2 Requirement Analysis Expanded

Requirement analysis ensures a deep understanding of the expectations and functionality needed from the system. In our case, the online bookstore had several core and optional features.  
  
Functional Requirements:  
- Book Listings: Display all available books with attributes like title, author, image, and price.  
- Search: Users should be able to search for books by title or author.  
- Filter: Filter options by price, genre, or author (optional in current phase).  
- Cart Functionality: Users can add books to a shopping cart.  
- Book Details Page: A modal or new page displaying additional information.  
  
Non-Functional Requirements:  
- Responsiveness: Pages should work smoothly on various screen sizes.  
- Performance: Pages must load quickly, even with large book lists.  
- Accessibility: Use semantic HTML for screen readers.  
- Maintainability: The codebase should be easy to update and expand.  
  
User Roles and Personas:  
- General User: Browses and searches for books.  
- Admin (future): Will be able to add, remove, or modify book listings.  
  
Data Requirements:  
- Book data stored in an array or JSON file.  
- Each book object contains ID, title, author, price, image path, and description.  
  
Use Cases:  
1. A user visits the homepage and sees a list of books.  
2. They use the search bar to find a specific book.  
3. They add a book to the cart.  
4. They navigate to the cart to review their selection.  
  
User Stories:  
- As a user, I want to browse books easily.  
- As a user, I want to find books by name or author.  
- As a user, I want to keep a list of selected books to purchase.  
  
Tools Used for Requirement Analysis:  
- Google Forms for peer interviews.  
- Online surveys to understand user preferences.  
  
Requirement analysis acted as the blueprint for system design and implementation. It aligned the developers' work with real user needs.  
  
In hindsight, documenting detailed requirements saved us time and effort during development. It allowed us to stay focused on building only the essential features within the given time frame.  
  
With all user and system requirements clearly defined, we moved into the design phase.  
Requirement analysis ensures a deep understanding of the expectations and functionality needed from the system. In our case, the online bookstore had several core and optional features.  
  
Functional Requirements:  
- Book Listings: Display all available books with attributes like title, author, image, and price.  
- Search: Users should be able to search for books by title or author.  
- Filter: Filter options by price, genre, or author (optional in current phase).  
- Cart Functionality: Users can add books to a shopping cart.  
- Book Details Page: A modal or new page displaying additional information.  
  
Non-Functional Requirements:  
- Responsiveness: Pages should work smoothly on various screen sizes.  
- Performance: Pages must load quickly, even with large book lists.  
- Accessibility: Use semantic HTML for screen readers.  
- Maintainability: The codebase should be easy to update and expand.  
  
User Roles and Personas:  
- General User: Browses and searches for books.  
- Admin (future): Will be able to add, remove, or modify book listings.  
  
Data Requirements:  
- Book data stored in an array or JSON file.  
- Each book object contains ID, title, author, price, image path, and description.  
  
Use Cases:  
1. A user visits the homepage and sees a list of books.  
2. They use the search bar to find a specific book.  
3. They add a book to the cart.  
4. They navigate to the cart to review their selection.  
  
User Stories:  
- As a user, I want to browse books easily.  
- As a user, I want to find books by name or author.  
- As a user, I want to keep a list of selected books to purchase.  
  
Tools Used for Requirement Analysis:  
- Google Forms for peer interviews.  
- Online surveys to understand user preferences.  
  
Requirement analysis acted as the blueprint for system design and implementation. It aligned the developers' work with real user needs.  
  
In hindsight, documenting detailed requirements saved us time and effort during development. It allowed us to stay focused on building only the essential features within the given time frame.  
  
With all user and system requirements clearly defined, we moved into the design phase.