HIGH LEVEL DESIGN DOCUMENT

Intelligent Fashion Generation and Retrieval System

UE17CS490A – Capstone Project Phase – 1

Submitted by:

Kavya Khatter	**PES1201701136**
Adithya M Vardhan	**PES1201700788**
Daksha Singhal	**PES1201700847**

Under the guidance of

Prof. KS Srinivas
Designation
PES University

August - December 2020

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING
PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013)
100ft Ring Road, Bengaluru – 560 085, Karnataka, India

PES UNIVERSITY

HIGH LEVEL DESIGN DOCUMENT

TABLE OF CONTENTS

1. Introduction 3
2. Current System 3
3. Design Considerations 4
   3.1 Design Goals 4
   3.2 Architecture Choices 4
   3.3 Constraints, Assumptions and Dependencies 6
4. High-Level System Design 7
   4.1 Logical user group 7
   4.2 Project Components 7
   4.3 Application Components 7
   4.4 System Functionalities 8
   4.5 Security 10
5. Design Description 10
   5.1 Master Class Diagram 11
   5.2 Reusability Considerations 11
6. ER Diagram / Swimlane Diagram / State Diagram 12
7. User Interface Diagrams 12
8. Report Layouts 13
9. External Interfaces 13
10. Help 13

Appendix A: Definitions, Acronyms and Abbreviations 14
Appendix B: References 14
Appendix C: Record of Change History 15
Appendix D: Traceability Matrix 15

1. Introduction

In the fashion industry, every designer has a unique taste, a unique sense of style and a designated aesthetic to their products. To deliver such a large variety of designs, fashion designers spend months together in putting up a montage suited to a particular theme in the time and place and are compensated accordingly. But what if we could cut short that process by automating the process? Study the previous designs of the designer and produce a large variety of apparel that can be selected to help push the process of elimination.

Using variations of generator models, we aim to develop a model that takes in several inputs varying in colour, shape, design and texture, then used to create randomized outputs of different textures and colours which bear a similar resemblance to the original works of the designer but also new combinations the designer could contemplate in order to gain a better insight.

To implement this, we propose to use an improvised version of GANs to provide an unsupervised separation of high-level attributes of the generated images. The generated images have been known to suffer from low-resolution, a problem to which we find a solution in the application of Super-Resolution GANs.

The application of such a product lies in many fields within fashion. Besides the proposed use case in apparel, implementations lie in designing of large draperies, accessories and jewellery.

2. Current System

At the moment, there seems to be no product out there with an objective similar to ours. Our innovative application of Generative Adversarial Networks to generate innovative new designs in apparel and clothing seems to be a unique solution as a product.

3. Design Considerations

3.1. Design Goals

The parameters we considered to be our guiding measurements towards an efficient product are:

- Accuracy: The Deep-Learning model should output valid designs specific to the needs of the client
- Speed: The model should be able to learn quickly and adapt to the changes of the different clients
- Privacy: Each client would be assured of the safety of the designs they've created
- Redundancy: models created would be stored for safe retrieval in the case of system failures
- Availability: Cloud-Storage would allow for quick access from different devices
- Ease of Use: Non-technical clients should be able to integrate the product seamlessly into their projects
- Scalability: With multiple clients running their ideas on the service, our product should be able to quickly scale up to meet the demand

3.2. Architecture Choices

In developing our product, we came across different approaches to achieve our objective:

1. Local Processing:

   To allow the clients to provide for their own Processing units based on their requirements. This allowed clients to choose the application architecture based on their constraints and requirements.

   Pros:

   - Faster retrieval of designs
   - smaller transfer time
   - requirement specific hardware

   Cons:

   - Fixed specs of hardware. Harder to change in case of a shift of requirements
   - Added expense to the client

2. Local Storage:

   To allow clients to store designs on local storage devices. This allows clients to set up their own data architecture and hardware specific to their requirements.

   Pros:

   - Faster dataset retrieval
   - Improved security
   - Client-specific hardware

   Cons:

   - Slower data retrieval for Processing unit
   - Incompatible data architectures
   - Data Redundancy
   - Added expense to the client
   - Scalability

3. Cloud Storage and Processing:

   To provide for Data storage and processing units based on the agreed requirements set by the client

   Pros:

   - Data redundancy
   - Easy scalability
   - Dynamic requisition of resources based on requirements
   - Compatible storage architecture

   Cons:

   - Slower Data Retrieval
   - Network constraints

Therefore, Cloud storage and Processing architecture is the most suitable for our product.

3.3. Constraints, Assumptions and Dependencies

The constraints of the system are:

- the size and features of the dataset entered
- The cloud platform used to run the model should have requisite computing power
- Scalability of the third-party cloud services being used
- Network constraints of client

The general assumptions of our product are:

- Clients possess sufficient designs to accurately train our model
- Server should be able to store and transfer large datasets of images
- client has stable internet connectivity

The general dependencies of the product are:

- CUDA enabled Processing Units
- Redundant cloud platform

4. High Level System Design

4.1. Logical user group

Based on the features of the product and the objective aimed at with the product, the major classes of users fall under :

1. Large companies: Companies that invest heavily into the designs coming out into their product line depend on their army of fashion designers to keep up with the demand and the dynamic market. Our model would help them focus on the features they like to repeat in their products while bringing out new features into the apparel.
2. Boutiques: Medium to small enterprises that pride themselves in possessing designs unique to their brand would be encouraged to use a tool that upholds that value while offering them a vast array of choices in different products.

4.2. Project Components

We have identified the following project components of our product:

1. Client Tier: The User Interface of our product, enabled through a website. This component allows the client to interact with our model and develop brand-specific designs without gaining access to other privileged information.
2. Data/Storage Tier: This component is tasked with storing information coming from the client as well as the Processing Unit. Tasks also include validating login information and storing time-specific task data.
3. Processing Tier: This component is the crux of our product. Containing the model ready to be trained for brand-specific apparel.

4.3. Application Components

The application components are arranged based on the project components they are a part of, as the following

4.3.1. Client Tier

The client tier contains applications that allow the client to gain seamless access to our product

1. Login
2. Submit designs
3. View designs
4. Text-Based design retrieval

4.3.2. Data Tier

The data tier contains applications that allow us to design at a redundant capacity.

1. Store input designs
2. Store generated designs

3. Validate logins
4. Store task data

4.3.3. Processing Tier

The processing tier consists of applications that interact with the model

1. Train model on brand-specific designs
2. Generate designs

4.4. System Functionalities

4.4.1. Login

Functionality to allow users to gain access to their designs and personalized data

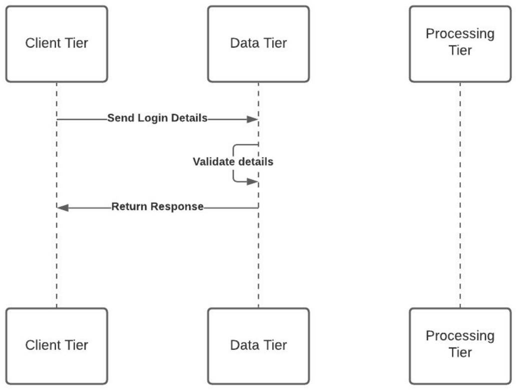

Fig. 4.1: Interaction Diagram for Login

4.4.2. Train Model

Functionality for users to input brand-specific designs to the model and train it to generate the most accurate designs

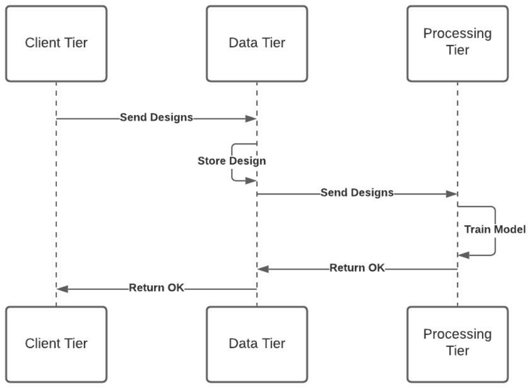

Fig 4.2: Interaction Diagram for training the model

4.4.3. Generate Designs

Functionality to use the trained model to generate new designs specific to the brand

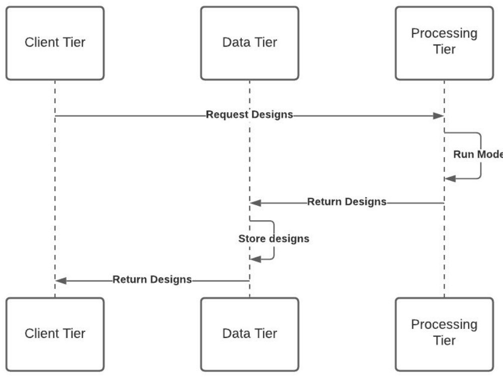

Fig 4.3: Interface Diagram of Generating Designs

4.4.4. Text-Based design retrieval

Functionality to retrieve designs based on high-level features

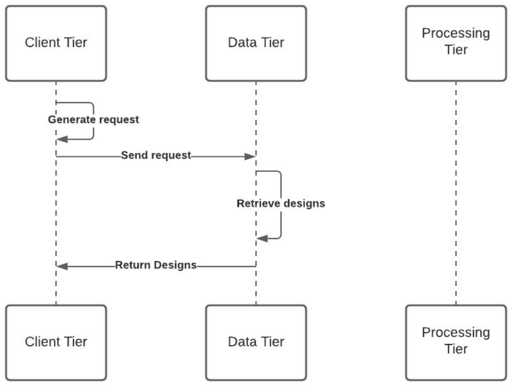

Fig 4.4: Interface Diagram for Text-Based Retrieval

4.5. Security

Security is an essential part of the system. Designers working with our product would be working within extremely private workspaces accessible only to them. Designs would be allowed to be shared/exported only by direct collaborators of the project.

5. Design Description

The main design features include 4 different parts: the Client UI, the database, the admin and the Processing component. Each component would be modularized efficiently for better productivity and reuse.

The Client UI component resides within the systems of our end users. It would include the interfaces for creating projects, tasks and displaying data, as well as providing an interface to view, share and export the designs from our model.

The Database component would be stored on a cloud platform. This would allow for better transferability and redundancy. The database component would allow for efficient storage and retrieval of images/data. The database component would store designs inputted from the model, the designs generated from the model and the user information.

The Processing Component would also be executed upon a cloud service. This would allow for better access to required computation at significantly lower costs. The Processing component would allow for easy access to the model along with easy retrieval of designs generated.

Finally, we have the admin component which oversees the other components and generates reports for the same.

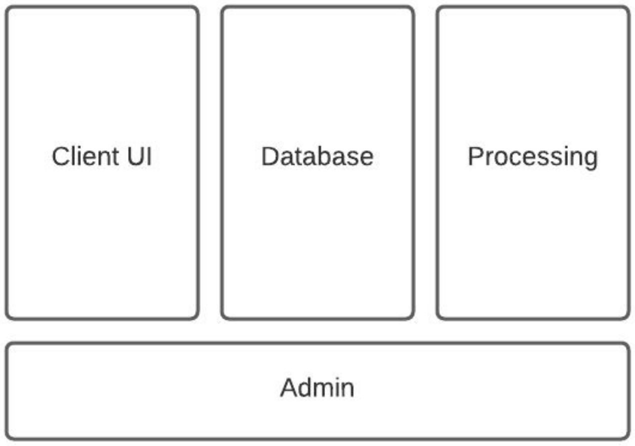

Fig. 5.1: Design Interface Diagram

5.1. Master Class Diagram

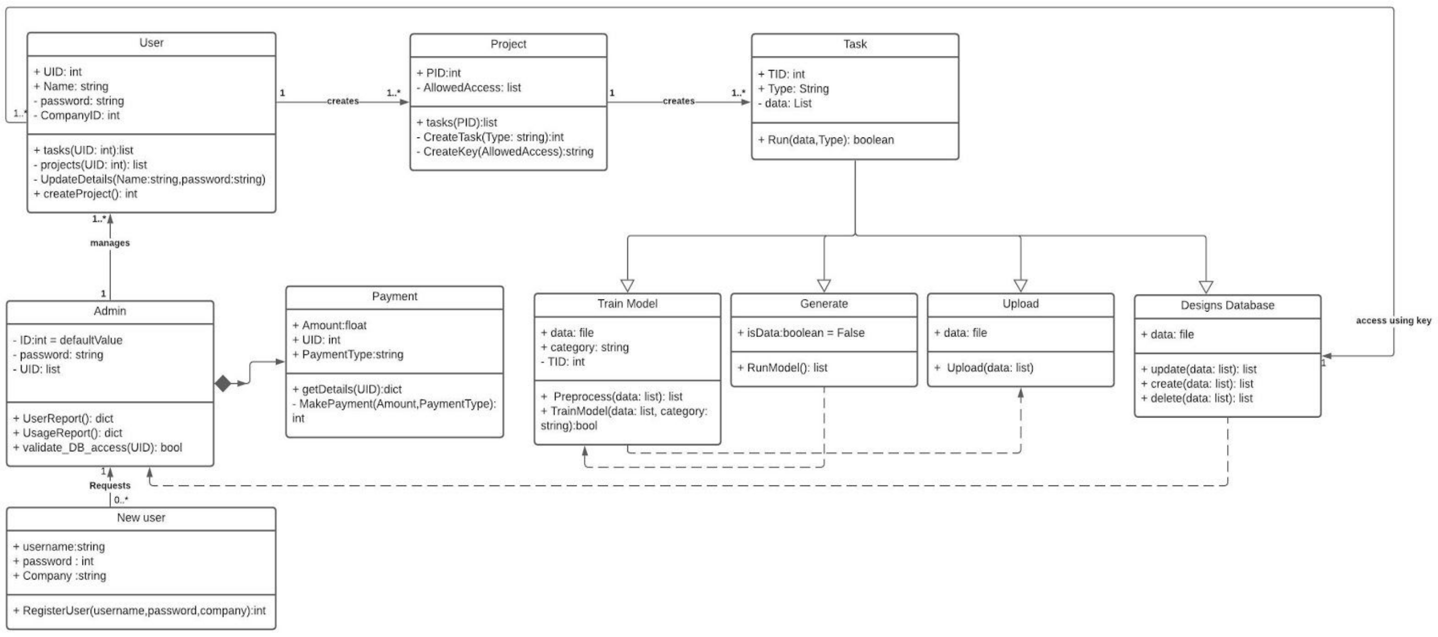

Fig. 5.2: Master Class Diagram

5.2. Reusability Considerations

For considerations regarding reusability of code and methodologies, we have:

- Singleton design pattern for database access and better security
- Decorator design pattern for different types of tasks assigned

6. Activity Diagram

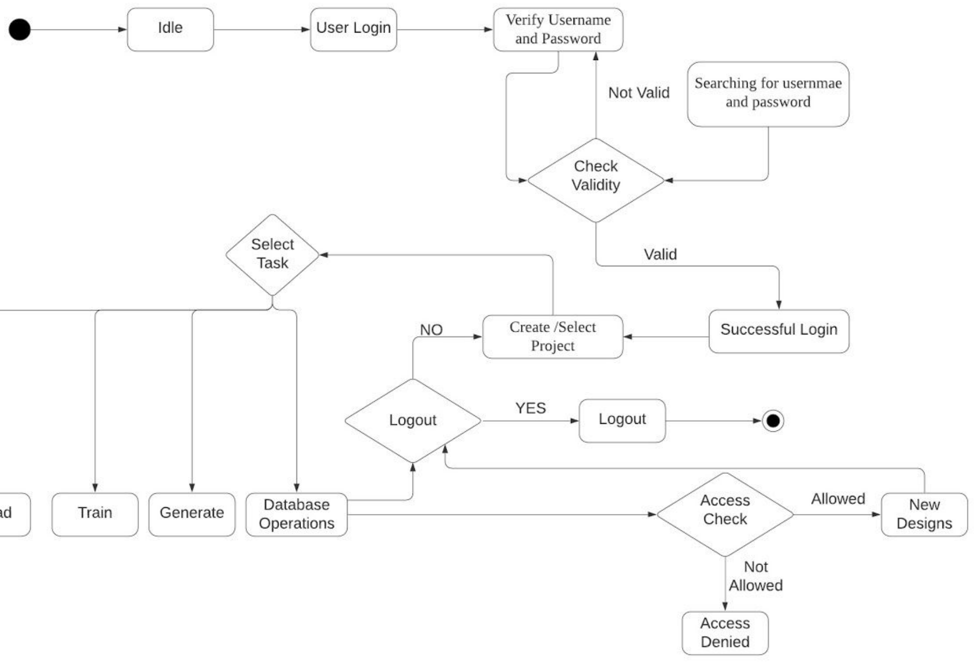

Fig 6.1: Activity diagram

7. User Interface Descriptions

Login.html - the first page of the UI, where the user logs into his profile

FrontPage.html - Display the different projects and tasks currently run by the user

Project.html - Display the features of a particular project and all the designs generated

Tasks.html - Display the list of currently running tasks as well as previous tasks run

Train.html - Page to upload images to train the model

Generate.html - Page to load images generated by the model

Admin.html - admin page to view user traffic and change settings for users

8. Report Layouts

Synchronous Reports would be generated in a periodic fashion, to describe user interactions as well as the tasks performed by the Processing unit.

User reports would include, but not be limited to, update on number of users, average product utilisation per user, median frequency of use and other usage patterns.

Task reports would include, but not be limited to, number of tasks executed in unit time(week/month/year), size of task executed, time taken for task, specifications of task and any inconveniences during task execution.

Asynchronous reports could include feedback related to product, overall bug report,

9. External Interfaces

9.1. User Interfaces

The model shall provide a very intuitive and simple interface to the fashion houses so that they can upload their fashion apparels like shirts and pants , and the model can generate new fashion images that stand on the same line of the particular fashion company.

The user is allowed to upload more than a certain number of images and particular resolution such that the images produced by the model have an expected degree of resolution. The time taken by the generator model to generate images would take not more than a week but may differ given the resolution and number of images.

The UI will pop an error message if the number of images provided by the user is less than a certain threshold and the resolution is lower than set by the admin.

9.2. Hardware Requirements

Server Side

The server is a high GPU based system to run the GAN model in order to generate images faster with higher resolution.

Client Side

The client side requires a browser such as Chrome which supports HTML & Javascript where they need to ready the design and upload it on to the server.

Linking Client And Server

Web frameworks such as flask will be used to link the client and server where the client can upload his choice of apparels and the server can then generate new fashion clothes and send it back to the client.

9.3. Software Requirements

Anaconda

- Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing, that aims to simplify package management and deployment.
- Version / Release Number:2020.02
- Operating Systems: Windows 10,Ubuntu, IOS
- Tensorflow,pytorch

Windows

- Windows 10 is a series of operating systems developed by Microsoft and released as part of its Windows NT family of operating systems.
- Version / Release Number:10

Database

- SQL is a domain-specific language used in programming and designed for managing data held in a relational database management system, or for stream processing in a relational data stream management system.
- Version / Release Number:15
- Operating Systems: Windows 10,Ubuntu, IOS

10. Help

Help will come in the form of all the documentation created post coding, which will explain the intended uses. Should time allow, detailed instructions will be written on how to create and implement the system with the intentions of publishing as an Open Source solution.

Appendix A: Definitions, Acronyms and Abbreviations

GPU : A graphics processing unit is a specialized, electronic circuit designed to rapidly manipulate and alter memory to accelerate the creation of images in a frame buffer intended for output to a display device.

CUDA : CUDA is a parallel computing platform and application programming interface model created by Nvidia.

NT : Windows NT is a family of operating systems produced by Microsoft, the first version of which was released on July 27, 1993. It is a processor-independent, multiprocessing and multi-user operating system.

GAN : A generative adversarial network is a class of machine learning frameworks designed by Ian Goodfellow and his colleagues in 2014.

HTML : Hypertext Markup Language is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets and scripting languages such as JavaScript.

PHP : PHP is a general-purpose scripting language especially suited to web development.

Appendix B: References

[1] Natsumi Kato, Naoya Muramatsu , Hiroyuki Osone, Yoichi Ochiai , Daitetsu Sato University of Tsukuba,*DeepWear: a Case Study of Collaborative Design between Human and Artificial Intelligence*

[2] Tero Karras ,Samuli Laine, Timo Aila NVIDIA,*A Style-Based Generator Architecture for Generative Adversarial Networks*

[3] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, Wenzhe Shi Twitter,*Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network*

[4] Roman Zeyde, Michael Elad, and Matan Protter,*On Single Image Scale-Up Using Sparse-Representations*

[5] Alec Radford, Luke Metz, and Soumith Chintala,Unsupervised representation learning with deep convolutional generative adversarial networks In International Conference on Learning Representations (ICLR), 2016.

[6]Seyedarian Hosseini, Thomas Boquet, Wojciech Stokowiec, Ying Zhang, Christian Jauvin, Chris Pal,Fashion-Gen: The Generative Fashion Dataset and Challenge,Negar Rostamzadeh

[7] Y. Pu, Z. Gan, R. Henao, X. Yuan, C. Li, A. Stevens,and L. Carin, "Variational autoencoder for deep learning of images, labels and captions," in Advances in neural information processing systems, 2016, pp. 2352–2360.