Software Requirements Specification

for

FindR

Version 1.1 approved

Prepared by

VARDHAN LOHIA,

LEE JUIN,

SOH SHING HAO,

JERICK LIM KHAI ZHENG,

OI YEEK SHENG

Team FindR, Nanyang Technological University

2022-10-13

Table of Contents

Preface i

1. Table of Contents ii
2. Revision History ii

Introduction 1

1. Purpose 1
2. Document Conventions 1
3. Intended Audience and Reading Suggestions 2
4. Product Scope 2
5. References 2

Overall Description 3

1. Product Perspective 3
2. Product Functions 3
3. User Classes and Characteristics 3
4. Operating Environment 3
5. Design and Implementation Constraints
6. User Documentation
7. Assumptions and Dependencies

External Interface Requirements

1. User Interface
2. Hardware Interfaces
3. Software Interfaces
4. Communications Interfaces

System Features

1. Registration
2. Login
3. Lost Account Retrieval
4. Search Item
5. Recommend Item
6. Add Friend
7. Make Wishlist

Other Nonfunctional Requirements

1. Usability Requirements
2. Reliability Requirements
3. Security Requirements
4. Performance Requirements

Appendix A: Data Dictionary

Appendix B: Analysis Models

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| LEE JUIN | 2022-10-04 | Initial write-up. | 1.0 |
| LEE JUIN | 2022-10-13 | Included section *1.3* and *2.4*. | 1.1 |

# Introduction

## Purpose

The e-commerce industry experienced a surging growth amidst the COVID-19 pandemic. As the world slowly transitions into the post-pandemic era, online shopping has slowly taken root and become a part of the norm within our society.

When shopping online, customers often spend a long time trying to find the best possible deal from various e-commerce platforms. With FindR, we streamlined the process of cross comparison, allowing customers to easily search, compare and purchase listings for an item sold in multiple platforms.

Suppose a user is interested in purchasing the latest iPhone series. They can quickly obtain all the listings on different platforms such as Lazada Singapore and Shopee Singapore with just a single search. The user no longer needs to navigate to different platforms and check the pricing.

FindR also recommends listings of items which may interest the user, based on the user’s search history. Furthermore, a wish list feature is also provided whereby a user may add listings of items they are interested in. Friends of the user may then purchase the listings on the wish list as gift for the user.

Overall, FindR provides a much better overall online shopping experience to our users by automating and streamlining the process of finding the best deals out there.

## Document Conventions

This section describes the conventional standards used throughout this document. It is imperative that all readers pay attention to the standards listed in this section.

**Font**: Times New Roman

**Heading:** Bold, Size 18

**Sub-heading:** Bold, Size 14

**Content:** Italic, Size 12

**Technical Standards:** ISO standard unless explicitly stated otherwise

Refer *Appendix A: Data Dictionary* for the definitions of special terms used throughout this documentation.

## Intended Audience and Reading Suggestions

This document is intended for all stakeholders, which include the users of *FindR* web application, the *FindR* development team, the *FindR* testing team, the project managers and the *FindR* marketing team.

This document begins by stating the purpose of the web application and several conventions used throughout the document. Next, a high-level overview of the application functionalities is introduced, followed by several design constraints and assumptions of the application. Then, the interface requirements of the application are stated. Finally, the document includes a detailed write-up of the system features and non-functional requirements of the application.

All stakeholders are advised to begin by reading section *1.1* Purpose, 1*.2 Document Conventions* and *Appendix A: Data Dictionary* to be familiarized with the purpose of the web application, as well as the documentation standards and technical terms definition used throughout this document.

The *FindR* development team is strongly encouraged to proceed with section *2. Overall Description* to have a high-level understanding of the application functionalities, design, and constraints. Then, section *4. System Features* follows, where the developers will gain a low-level understanding of each system features to be included in the application. Finally, the developers should read section *3. External Interfaces Requirements* and *5. Other Nonfunctional Requirements* to understand the requirements specified for the application to function as desired.

On the other hand, the users of *FindR* web application, the *FindR* testing team, the project managers and the *FindR* marketing team are encouraged to proceed reading this document in sequential order.

## Product Scope

<Provide a short description of the software being specified and its purpose, including relevant benefits, objectives, and goals. Relate the software to corporate goals or business strategies. If a separate vision and scope document is available, refer to it rather than duplicating its contents here.>

## References

1. ISO standard:

<https://www.iso.org/standards.html>

1. Django Framework (v4.1):

<https://docs.djangoproject.com/en/4.1/>

1. ReactJS Library (v18.2.0):

<https://reactjs.org/docs/getting-started.html>

1. REST-styled API:

<https://www.service-architecture.com/articles/web-services/representational-state-transfer-rest.html>

1. PostgreSQL Database (v14.5):

<https://www.postgresql.org/docs/14/index.html>

# Overall Description

## Product Perspective

<Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.>

## Product Functions

<Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, so only a high level summary (such as a bullet list) is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or object class diagram, is often effective.>

## User Classes and Characteristics

<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>

## Operating Environment

This section breaks down the operating environment into two sub-categories —— production environment and development environment.

All stakeholders except the *FindR* development team and the *FindR* testing team are not required to be familiarized with the specifics of the development environment. On the other hand, the *FindR* development team and the *FindR* testing team must be familiarized with both environments.

### Production Environment of FindR

This sub-section describes the setting of which the web application is put into operation.

|  |  |
| --- | --- |
| **Setting** | **Description** |
| HTML Support | The web application requires an internet browser which supports at least HTML5 or above.  *Note that HTML5 is the latest standard of HTML at the time of writing this documentation.* |
| CSS Support | The web application requires an internet browser which supports at least CSS3 or above.  *Note that CSS3 is the latest standard of CSS at the time of writing this documentation.* |
| JavaScript Support | The web application requires an internet browser which supports JavaScript. |

### Development Environment of FindR

This sub-section describes the setting of which the web application is built and tested on during development phase.

|  |  |
| --- | --- |
| **Setting** | **Description** |
| Front-end development using ReactJS. | ReactJS is an open-source front-end JavaScript library maintained by Meta. The *FindR* development team uses ReactJS to build all user interfaces of the web applications as it features reusable components which drastically speed-up the development process.  Edition: ReactJS (version 18.2.0) |
| Back-end development using Django. | Django is an open-source, Python-based web framework maintained by the Django Software Foundation that follows the Models – View – Template (MVT) architectural pattern. The *FindR* development team uses Django to build the web server, as well as the Django REST framework to build the APIs of the web application.  Edition: Django (version 4.1.1) |
| Database using SQLite | SQLite is an embedded database engine that is SQL-compliant. The *FindR* development team plans to use SQLite temporarily before migrating to a scalable database. The *FindR* is considering switching to PostgreSQL.  Edition: SQLite (version 3.39.3) |

## Design and Implementation Constraints

<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer’s organization will be responsible for maintaining the delivered software).>

## User Documentation

<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>

## Assumptions and Dependencies

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>

# External Interface Requirements

## User Interfaces

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

## Hardware Interfaces

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

## Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

## Communications Interfaces

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>

# System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

## System Feature 1

<Don’t really say “System Feature 1.” State the feature name in just a few words.>

4.1.1 Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

4.1.2 Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

4.1.3 Functional Requirements

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-1:

REQ-2:

## System Feature 2 (and so on)

# Other Nonfunctional Requirements

## Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

## Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied.>

## Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

## Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

## Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>

# Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>

Source: http://www.frontiernet.net/~kwiegers/process\_assets/srs\_template.doc