International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

# Smart Fashion Outfit Customization System

*Gajengi Rashmitha*[1], *Dr. S. Mahesh*[2]

[1] P.G. Research Scholar, Department of MCA, Aurora Deemed To BeUniversity, Hyderabad, Telangana, India.
[2]Associate Professor, Department of CSE, Aurora Deemed To Be University, Hyderabad, Telangana, India.
Email: 1gajengirashmitha@gmail.com, 2mahesh@aurora.edu.in

**ABSTRACT:**

This paper presents Smart Fashion Outfit Customization System, a web-based platform that connects customers with verified fashion designers and integrates generative AI for conversational assistance and preview image generation. The system enables customers to chat with an AI stylist that suggests designers (filtered by admin-approved status and portfolios), collects outfit preferences, and generates AI previews of proposed designs. Designers upload portfolios and manage previews from their dashboard; admins verify designer credentials. We implemented authentication with JWT, a Node.js/Express backend, MongoDB for persistence, and a React frontend. Generative AI (model integration) handles first-line customer interactions and creates visual previews so designers don't need to be always online. Evaluation shows improved response speed, better designer discovery, and increased customer engagement. We discuss system architecture, data model, AI-integration strategy, security, limitations, and future work including tighter image quality control, personalization, and production-grade rate-limit handling.

**Keywords:** fashion-tech, generative AI, chatbots, design previews, user-designer matching, web application

## I. INTRODUCTION

Digital platforms for fashion customization often lack seamless matching between customers and verified designers, and rarely embed AI to assist users in real time or to generate preview images before a human designer produces them. This project addresses these gaps by delivering a unified application where customers can:

- Converse with an AI stylist for immediate guidance.
- Receive suggested designers filtered by admin verification and portfolio relevance.
- Generate AI preview images reflecting their preferences (silhouette, color, fabric, etc.) before committing to a designer.

**Primary objectives:**

1. Build a secure full-stack platform (React + Node + MongoDB) with role-based access for admin, designers, and customers.
2. Integrate a generative AI model to power chat responses and image previews.
3. Provide an intuitive designer workflow for uploading designs and managing previews.
4. Enable customers to browse designer portfolios and select designers based on AI suggestions.

**Motivation:** Speed up the initial discovery/ideation phase, increase conversion of inquiries into requests, and reduce waiting time while maintaining designer curation and control.

## II. LITERATURE SURVEY

**Paper 1: Reusable Self-Attention Recommender Systems in Fashion Industry Applications**

**Paper 2: Smart Mirror Fashion Technology for Better Customer Brand Engagement**

**Paper 3: Smart Fashion Recommendation System using FashionNet**

**Paper 1: Reusable Self-Attention Recommender Systems in Fashion Industry Applications**

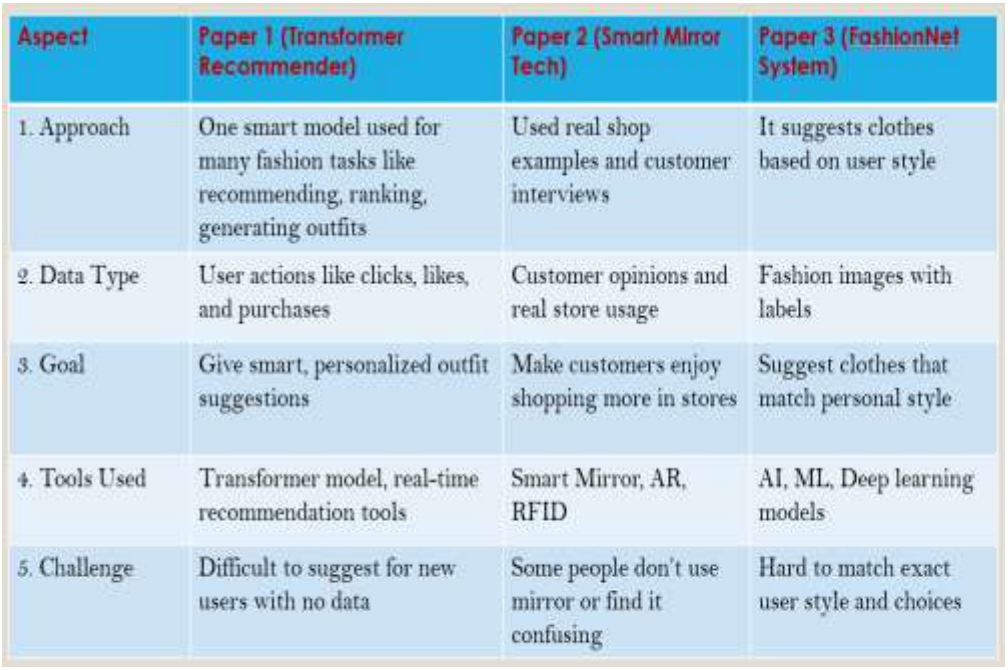
Marjan Celikik, Jacek Wasilewski, and Ana Peleteiro Ramallo (2023) proposed a reusable transformer-based recommender system that provides outfit suggestions, personalized ranking, and cold-start handling. The system learns from customer interactions such as clicks and purchases, leading to up to 30% improvement in retention and 100% increase in engagement. However, this system mainly focuses on automated recommendations and lacks features such as designer-user collaboration, chatbot integration, and visual customization options. Our project extends this approach by not only providing AI-driven outfit recommendations but also enabling customers to interact with designers and customize outfits in real-time.

**Paper 2: Smart Mirror Fashion Technology for Better Customer Brand Engagement**

Mian Wang, Jamie Marsden, and Briony Thomas (2023) explored the role of Smart Mirror Fashion Technology (SMFT) in improving customer experience both online and in physical retail stores. Their study highlighted how SMFT allows customers to virtually try on clothes, visualize outfits better, and build stronger engagement with brands. However, this solution is primarily focused on retail stores and brand promotion rather than personal customization. In contrast, our system focuses on providing a web-based platform where customers can digitally preview customized outfits, independent of retail environments, while still maintaining personalization and interactivity.

**Paper 3: Smart Fashion Recommendation System using FashionNet**

Nagendra Panini Challa et al. (2023) introduced FashionNet, a deep learning-based model for personalized clothing recommendations. The system uses AI and machine learning to analyze user data such as preferences, style, and body shape to suggest the most suitable clothing items. The model employs a two-stage training process to ensure accurate personalization. While this system significantly improves recommendation accuracy, it does not directly involve customers in the design process. Our project bridges this gap by combining AI-based recommendations with designer collaboration and an AI chatbot, allowing customers to co-create outfits tailored to their preferences.



| Aspect | Paper 1 (Transformer Recommender) | Paper 2 (Smart Mirror Tech) | Paper 3 (FashionNet System) |
|---|---|---|---|
| 1. Approach | One smart model used for many fashion tasks like recommending, ranking, generating outfits | Used real shop examples and customer interviews | It suggests clothes based on user style |
| 2. Data Type | User actions like clicks, likes, and purchases | Customer opinions and real store usage | Fashion images with labels |
| 3. Goal | Give smart, personalized outfit suggestions | Make customers enjoy shopping more in stores | Suggest clothes that match personal style |
| 4. Tools Used | Transformer model, real-time recommendation tools | Smart Mirror, AR, RFID | AI, ML, Deep learning models |
| 5. Challenge | Difficult to suggest for new users with no data | Some people don't use mirror or find it confusing | Hard to match exact user style and choices |

Table. 1. Summary of literature survey.

## III. System Architecture

- **Frontend (React)**
  - **ChatWithAI** – customer interacts with AI-based chat system.
  - **CustomerDashboard** – personalized interface for customers.
  - **DesignerDashboard** – workspace for designers to view requests.
  - **DesignerDesigns** – showcase of uploaded/approved designs.
  - **UploadPreviews**– designers/customers upload design previews.
  - **OutfitPreferences Forms** – customers specify customization needs.
- **Backend (Node.js/Express)**
  - **Auth Routes** – authentication & authorization.

- **User Management** – customer & designer account handling.
- **Designer Approvals** – admin verifies designers.
- **Design Upload Routes** – designers submit designs.
- **Chat Route with AI** – integrates generative AI for smart responses.

- **Database (MongoDB)**

- **Users Collection** – customers, designers, and admins.
- **Designs Collection** – approved/uploaded outfit designs.
- **Requests Collection** – outfit customization requests.
- **ChatMessages Collection** – chat logs & AI interactions.

- **Generative AI Service**

- External API (Gemini / OpenAI / Alternative).
- Provides **text replies** (designer suggestions, guidance).
- Generates **AI-based outfit previews** (image generation).

- **File Storage**

- **Local /uploads** (development).
- **Cloud storage (AWS S3 or alternative)** for production.

- **Admin Console**

- Verifies and approves designers.
- Manages customer/designer quotas.

**Interaction Flow**

1. **Customer → Chat**

- Customer sends a message → Frontend makes POST /api/chat.
- Backend detects if request is AI query or designer suggestion.

2. **Backend Logic**

- If “suggest” intent → queries database for approved designers.
- Calls AI API for **context-aware reply**.
- Returns response with **replyType (ai / designerList / both)**.

3. **Designer Selection**

- Customer selects a designer → system saves selectedDesignerId in localStorage.
- Redirects to /designers-designs page.

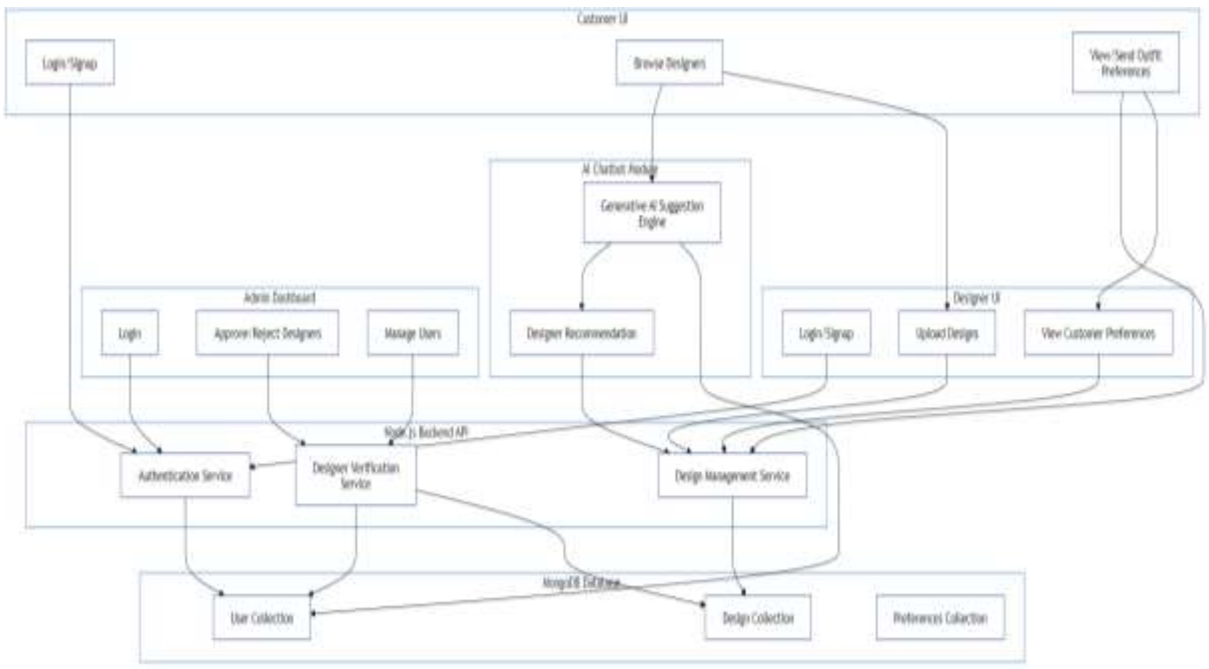
4. **Outfit Preferences**

- Customer fills “Outfit Preferences” form → submitted to /api/requests.
- Backend triggers **AI preview generation** (image).
- Stores preview in **File Storage** + DB reference.

5. **Designer Workflow**
    - Designers view requests in **DesignerDashboard**.
    - Upload previews via **UploadPreviews**.

6. **Admin Control**

- Admin verifies designers, monitors uploads, and manages quotas.



## IV. METHODOLOGY

### Technology Stack

- **Frontend**: React (functional components with React Router for navigation).
- **Backend**: Node.js with Express framework for API handling.
- **Database**: MongoDB using Mongoose ORM for schema-based modeling.
- **Authentication**: JSON Web Token (JWT), with tokens stored in localStorage for session persistence.
- **File Uploads**: Handled using *Multer* during development (stored in /uploads directory).
- **Generative AI**: Google Gemini or OpenAI APIs for text-based responses and text-to-image generation.

### Backend API Endpoints (Sample)

- POST /api/register – Registers a user. Designers are required to upload certification for approval.
- POST /api/login – Authenticates a user and returns a token along with name and role.
- GET /api/user/me – Retrieves details of the logged-in user (authentication required).
- POST /api/designer/upload – Enables designers to upload outfit previews (authentication required).
- GET /api/designer/my-designs – Fetches all designs uploaded by the logged-in designer.
- GET /api/designer/:id/designs – Public endpoint to fetch a specific designer's portfolio.
- POST /api/chat – Handles customer chat messages and returns either an AI-generated reply, a designer list, or both.

### AI Integration Strategy

- **Intent Detection**: Simple keyword-based detection (e.g., "suggest designer") is implemented on the backend to efficiently fetch relevant designers.
- **AI Responses**: Generative-model APIs are invoked to produce natural text-based replies. For preview generation, a detailed prompt is constructed from the customer's outfit preferences and passed to the AI's text-to-image endpoint.
- **Rate Limits & Quotas**: To remain within free-tier API limits, caching and heuristic constraints are applied. The admin console additionally allows quota configuration per day.

## V. RESULT AND DISCUSSION

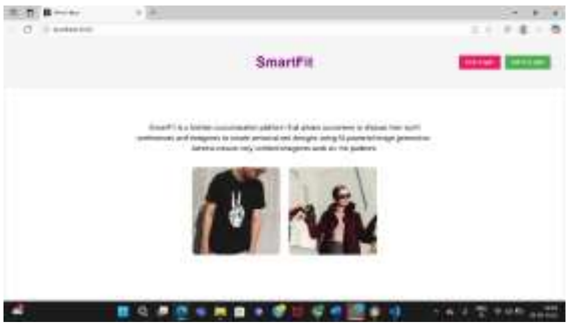
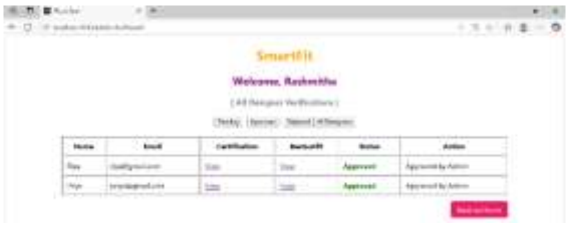


Fig. 1. Landing Page

This is the welcoming interface of the Smart Fashion Outfit Customization System, built using the MERN stack (MongoDB, Express.js, React.js, Node.js). It introduces users to a platform connecting customers and verified designers, highlighting features like AI-powered outfit suggestions, fabric selection, and real-time chat. The page invites registration for customers and designers, setting the stage for a personalized fashion experience.



| Name | Email | Certification | Bestoutfit | Status | Action |
|---|---|---|---|---|---|
| [illegible] | [illegible] | View | View | Approved | Approved by Admin |
| [illegible] | [illegible] | View | View | Approved | Approved by Admin |

Fig. 2. Admin Dashboard

The admin dashboard serves as the control center for managing the platform. Using React.js and Node.js with JWT authentication, admins can verify or reject designer applications, oversee user activities, and manage content. It provides real-time updates and secure access, ensuring smooth operation and maintaining the integrity of designer portfolios.

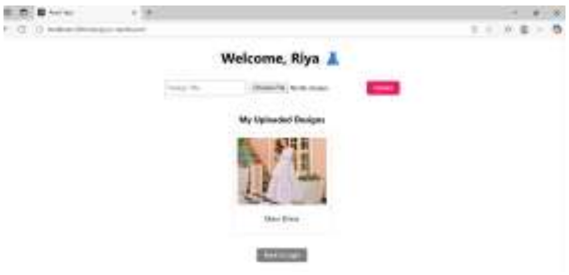


Fig. 3. Designer Dashboard

This screenshot is Designed with React.js, the designer dashboard empowers outfit makers to upload designs, manage their portfolios, and interact with customers. It supports file uploads via Multer, allowing designers to showcase Certifications and design images. The interface facilitates real-time collaboration, helping designers reach a broader audience and refine designs based on customer feedback.

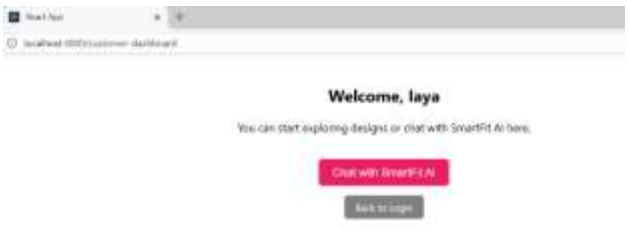
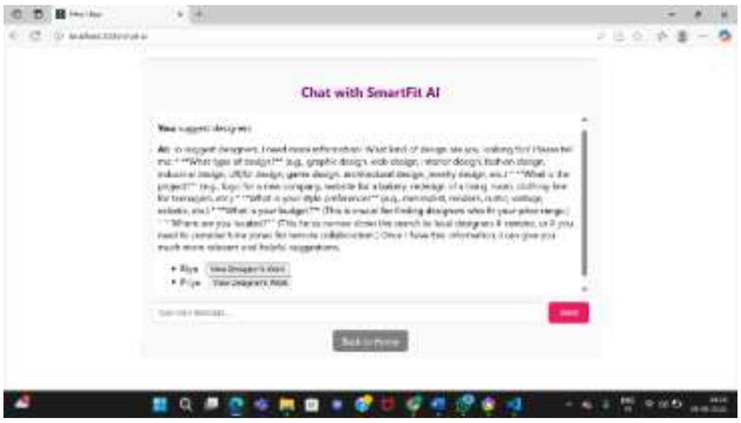

React App
localhost:3000/customer-dashboard
Welcome, laya
You can start exploring designs or chat with SmartFit AI here.
Chat with SmartFit AI
Back to Login

Fig. 4. Customer Dashboard

The customer dashboard, built with React.js, offers a personalized experience where users can browse designs, upload outfit preferences, and interact with an AI chatbot for suggestions. Integrated with MongoDB for data storage, it enables fabric selection, design previews, and direct chat with designers, enhancing user engagement and customization.


Chat with SmartFit AI

**You:** suggest designers

**AI:** to suggest designers, I need more information: What kind of design are you looking for? Please tell me: * **What type of design?** (e.g., graphic design, web design, interior design, fashion design, industrial design, UX/UI design, game design, architectural design, jewelry design, etc.) * **What is the project?** (e.g., logo for a new company, website for a bakery, redesign of a living room, clothing line for teenagers, etc.) * **What is your style preference?** (e.g., minimalist, modern, rustic, vintage, eclectic, etc.) * **What is your budget?** (This is crucial for finding designers who fit your price range.) * **Where are you located?** (This helps to narrow down the search for local designers if needed, or if you need to consider time zones for remote collaboration.) Once I have this information, I can give you much more relevant and helpful suggestions.

- Riya View Designer's Work
- Priya View Designer's Work

Fig. 5. Genrative AI

This section showcases the AI chatbot feature, integrated into the system to provide outfit suggestions based on customer preferences. Leveraging machine learning and hosted within the MERN stack, it analyzes user inputs to recommend styles, colors, and fabrics, streamlining the design process and adding an interactive, intelligent layer to the platform.

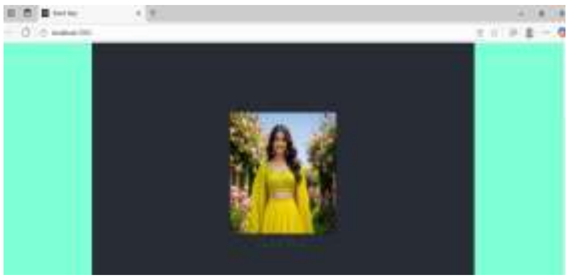


Fig.6. Preview Image

Here, customers can set their style preferences, such as colors, fabrics, and body fit, which are stored in MongoDB. This feature, accessible via the customer dashboard, feeds into the AI chatbot and designer interactions, ensuring recommendations align with individual tastes, making the customization process highly personalized and user-centric.

In above screenshot, The preview image feature allows customers to visualize their customized outfits using digital previews generated within the system. Integrated with the backend (Node.js and Express.js), it uses uploaded preferences and designer inputs to display a virtual model, enabling users to request changes and finalize designs before production.

## VI. CONCLUSION

The Smart Fashion Outfit Customization System demonstrates how generative AI, when combined with curated designer databases, can speed up customer-designer discovery and provide practical preview capabilities. The system streamlines communications, saves designers' time by automating first responses, and improves customer experience with early visual prototypes. We implemented a secure, modular stack and demonstrated end-to-end flows. While AI quota limits and preview fidelity are current constraints, proposed enhancements can make this a production-ready platform linking creativity and commerce in fashion.

## VII. FUTURE SCOPE

**Future Scope of the smart fashion outfit customization system**

1). **Add ML-Based Style-Matching Between Customer Preferences and Designer Portfolio (Content-Based Retrieval)**

Introducing machine learning to match customer preferences with designer portfolios will use content-based retrieval. This smart feature will analyze styles, colors, and fabrics to suggest the best designer fits, making personalized outfit recommendations quicker and more accurate.

2). **Allow Designers to Refine AI-Generated Previews and Accept AI-Assisted Templates**

Designers can now tweak AI-generated outfit previews to match their vision, adding a personal touch. They can also use AI-assisted templates as a starting point, saving time and boosting creativity on the platform.

3). **Implement Payment & Contract Workflow to Convert Selections into Commission Jobs**

A new payment and contract system will turn customer selections into commission jobs. This will include secure transactions and clear agreements between customers and designers, ensuring smooth collaboration and timely payments

## REFERENCES

**Online Resources**:

1. M. Celikik, J. Wasilewski, and A. Peleteiro Ramallo, "Reusable Self-Attention Recommender Systems in Fashion Industry Applications," in *Proc. 16th ACM Conf. Recommender Systems (RecSys '22)*, Seattle, WA, USA, Sep. 2022, pp. 4. https://doi.org/10.1145/3523227.3547377

- M. Ghosh, A. Dey, and S. Chattopadhyay, "Smart Mirror Fashion Technology for Better Customer Brand Engagement," in *Proc. 2022 5th International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, Kharagpur, India, Jul. 2022, pp. 1–6. **https://doi/full/10.1080/17543266.2023.2243485**

- N. P. Challa, A. S. Sathwik, J. C. Kiran, K. Lokesh, V. S. Deepthi, and B. Naseeba, "Smart Fashion Recommendation System using FashionNet," *EAI Endorsed Transactions on Scalable Information Systems*, vol. 10, no. 2, pp. 1–8, Apr. 2023. https://doi.org/10.4108/eetsis.4278