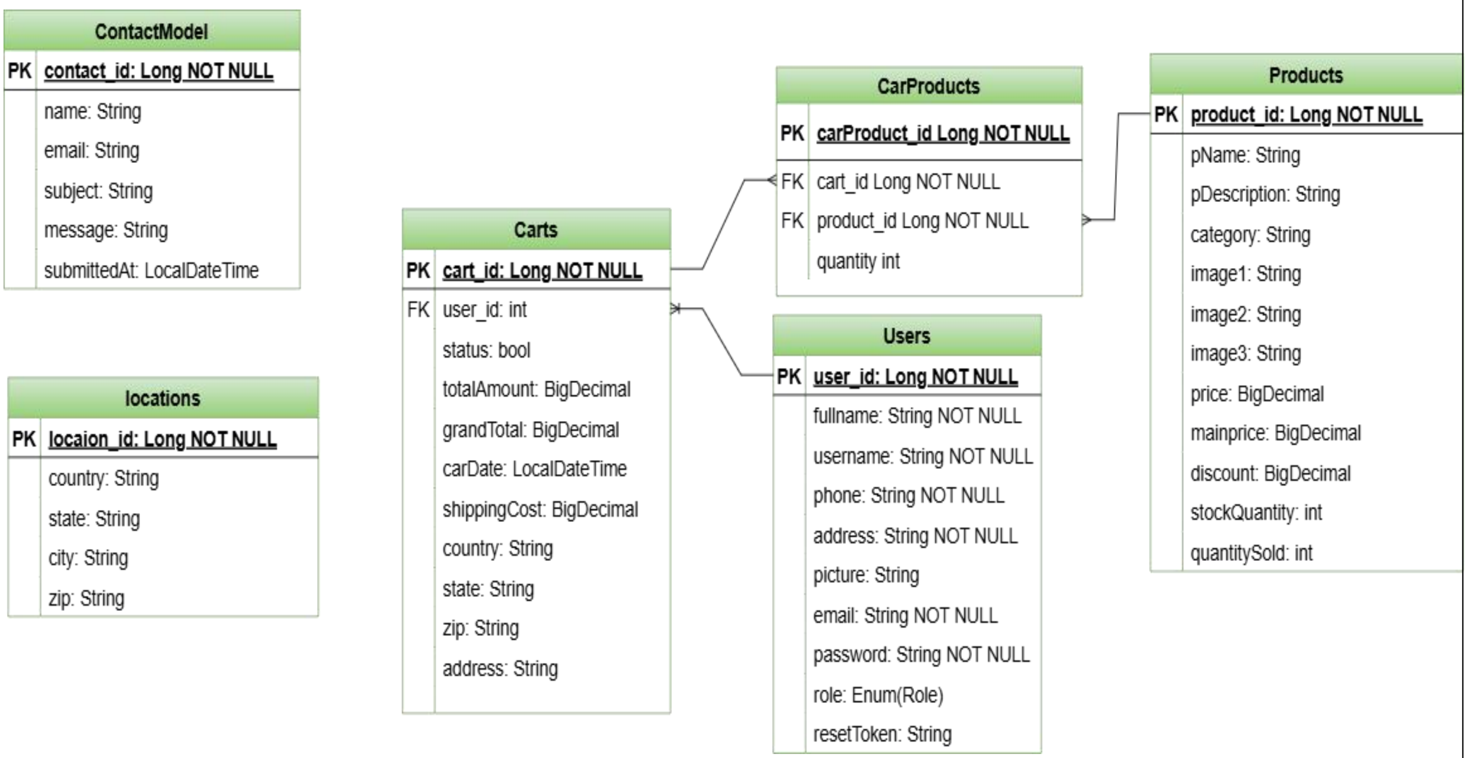
Entity Relationship Diagram (ERD)

Represents the logical structure of the database by showing entities, attributes, and relationships.



Entity Diagram

Focuses on the entities used in the system and their connections, often aligned with the ERD.



Thank You