Bartagamen

Team 2 Bartagamenmenschen

Bryce, Janik, Race

**Software Requirements Specification**

**Document**

**Version: (0.0.0)** **Date: (03/04/2022)**

**Table of Contents**

[1 Purpose 3](#_Toc517696147)

[2 Scope 3](#_Toc517696148)

[3 User characteristics 3](#_Toc517696149)

[3.1 Key users 3](#_Toc517696150)

[3.2 Secondary users 4](#_Toc517696151)

[3.3 Unimportant users 4](#_Toc517696152)

[4 Product perspective 4](#_Toc517696153)

[4.1 System Context 5](#_Toc517696154)

[4.2 User interfaces 5](#_Toc517696155)

[4.3 Software interfaces 6](#_Toc517696156)

[4.4 Hardware interfaces and Memory constraints 6](#_Toc517696157)

[4.5 Deployment requirements 7](#_Toc517696158)

[5 Assumptions and Dependencies 7](#_Toc517696159)

[6 Specific requirements 7](#_Toc517696160)

[6.1 System Functional Requirements 8](#_Toc517696161)

[6.2 Logical Database Requirements 8](#_Toc517696162)

[6.3 Software System Attributes 8](#_Toc517696163)

[6.3.1 Usability 8](#_Toc517696164)

[6.3.2 Performance 8](#_Toc517696165)

[6.3.3 Reliability/Dependability 8](#_Toc517696166)

[6.3.4 Security 8](#_Toc517696167)

[6.3.5 Maintainability 9](#_Toc517696168)

# 1 Purpose

*This section describes*

1. *the business background (or problem context).*
2. *why a new system is needed. You should describe the existing problems, issues, or deficiencies in the business where the use of the new system can bring business values (by addressing the problems, issues, or deficiencies).*

# 2 Scope

*Describe the scope of the software by clearly list the desired objectives. This sets up a scope for the new system to be developed. You may choose to address only a few of the problems, issues, or deficiencies identified in the business. In particular, you should*

1. *Give an appropriate name to the system (e.g., PSU Campus Map, Super Team Editor, etc.) and reference it by name in the rest of the document;*
2. *Explain goals (what the software will do);*
3. *Describe the application of the software, including potential benefits.*

# 3 User characteristics

*Identifying the potential users of the product. Describe general characteristics of the intended groups of users (stakeholders) of the product, especially focusing on characteristics that may influence usability, such as educational level, experience, disabilities, and technical expertise.*

## 3.1 Key users

*They are critical to the continued success of the product. Give greater importance to requirements generated by this category of user.*

* *User role responsibilities: what to do with the product.*
* *Subject matter experience: Summarizes the users’ knowledge of the business (domain). Rate as novice, journeyman, or master.*
* *Technological experience: Describes the users’ experience with relevant technology. Rate as novice, journeyman, or master.*
* *Other user characteristics: Describe any characteristics of the users that have an effect on the requirements and eventual design of the product. For example:* 
  + *Physical abilities/disabilities*
  + *Intellectual abilities/disabilities*
  + *Attitude toward technology*
  + *Education*
  + *Linguistic skills*
  + *Age group*
  + *Gender*

## 3.2 Secondary users

*They will use the product, but their opinion of it has no effect on its long-term success. Where there is a conflict between secondary users’ requirements and those of key users, the key users take precedence.*

* *User role responsibilities: what to do with the product.*
* *Subject matter experience: Summarizes the users’ knowledge of the business (domain). Rate as novice, journeyman, or master.*
* *Technological experience: Describes the users’ experience with relevant technology. Rate as novice, journeyman, or master.*
* *Other user characteristics: Describe any characteristics of the users that have an effect on the requirements and eventual design of the product. For example:* 
  + *Physical abilities/disabilities*
  + *Intellectual abilities/disabilities*
  + *Attitude toward technology*
  + *Education*
  + *Linguistic skills*
  + *Age group*
  + *Gender*

## 3.3 Unimportant users

*This category of user is given the lowest priority. It includes infrequent, unauthorized, and unskilled users, as well as people who misuse the product.*

* *User role responsibilities: what to do with the product.*
* *Subject matter experience: Summarizes the users’ knowledge of the business (domain). Rate as novice, journeyman, or master.*
* *Technological experience: Describes the users’ experience with relevant technology. Rate as novice, journeyman, or master.*
* *Other user characteristics: Describe any characteristics of the users that have an effect on the requirements and eventual design of the product. For example:* 
  + *Physical abilities/disabilities*
  + *Intellectual abilities/disabilities*
  + *Attitude toward technology*
  + *Education*
  + *Linguistic skills*
  + *Age group*
  + *Gender*

# 4 Product perspective

## 4.1 System Context

Diagram

Description automatically generated

## 4.2 User interfaces

[SRS:UX:1] The application shall support mobile phones in a portrait orientation.

[SRS:UX:2] The application shall have a constant navigation bar at the bottom of the screen.

[SRS:UX:3] The navigation bar shall have buttons that redirect to the Home Screen, Pet List Screen, and the Monthly Calendar View Screen.

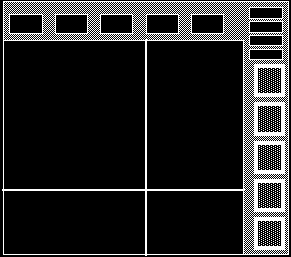
[SRS:UX:4] The Pet List Screen shall auto highlight the day based on current mobile calendar info.

*Specify the required characteristics of each interface between the software product and its users. Keep in mind that we are specifying requirements, so we here only care about the requirements on interfaces (this is NOT the place to put user interface designs by the software developers). For each requirement on an interface, it may include:*

* *Required screen formats (text only or multi-media),*
* *Required page or window layouts,*
* *Required content of any reports, menus, error messages,*
* *Required backup and recovery operations, or*
* *Required user-initiated operations or function keys.*

*Some examples:*

1. *It is required that the system provides the option of long and short error messages.*
2. *It is required that the system allows users to save work-in-progress and load previous works*
3. *It is required that the system interfaces have the following screen layout:*



## 4.3 Software interfaces

*Specify the use of other required software systems (e.g., a data management system, an operating system, or a mathematical package), and interfaces with other software systems.*

*[Note: This is only for* ***customer-specified systems*** *that you* ***have*** *to interact with. Choosing SQL Server 7 as a DB without a customer requirement is a Design choice, not a requirement. This is a subtle but important point to writing good requirements and not over-constraining the design.]*

* *For each required software product, if possible, specify: a) Software name; b) Software specification number; c) Software version number; d) Source.*
* *For each interface specify the required message formats. If the interface is well-documented elsewhere, provide a reference to the document defining the interface.*

*For instance if your customer uses SQL Server 7 DB and you are required to use that, then you need to specify*

* *The system must use SQL Server 7 as its database component.*
* *Communication with the DB is through ODBC connections.*

## 4.4 Hardware interfaces and Memory constraints

*Specify the logical characteristics of each interface between the software and the hardware elements of the system. This includes configuration characteristics (number of ports, instruction sets, etc.). It also covers such matters as what devices are to be supported, how they are to be supported, and protocols.*

*Specify required characteristics and constraints on primary and secondary memory.*

*For example,*

* *The terminal output should support full-screen of 1080X800 pixel.*
* *Because the target devices have only 2G of RAM, it is required that the design footprint should not exceed 2G.*

*This is mainly applicable to embedded systems. If this is not applicable to your course project, simply remove this subsection. Don’t just make up something here.*

## 4.5 Deployment requirements

*Specification of the deployment environment that is required for the installation and operation of the software.*

*If any modifications to the customer’s work area would be required by your system, then document that here. Any equipment the customer would need to buy or any software setup that needs to be done so that your system will install and operate correctly should be documented here.*

*This could be hardware-specific, For instance, “A 100Kw backup generator and 10000 BTU air conditioning system must be installed at the user site prior to software installation”.*

*This could also be software-specific like, “New data tables created for this system must be installed on the company’s existing DB server and populated prior to system activation.”*

# 5 Assumptions and Dependencies

*List each of the factors that affect the requirements stated in the SRS. These factors are not design constraints on the software but any changes to these factors can affect the requirements in the SRS.*

*For example, an assumption may be that a specific operating system will be available on the hardware designated for the software product. If, in fact, the operating system is not available, the SRS would then have to change accordingly.*

# 6 Specific requirements

*This is a specification, a designer should be able to read this spec and build the system without bothering the customer again.*

* *Specify all of the software requirements to a level of detail sufficient to enable designers to design a software system to satisfy those requirements.*
* *Specify all of the software requirements to a level of detail sufficient to enable testers to test that the software system satisfies those requirements.*
* *Each requirement should be uniquely identifiable for traceability. Usually, they are numbered R.1, R.1.1, R.1.2.1 etc. so that each can be cross-referenced in other documents.*
* *Each requirement should also be testable. Avoid imprecise statements like, “The system shall be easy to use.” What is easy to use?*

*Use proper terminology:*

* *A required, must have feature: The system shall…*
* *An optional, nice-to-have feature that may never make it to implementation: The system may…*

*Avoid over-constraining your design. Do not require specific software packages, etc., unless the customer specifically requires them.*

*Avoid examples: Don’t say things like, “The system shall accept configuration information such as name and address.” The designer doesn’t know if that is the only two data elements or if there are 200. List every piece of information that is required so the designers can build the right UI and data tables.*

## 6.1 System Functional Requirements

*Provide a list of the major functions that the software will perform. For example, an SRS for an accounting program may use this part to address customer account maintenance, customer statement, and invoice preparation without mentioning the vast amount of detail that each of those functions requires.*

*As mentioned in lecture, you should try the best to use the User Story Template to document system functional requirements.*

## 6.2 Logical Database Requirements

*Specify the logical requirements for any information that is to be placed into a database, including data entities and their relationships, and Integrity constraints.*

*If the customer provided you with data models, those can be presented here. ER diagrams (or static class diagrams) can be useful here to show entity relationships. Remember a diagram is worth a thousand words of confusing text.*

## 6.3 Software System Attributes

*Specify non-functional requirements, that is, the required attributes of the software product.*

***For each requirements in this section, make sure it is testable***

### 6.3.1 Usability

*Usability requirements for the software system include measurable effectiveness and satisfaction criteria in specific contexts of user interactions.*

### 6.3.2 Performance

*Specify both the static and the dynamic numerical requirements placed on the software or on human interaction with the software as a whole.*

* *Static numerical requirements (capacity) may include the following the number of simultaneous users to be supported; amount and type of information to be handled.*
* *Dynamic numerical requirements may include, for example, the numbers of transactions and tasks and the amount of data to be processed within certain time periods for both normal and peak workload conditions.*

*The performance requirements should be stated in measurable terms. For example, 95 % of the transactions shall be processed in less than 1 second.*

### 6.3.3 Reliability/Dependability

*Specify the factors required to establish the required reliability/dependability of the software system at time of delivery.*

### 6.3.4 Security

*Specify the requirements to protect the software from accidental or malicious access, modification, or destruction. Specific requirements in this area could include the need to:*

*1) Utilize certain cryptographic techniques;*

*2) Keep specific log or history data sets;*

*3) Authenticate system users;*

*4) Check data integrity for critical variables;*

*5) Assure data privacy.*

### 6.3.5 Maintainability

*Specify attributes of software that relate to the ease of maintenance of the software itself. These may include requirements for certain modularity, interfaces, or complexity limitation. Requirements should not be placed here just because they are thought to be good design practices.*