Software Requirements Specification

FashionFinder

Team Members: Maria Alsamaien, Alex Gjeka,

Angjelo Mana & Oloofa Kalid

Table of Contents

1. Introduction

The complete Software Requirements Specification (SRS) document for FashionFinder offers a comprehensive blueprint for the development and implementation of the website. It encompasses detailed descriptions of the project's objectives, functionalities, user characteristics, interfaces, software requirements, and constraints. Providing stakeholders with a thorough understanding of FashionFinder's scope and operation. Through its structured organization and comprehensive content, the SRS serves as an essential reference point for guiding stakeholders throughout the entirety of the FashionFinder project.

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to give a thorough explanation of the FashionFinder website's functionality. This document will go over every planned features of the system and provide an early look at the user interface of the software application. Furthermore, this document will encompass all technological aspects essential for the FashionFinder website's functionality.

This document is intended for all individuals participating in the FashionFinder project. Each individual member will rely on the SRS to understand the project's objectives, functionalities, constraints, and technical aspects. The SRS serves as a crucial reference point to ensure the project's success.

1.2 Scope

This project entails the development and delivery of a user-centered website designed to facilitate a personalized shopping experience through innovative features. The core functionality of the website revolves around a unique "personal designer" feature, which allows users to find outfits tailored to their preferences. Key features include the ability to browse outfits sourced from brands permitting web scraping, refine searches through communication with the personal designer, and save outfits to customizable albums. Notably, the website will not incorporate any direct purchasing capabilities, as it does not function as a vendor for the clothing items showcased.

Included Features

- Personal designer functionality to refine outfit searches.

- Creation and deletion of user albums.
- Addition and removal of items from user albums.
- Customer review system.
- Display of outfit recommendations in image format with corresponding links.
- Customer support services.
- Outfit sharing functionality.
- Account creation and management (including screen name, password, and email changes).
- Authentication mechanisms.
- Brand preferences and budget features to enhance user experience.

Excluded Features

- Direct purchasing capabilities.
- Vendor functionalities for clothing sales like:
    - Order Tracking
    - Returns and Exchanges
    - Payment and Billing
    - Shipping and Logistics
    - Promotions

Expected Objectives and Goals

The primary objective of this project is to provide users with a seamless and intuitive platform for discovering and creating personalized outfits. This website aims to enhance user engagement and satisfaction by offering tailored recommendations, efficient search functionalities, and convenient album management features.

FashionFinder is for all users of varying technical backgrounds. As it will prioritize user data protection by implementing encryption techniques for data and authentication methods like user email verification. Which will reduce the risk of unauthorized access and potential data breaches.

FashionFinder will also maintain a comprehensive process for monitoring compliance with web scraping policies to ensure adherence to legal guidelines and industry standards. This process involves regular reviews of web scraping activities as well as verifying the terms of service and usage policies of the websites being scraped. In case of any legal challenges or changes in those policies, we will adapt by revising data collection methods, or implementing additional security measures to ensure compliance with evolving legal requirements.

FashionFinder will also ensure its APIs can handle a lot of users by testing them thoroughly and using methods that let the system grow smoothly when more people use it. It will also set a limit on how many requests users can make and store commonly used data to make things faster, keeping the website running smoothly for everyone. We also use caching, which means storing commonly used data to make things faster and keep our servers running smoothly.

The goals include

- Enhancing user experience through personalized recommendations.
- Facilitating efficient outfit discovery and creation processes through the personal designer feature.
- Promoting user interaction and engagement through album creation, sharing, and reviewing functionalities.
- Ensuring account management and authentication processes to safeguard user privacy and security.

1.3 Definitions and Abbreviations

Term	**Definition**	**Abbreviations**
Stakeholder	Someone who interacts with the software but is not a developer.	

User	Someone who goes on the website and interacts with it.	
User Interface	Point of interaction between the user and the system.	UI
Artificial Intelligence	A field which combines computer science and robust datasets, to enable problem-solving.	AI
Personal Designer	Generates outfit suggestions that align with the user's preferences.	
Web scrapping	Refers to the legal extraction of data from a website.	
bcryptjs	A password hashing algorithm that enables storing passwords as hashed passwords instead of plaintext.	
JSON Web Token	Authentication and authorization tool.	JWT
Application Programming Interface	A way for two or more computer programs or components to communicate with each other.	API
Middleware	A software that lies between an operating system and the applications running on it.	
Axios	A promise-based HTTP library that lets developers make requests to either their own or a third-party server to fetch data.	
Mongoose	A JavaScript object-oriented programming library that creates a connection between MongoDB and the Node.js JavaScript runtime environment.	
AES-256	AES-256 encryption uses the 256-bit key length to encrypt as well as decrypt a block of messages.	

Table 1 – Definitions and Abbreviations

1.4 References

- Altexsoft. (2023, December 30). *Nonfunctional requirements (nfr):examples, types, approaches*. https://www.altexsoft.com/blog/non-functional-requirements/
- “What Is Artificial Intelligence (AI) ?” *IBM*, www.ibm.com/topics/artificial-intelligence.

1.5 Overview

The FashionFinder website's software requirements document offers a comprehensive overview of the platform's functionalities, guiding stakeholders through its development and implementation phases. Readers can expect a detailed description of FashionFinder's objectives and goals, including its primary functions. The document begins with a broad perspective, outlining the project's scope and constraints, before going into specific interfaces and software requirements crucial for the FashionFinder's website. Additionally, it provides insights on the project requirements through tables. The rest of the SRS contains detailed specifications regarding user interfaces, system interfaces, functional and non-functional requirements, as well as constraints influencing the platform's design and development process. This structured approach ensures clarity and coherence, facilitating effective communication and alignment among stakeholders throughout the FashionFinder project.

2. General Description

This section offers a complete view of the FashionFinder website, offering its architecture, functionality, and impact on user experience. Also, it addresses any constraints and assumptions essential in the AI designer's design process, providing informational context for understanding its capabilities and limitations. Importantly, this section doesn't list specific rules but gives an overall picture that helps understand the detailed rules in later sections.

2.1 Product Perspective

We will be delivering a fully functioning website where the user will communicate with an AI personal designer that will find outfits based on an event the user is going to or on the user’s preferences. Features consist of creating an account, signing in and logging out of the account, creating/deleting albums, adding/removing items from albums, communicating with the personal designer. For more on the functions and features, refer to the 2.2 Product Functions section.

We want users to be able to get to our website and immediately get to shopping. Therefore, upon first arriving to the FashionFinder website, the user will be on our homepage, where they can communicate with the personal designer. We will also make it so that the following features are accessible within 3 clicks from any page within our website: Logging in, logging out, creating an account, using the personal designer, contact customer support, and changing/setting profile preferences.

Upon creating an account, a verification email will be sent to the user's email, where they will click onto a link to confirm that they have attempted to create an account. The user's password will also be encrypted before saving it to the database.

2.2 Product Functions

Users can create an account and sign in or use the website without signing in. Signed in users will have the ability to save brand and budget preferences to their accounts so that they will not need to enter these in when communicating with the personal designer. In the case where they already have these preferences set and enter new preferences to the personal designer, these inputs will override their set preferences. Users will enter in what their clothing preferences are, the personal designer will then ask questions to try and refine down its search and get a better understanding of what the user is looking for. If the user enters an event they are going to, then the personal designer will ask questions to try and gage the theme of the event to better match an outfit that would work for said event.

Once an outfit has been found, the personal designer will then return the outfit back to the user in image and link form. These links will be for each individual item within the outfit and redirect the user to where they can buy that individual item. The personal designer will also conduct price matching, where if there is an identical item on different websites listed at different price points, the cheapest option will be returned to the user. The returned outfit will be sharable, there will be a share option, which will generate a link. When this link is opened, the user will be redirected to a page showing the images and links for each individual item.

Signed in users will have the ability to create albums to which they can then save outfits too. Albums are a way for users to organize different categories of outfits and look at later in the case where they like an outfit but want to continue using the personal designer and have that outfit saved somewhere. For example, a user could create a streetwear album, and then save outfits to it that they deem fall under the streetwear category. These albums will also be sharable via a link.

We will also be implementing a review system. Users will have the option to leave a review after an outfit has been returned to them. This will be a feature that signed in or signed out users will have access to. It will use a star system, where the minimum stars would be 1 and the maximum being 5. 5 being high satisfaction, whereas 1 is low satisfaction.

Contacting customer support will be a feature also available to all users, regardless of login status. Users will fill out a form where they can get in contact with the FashionFinder team for any questions or concerns they have. These messages will be sent via email, and all replies from the FashionFinder team will be an email.

2.3 User Characteristics

Our intended users do not need any technical expertise, experience, or a certain level of education to use FashionFinder. Our main audience would be those who mainly conduct their shopping for clothes online.

2.4 General Constraints

The main constraint in the implementation of FashionFinder falls within how we will be conducting our web scraping. We will only be able to pull data on clothing from websites that legally allow us to web scrape them, which can limit the clothing that can then be returned to the user. Another constraint we have is accurately scraping information from these websites considering they may have their data stored in varying formats.

2.5 Assumptions and Dependencies

FashionFinder does have a few dependencies which, in the case of depreciation, could lead to the website not working or lead to user data being stored with no security. We are using bcryptjs to encrypt the user's passwords before saving them to the database. Along with that we are also using axios to make HTTP requests. JSON Web Token is being used for authentication and authorization for our website. It is also used when checking for accessibility to different pages of our website (if a user has a token, they can access pages where logging in is required). Mongoose is also used for interaction and data modeling with our database.

3. Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

Upon first coming to the FashionFinder website, the user will be on our home page where they can interact with the personal designer. They will see the navigation bar as well as the input box followed by a button to send their message to the personal designer. The navigation bar at the top of the page is present on every page of the website.

The Create Account page follows a similar format in terms of colors and simplicity. It will have 5 input boxes, asking for: email address, first name, last name, password, and confirm password. The first name and last name boxes are optional while the rest are required. There will also be text at the bottom asking the user if the user has an account and this is followed by a link to the login page. Password requirements will also be present on the create account page, these requirements being: At least 1 capital letter, at least 1 number, minimum 8 characters long. Errors will be presented above the input box where the error occurred. Errors being: “Passwords do not match”, “Email already linked with an account”, “Required fields not filled”, “Password requirements not met”.

The login page asks only for the users email and password. Under the “NEXT” button, there is also text asking the user if they do not have an account with FashionFinder, to which they can then press the link which reads, “Create Account”. Errors for this page are “Invalid login credentials”, and “Account does not exist”. Both error messages appear in red text and will be above the email input box.

The contact page is where users will be able to contact us with any questions or concerns, they have about FashionFinder. This page has a message box where users can send us a message. Which is then followed by a button for the user to submit their message.

The albums page is where all the user's albums will appear, this is also where they will create and delete albums. On this page, at the top right will be a button saying, “Create Album”. Upon pressing this button, the user will be prompted to enter a name for the album. Albums will be displayed on the page from left to right, once the screen's width is filled, the next album will be placed on a new line within the page. If the user selects an album, they will be redirected to that specific album where they can view all saved outfits. When the user has found an outfit that they’d like to save, they will get the “Save to album option”. Once pressed, a popup will appear with their existing albums, when they select the one they'd like to save to, the outfit will appear in that album when the album is pressed.

Our website will be responsive to cater toward the different screen sizes of our users. In the case where the users minimize their screen or if they are using a smartphone, the navigation will turn into a hamburger icon, which when pressed, will dropdown a list of the different pages for the user to press and navigate to.

3.1.3 Software Interfaces

We have created APIs for registering, logging in, logging out, and an interface for connecting with the MongoDB Atlas database we are using for this project. The register and login APIs both communicate with the database to check if a user exists or not, in the case where the user does exist, the server will send a message to a client registering, letting them know that there is already an account associated with that email. The login API will check if the user exists within the database and allow the user to login given valid login credentials. Along with this, JSON Web Token will be saved to the user's cookies. The logout API will remove the token from the users' cookies, leading to a successful logout.

3.1.4 Communication Interfaces

FashionFinder uses middleware to manage user access to different pages hosted over HTTP. This middleware acts as a communication interface between the website and access control logic. When a user requests to access a page, the middleware will intercept this request and examine the path the user is trying to access. If the path is one that they have access to, they will be directed to the page. In the case where the user does not have access to the page they wanted, they will be redirected to a path that we have set to public. For example, in the case a signed-out user wants to access the "albums" page, they will be redirected to the login page as they do not have access to this feature.

The middleware must be able to handle increased traffic, therefore we will be using the load balancing strategy to distribute traffic across servers. Considering there might be new features and implementations added, our middleware must also be able to integrate with these changes so that access control is still maintained throughout the system.

3.1.5 System Architecture and Security Measures

Database Configuration and Management with MongoDB Atlas:

MongoDB serves as the primary database solution for our application, providing a secure cloud-based database platform. To ensure the security of our database connection, we will use practices such as AES-256 encryption for data. Also, authentication mechanisms, including username/password authentication, are used to control access to the database. In terms of managing database connections, our application implements robust error-handling mechanisms to handle connection failures to MongoDB Atlas. This involves implementing retry features to ensure database connectivity.

Data Privacy and Compliance with Regulations:

Our application prioritizes the protection of personal user data through various measures. To anonymize personal user data, we utilize techniques such as tokenization, which prevent individual identities from causing issues. Also, sensitive data is encrypted with encryption algorithms such as AES-256. Encryption key management practices are implemented to manage encryption keys and ensure data security.

Middleware Maintenance and Scalability:

Our application's middleware layer is made to be both manageable and scalable to support the application's long-term development. Regular reviews and updates are done to maintain access control rules to consider changes in user roles and security needs. In addition, horizontal scaling approaches are used to optimize the middleware layer for scalability. By doing this, it is ensured that the middleware layer can efficiently manage rising traffic and user load. To maintain smooth connection with the rest of the application architecture, middleware modifications are tested to ensure compatibility with new features or pages.

Fallback or Error Page for Unauthorized Access Attempts:

In case of unauthorized access attempts, our application provides a fallback or error page. The design of these pages includes clear and informative messaging to inform users. Also, contact information for support is displayed on our website, allowing users to seek assistance in resolving access issues. By prioritizing user experience and providing helpful guidance on error pages.

3.2 Functional Requirements

ID	FR1
Title	**Account Creation**
Description	Users can create an account with a minimum amount of information (only email address and password).
Inputs	User-provided credentials (email address and password).
Processing	• Pass email requirements like (including @gmail.com, @hotmail.com, @yahoo.com) • Pass password requirements listed under the password box
Outputs	• show a message to the user saying, "Account has been successfully created." • User-provided credentials (email address and password) will be successfully saved in the database. • Users will be directed to sign in page after successful registration.
Error Handling	Display error messages for invalid inputs through the use of inline validation which will be in red text. Also, the message will include the error and how users can fix it as a way to guide users. • If user enters an email address that's already registered, then a message like "this email already exists, please use a different email address or sign in" will be displayed to user. • If user enters a password that doesn't pass the password rules, then the user will get a message saying, "password must meet requirements, please try again." • If user leaves one of the fields empty, then an "All fields are required" message will pop up. • If the user enters an incorrect password, a message saying, "Incorrect password, please try again" will show up.
Dependencies	N/A

Priority	High

ID	FR2
Title	**Forget/ Change Password**
Description	Users will have the ability to change their password whenever they need to or in the case that they forgot it, they can click on the “forget password” button.
Inputs	User email and new password.
Processing	Verify user email address existence. Prompt for a new password.
Outputs	• Show a message to the user saying, “Password has been successfully changed.” • User’s password will be updated in the database.
Error Handling	• If user enters an email address that’s not already registered in the system, a message will pop up saying “must enter a registered email address.” • If user enters a password that doesn’t match the password rules displayed below the “password” box, then the user will get a message saying, “password must meet requirements.” • If user enters the same password as the old one, a message should pop up saying “cannot use old password. Please try a different one.” • If user leaves one of the fields empty, then an “All fields are required” message will pop up.
Dependencies	FR1
Priority	Mid

ID	FR3
Title	**Signing in**
Description	Users will be able to sign into their account using their email and password.
Inputs	User-provided credentials (email address and password).
Processing	Verify user email address existence and check for completion of verification step.
Outputs	In the case that the user completed the verification email, user will be signed in successfully and can start creating albums.
Error Handling	Display error messages for invalid inputs through the use of inline validation. • If user enters an email address that's not already registered in the system, a message will pop up saying "must enter a registered email address." • If user enters a password that doesn't match the password rules, then the user will get a message saying, "password must meet requirements." • If user enters the wrong password, user will get a message saying, "incorrect password, please try again." • If user leaves one of the fields empty, then an "All fields are required" message will pop up.
Dependencies	FR1 & FR4

Priority	High

ID	FR4
Title	**User Verification**
Description	A verification email will be sent to the user within one minute after registering to verify their email address. Users will not be able to sign in if they have not completed this step.
Inputs	Users' email address used when registering.
Processing	Send a verification email within one minute of creating an account.
Outputs	User verified status. A message saying, "verification completed" or "your email has been verified" will show up for the user to ensure that account has been verified. Once the user is verified, they will be able to sign in.
Error Handling	In the case that the verification wasn't successful, a message will pop up saying "Verification not successful, please close out the page and try again."
Dependencies	FR1
Priority	High

ID	FR5
Title	**Customer Support**
Description	Users will have the ability to send us a message through the “contact us” form.
Inputs	Message input box.
Processing	• Make sure message input field is filled out. • User’s message will be saved in the database. • User’s message will also be sent to the administrators through email. • Once a message is received, the user will get a response within two business days.
Outputs	Confirmation message to user. Display a message saying, “Your message has been sent successfully.” Also, once message is sent, user will get an automated email saying, “Thank you for your inquiry, we will get back to you within two business days.”
Error Handling	In the case that the message wasn’t sent, a message saying, “Error sending message” will be displayed to user.
Dependencies	FR3
Priority	Low

ID	FR6
Title	**Album Creation**
Description	Once users are signed in, they will have the ability to create albums which they can add items to.
Inputs	Album name.
Processing	Store album/ albums.
Outputs	Created album in user's account. "Album created" message is possible but not mandatory to have.
Error Handling	Display error message if album was not created. "Album wasn't created, try again." Or in the case that user used a name that's already used, "an album with this name exits, please try a different one" message will pop up.
Dependencies	FR3
Priority	Mid

ID	FR7
Title	**Album Deletion**
Description	Users will have the ability to delete albums.
Inputs	Album selection.
Processing	Validate selected album.
Outputs	Deleted album from user's account.
Error Handling	Display an error message if album wasn't deleted, "Album was not deleted, try again."
Dependencies	FR3
Priority	Low

ID	FR8
Title	**Album Managment- Adding Items**
Description	Users will have the ability to add items to their albums.
Inputs	Album selection, item selection.
Processing	Validate selected album and item/items.
Outputs	Add item/items to user’s album.
Error Handling	Display an error message if the item wasn’t added to the album, “Item was not added, try again.”
Dependencies	FR3 & FR7
Priority	Low

ID	FR9
Title	**Album Managment- Deleting Items**
Description	Users will have the ability to delete items from their albums.
Inputs	Album selection, item selection.
Processing	Validate selected album and item/items.
Outputs	Delete item/items from user’s album.
Error Handling	Display an error message if the item wasn’t deleted from the album, “Item was not deleted, try again.”
Dependencies	FR3 & FR7
Priority	Low

ID	FR10
Title	**Album Sharing**
Description	Users can share albums with others through the platform.
Inputs	Album selection.
Processing	Validate selected album.
Outputs	Album shared with others.
Error Handling	Display an error message if the album was not shared successfully.
Dependencies	FR3 & FR15
Priority	Low

ID	FR11
Title	**Brand Preference**
Description	Users will be able to pick what brands they would like to shop from.
Inputs	Brand selection.
Processing	Validate selected brand/ brands.
Outputs	Saved brand preferences.
Error Handling	Display an error message if brand preferences were not saved. “Brand preferences were not saved, please try again.”
Dependencies	FR3
Priority	Low

ID	FR12
Title	**Budget Input Box**
Description	Users will be given the ability to specify their budget.
Inputs	Budget selection.
Processing	Validate budget selection.
Outputs	Saved budget preferences.
Error Handling	Display an error message if budget preferences were not saved. "Budget preferences were not saved, please try again."
Dependencies	FR3
Priority	Low

ID	FR13
Title	**Customer Review System**
Description	There will be a review system for users to review the matches they get.
Inputs	User review.
Processing	Store reviews.
Outputs	Review stored in database.
Error Handling	Display a message if something went wrong with the review posting. "Something went wrong, please try again."
Dependencies	FR3, FR14 & FR15
Priority	Low

ID	FR14

Title	**Chat with Personal Designer**
Description	Users will be able to chat with an AI tool to help them find any clothing items they ask for.
Inputs	User input.
Processing	Natural language processing.
Outputs	Outfit/ clothing item is returned to the user.
Error Handling	Display an error message to user if something goes wrong during the chatting process. • If the AI is taking longer than 10 seconds to provide results, then there should be a message saying, “this is taking a little bit longer, we apologize.” • In the case that a match is not found, a message saying, “A match can’t be found, we apologize. Please try again using different key words” will pop up to the user to inform them. • In the case that a match is found but not over the threshold, “A perfect match can’t be found, we have matches that are less similar, press ok to continue.”
Dependencies	FR3
Priority	High

ID	FR15

Title	**Outfit Matching**
Description	Outfits will be displayed to the user in an image and link form.
Inputs	User chatting with AI.
Processing	Matching process.
Outputs	Displayed outfits to the user in the form of an image and a link.
Error Handling	If AI is unable to return something to user, then a message like “Unable to return any matches for you, please try again using simpler words” is necessary.
Dependencies	FR3, FR14
Priority	Mid

ID	FR16
Title	**Outfit Filtering**
Description	Users will be able to filter outfits based on dynamic filters.
Inputs	Users to click on any of the filters.
Processing	Outfits are being chosen based on selected filters.
Outputs	Outfits are displayed based on selected filters.
Error Handling	Display error messages if something goes wrong during the process.
Dependencies	FR3
Priority	Low

3.3 Non-Functional Requirements

ID	NFR1
Title	**Performance**
Description	Defines how fast the software system and its components will respond to user's actions. This system should achieve an average response time of 3 seconds within the user interaction.

ID	NFR2
Title	**Reliability**
Description	Specifies the ability of the system to deliver services as specified without failures. Reliability measures the ability of a system to function correctly. The system should have a reliability of 95% for every month.

ID	NFR3
Title	**Availability**
Description	Describes the system's ability to deliver services when requested regardless of whether it is functioning correctly (according to the requirements) or not. The system should have an availability of 85%.

ID	NFR4
Title	**Security**
Description	Defines how well the system and its data is protected against hackers. The system must ensure a high level of security to protect against unauthorized access, data breaches, and other security threats. This includes encryption, access controls, and regular security audits.

ID	NFR5
Title	**Maintainability**

Description	Defines the time needed for a solution or its component to be fixed, changed or added. The system must be designed and implemented with maintainability in mind for ease of future updates and modifications.

ID	NFR6
Title	**Portability**
Description	Determines if the system or its elements can work in different environments. The system must be capable of running on multiple operating systems, including Windows and macOS without any modifications. The web application should be fully functional on all major web browsers, including Chrome, Firefox, Safari, and Edge, with consistent performance and layout.

3.4 Design Constraints

3.4.1 Hosting Constraints

Hosting Constraint 3.4.1.1
ID: HSC3.4.1.1
Title: Database Connection Limitation
Constraint: The MongoDB database server utilized for the FashionFinder must be configured to handle no more than 10 concurrent connections to maintain optimal performance.

Hosting Constraint 3.4.1.2
ID: HSC3.4.1.2
Title: Storage Limitation
Constraint: The MongoDB Atlas cluster allocated for FashionFinder, used with Next.js, must not exceed 2 gigabytes of storage capacity to prevent potential data loss.

Hosting Constraint 3.4.1.3
ID: HSC3.4.1.3
Title: Memory Allocation Restriction

Constraint: The server hosting the Next.js application must allocate no more than 1 gigabyte of memory for executing application code and handling user requests, ensuring efficient resource use.

Hosting Constraint 3.4.1.4
ID: HSC3.4.1.4
Title: Disk Space Limitation
Constraint: The server hosting the Next.js application must have a maximum disk storage capacity of 1 gigabyte to accommodate files and data without exceeding the available disk space.

3.5 Logical Database Requirements

3.5.1 Database Constraints

Database Constraint 3.5.1.1
ID: DBC3.5.1.1
Title: Data Encryption Requirement
Constraint: All sensitive user data stored in the MongoDB database must be encrypted using AES-256 encryption algorithm to comply with company security policies and regulations.

Database Constraint 3.5.1.2
ID: DBC3.5.1.2
Title: Scalability Requirement
Constraint: The MongoDB database design must support horizontal scalability to accommodate a potential increase in data volume, ensuring efficient performance and availability.

Database Constraint 3.5.1.3
ID: DBC3.5.1.3
Title: Compatibility with Existing Systems
Constraint: The MongoDB database schema and APIs must be compatible with existing backend systems to allow for smooth data exchange.

Database Constraint 3.5.1.4
ID: DBC3.5.1.4

Title: Indexing Limitation

Constraint: Due to hardware limitations, the number of indexed fields per collection in the MongoDB database must not exceed 64, ensuring efficient query performance.

Database Constraint 3.5.1.5

ID: DBC3.5.1.5

Title: Backup and Recovery Protocol

Constraint: The MongoDB database implementation must include regular backups and recovery options, to minimize data loss and ensure business continuity.

Database Constraint 3.5.1.6

ID: DBC3.5.1.6

Title: Data Retention Policy

Constraint: The MongoDB database must enforce a data retention policy regarding the storage and deletion of user data, ensuring compliance and privacy protection.

3.6 Other Requirements

User feedback clarification

The Customer Review System (FR13) serves as a platform for users to share their feedback regarding their interactions with our website. The feedback from each customer will be systematically gathered and stored in the database with the aim of enhancing the overall user experience. Our development team will conduct monthly reviews of the accumulated feedback, implementing any necessary changes or improvements to further refine our website.

Feedback will be categorized into two types: positive and negative. Positive feedback informs us about what we have done well and contributes to our improvement. On the other hand, negative feedback highlights areas where changes or improvements are necessary. We welcome and appreciate both types of feedback.

Examples of positive feedback

1. "I love the user-friendly interface! It makes navigation a breeze."
2. "The customer service team was incredibly helpful and resolved my issue within the expected time. Thank you for your excellent support!"

Examples of negative feedback

1. "The accuracy of AI-generated results could be enhanced for better matches. "
2. " Pages take too long to load, which is frustrating and impacts the overall experience."

3.7 Analysis Models

This is a general data flow diagram that shows the main process of our website through the user who's successfully signed in (Figure 5).

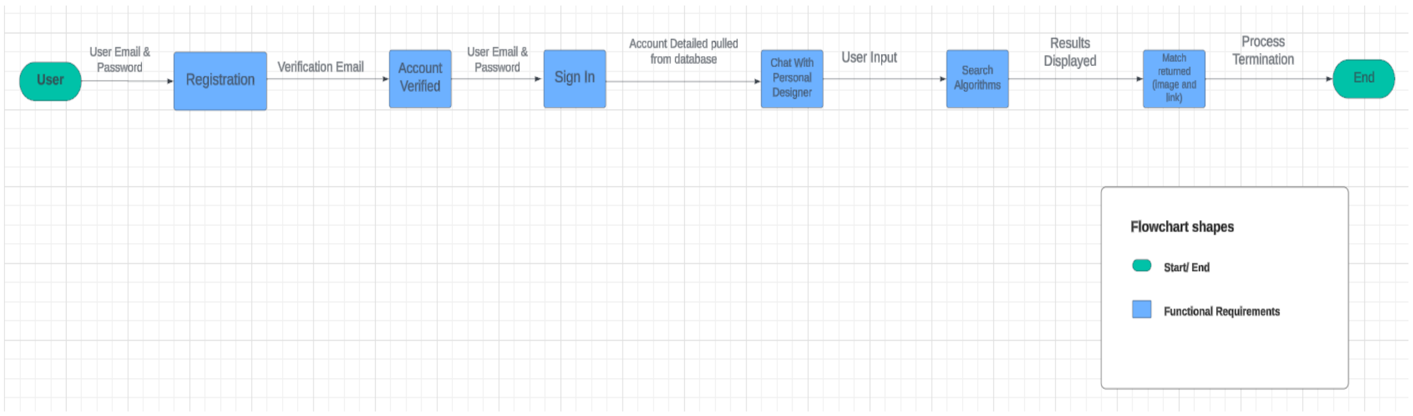

Figure 5