**Software Requirements Specification Template**

The following annotated template shall be used to complete the Software Requirements Specification (SRS). The instructor must approve any modifications to the overall structure of this document.

**Template Usage:**

Text contained within angle brackets (‘<’, ‘>’) shall be replaced by your project-specific information and/or details. For example, <Project Name> will be replaced with either ‘Smart Home’ or ‘Sensor Network’.

Italicized text is included to briefly annotate the purpose of each section within this template. This text should not appear in the final version of your submitted SRS.

This cover page is not a part of the final template and should be removed before your SRS is submitted.

**Acknowledgements:**

Sections of this document are based upon the IEEE Guide to Software Requirements Specification (ANSI/IEEE Std. 830-1984).

Course Advising Application

Software Requirements Specification

1.0

29 January, 2019

David Herrington

Team Leader/Programmer

Simran Bhamra

Designer/Programmer

Michael Fishler

Programmer/Researcher

\*All members should be listed here

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Description** | **Author** | **Comments** |
|  | Version 1 |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Document Approval**

The following Software Requirements Specification has been accepted and approved by the following:

|  |  |  |  |
| --- | --- | --- | --- |
| **Signature** | **Printed Name** | **Title** | **Date** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

**1. Introduction 1**

1.1 Purpose 1

1.2 Scope 1

1.3 Definitions, Acronyms, and Abbreviations 1

1.4 Overview 1

**2. General Description 1**

2.1 Product Perspective 1

2.2 Product Functions 2

2.3 Users and Characteristics 2

2.4 General Constraints 2

2.5 Assumptions and Dependencies 2

2.6 Operating Environment 2

**3. Specific Requirements 2**

3.1 External Interface Requirements 3

*3.1.1 User Interfaces 3*

*3.1.2 Hardware Interfaces 3*

*3.1.3 Software Interfaces 3*

*3.1.4 Communications Interfaces 3*

3.2 Functional Requirements 3

*3.2.1 <Functional Requirement or Feature #1> 3*

*3.2.2 <Functional Requirement or Feature #2> 3*

3.3 Use Cases 4

*3.3.1 Use Case #1 4*

*3.3.2 Use Case #2 4*

3.4 Non-Functional Requirements 4

*3.5.1 Performance 5*

*3.5.2 Reliability 5*

*3.5.3 Availability 5*

*3.5.4 Security 5*

*3.5.5 Maintainability 5*

*3.5.6 Portability 5*

3.5 Design Constraints 5

3.6 Logical Database Requirements 5

3.7 Other Requirements 5

**4. Analysis Models 5**

4.1 Sequence Diagrams 6

**5. Change Management Process 6**

**A. Appendices 6**

A.1 Appendix 1 7

A.2 Appendix 2 7

**List of Figures**

[Figure 1: Data Flow Diagram Example 1 6](#_2u6wntf)

[Figure 2 Data Flow Diagram Example 2 6](#_19c6y18)

**List of Tables**

[Table 1: Use case 1 4](#_49x2ik5)

# 1. Introduction

<The introduction to the Software Requirement Specification (SRS) document should provide an overview of the complete SRS document. While writing this document please remember that this document should contain all of the information needed by a software engineer to adequately design and implement the software product described by the requirements listed in this document.>

## 1.1 Purpose

<What is the purpose of this SRS and the (intended) audience for which it is written?>

## 1.2 Scope

<This subsection should:

(1) Identify the software product(s) to be produced by name; for example, Host DBMS, Report Generator, etc.

(2) Explain what the software product(s) will, and, if necessary, will not do

(3) Describe the application of the software being specified. As a portion of this, it should:

(a) Describe all relevant benefits, objectives, and goals as precisely as possible. For example, to say that one goal is to provide effective reporting capabilities is not as good as saying parameter-driven, user-definable reports with a 2 hour turnaround and on-line entry of user parameters.

(b) Be consistent with similar statements in higher-level specifications (for example, the System Requirement Specification), if they exist. What is the scope of this software product?>

## 1.3 Definitions, Acronyms, and Abbreviations

<This subsection should provide the definitions of all terms, acronyms, and abbreviations required to properly interpret the SRS. This information may be provided by reference to one or more appendixes in the SRS or by reference to other documents.>

## 1.4 Overview

<This subsection should:

(1) Describe what the rest of the SRS contains

(2) Explain how the SRS is organized.>

# 2. General Description

<This section of the SRS should describe the general factors that affect the product and its requirements. This section does not state specific requirements; it only makes those requirements easier to understand.>

## 2.1 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.>

## 2.2 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 is often effective.>

## 2.3 Users 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.>

## 2.4 General Constraints

<This subsection of the SRS should provide a general description of any other items that will limit the designer/developer’s options for designing/developing the system.>

## 2.5 Assumptions and Dependencies

<This subsection of the SRS should list each of the factors that affect the requirements stated in the SRS. These factors are not design constraints on the software but are, rather, any changes to them that can affect the requirements in the SRS. For example, an assumption might 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.>

## 2.6 Operating Environment

<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>

# 3. Specific Requirements

<This will be the largest and most important section of the SRS. The customer requirements are embodied within Section 2 (functions), but this section will give the D-requirements that are used to guide the project’s software design, implementation, and testing.

Each requirement in this section should be:

* Correct
* Traceable (both forward and backward to prior/future artifacts)
* Unambiguous
* Verifiable (i.e., testable)
* Prioritized (with respect to importance and/or stability)
* Complete
* Consistent (with other requirements)
* Uniquely identifiable (usually via numbering like 3.4.5.6)

Attention should be paid to the carefully organize the requirements presented in this section so that they may easily accessed and understood. Furthermore, this SRS is not the software design document, therefore one should avoid the tendency to over-constrain (and therefore design) the software project within this SRS.>

## 3.1 External Interface Requirements

### 3.1.1 User Interfaces

### 3.1.2 Hardware Interfaces

### 3.1.3 Software Interfaces

### 3.1.4 Communications Interfaces

## 3.2 Functional Requirements

This section describes specific features of the software project. If desired, some requirements may be specified in the use-case format and listed in the Use Cases Section.

### 3.2.1 Log in via Google Account

**ID: FR1**

#### 3.2.1.1 Description and Priority

This feature will allow both students and advisors to log in with a Google account or email link, this will

simplify the sign in process. This feature is high priority (9) as a user must sign in to use the program and use the

push notifications.

#### 3.2.1.2 Stimulus/Response Sequences

**Option 1 - Google login**

Stimulus: User opens the application and is prompted to “Sign in with Google”.

Response: The user is taken to the Google sign in page and fill in their details.

Stimulus: The user is authenticated with the application via the Google login.

Response: The user is taken to their homepage, if this is the first time the user has logged on then the user will be

taken to the course survey or professor home based on the account type they have selected.

**Option 2 - Email login**

Stimulus: User opens the application and does not have a Google account and selects “Sign-in with email”.

Response: The user enters their email and a link is sent to the email account.

Stimulus: The user logins into the account via the link in their account and is sent to the survey or the professor

home based on the account type they have selected.

#### 3.2.1.3 Functional Requirements

* In order for the end user to use this feature they must have a Google account or access to an email address and a stable internet connection.
* To execute the above use cases, Google Firebase authentication must be used both for Google account logins and email link logins.
* There is no case where users will not be able to login as each student and professor is associated with a DeSales University email address. However, unexpected errors in logins will be handled by the Google authenticator.

### 3.2.2 Students can see and book appointments with advisors in their major/minor.

**ID: FR2**

#### 3.2.2.1 Description and Priority

Students are allowed to schedule time during advising week with their advisors for various purposes. Students are only to see advisors within their major/minor, depending on what the student selected as his/her major at the initialization stage. This is be a high (9) priority.

#### 3.2.2.2 Stimulus/Response Sequences

Stimulus: User requests to schedule an appointment.

Response: System shows advisor’s availability

Stimulus: User selects an open time slot

Response: System shows confirmation window with additional meeting configuration settings

Stimulus: User enters in reason for and confirms the appointment

#### 3.2.2.3 Functional Requirements

* Point and click UI for carrying out task - this is the only input for this task
* Calendar views for displaying schedule/time slots
* Pop-up windows for confirming appointments
* Text input for entering appointment reason

### 3.2.3 Exporting mock schedule to PDF.

**ID: FR3**

#### 3.2.3.1 Description and Priority

The user should be able to download a PDF copy of their appointments and class schedule. Student users should

be able to download the PDF version of the current schedule that they have made while professor users should be able to download PDF versions of their appointments for the day/week depending on what view they have picked and select students schedules. This function is low priority (3).

#### 3.2.3.2 Stimulus/Response Sequences

**Option 1 - Student schedule download**

Stimulus: The student user clicks to export their class schedule.

Response: The PDF version of the class schedule is downloaded to the device.

**Option 2 - Professor appointment download**

Stimulus: The professor user selects their preferred view (day/week) in the calendar and clicks to export.

Response: The PDF version of the appointment schedule is downloaded to the device.

**Option 3 - Professor student schedule download**

Stimulus: The professor user selects the student they want to view and clicks to export their class schedule.

Response: The PDF version of the appointment schedule is downloaded to the device.

#### 3.2.3.3 Functional Requirements

* To use this feature the user must allow downloads from the web application.
* Software capabilities required are TBD.

### 3.2.4 Generating course suggestions based on major/minor.

**ID: FR4**

#### 3.2.4.1 Description and Priority

After a student inputs his/her major and/or minor, and after the student inputs completed courses, the program will display courses that the student needs to take in order to graduate, and/or courses being offered for the next semester.

#### 3.2.4.2 Stimulus/Response Sequences

Stimulus: User requests to see suggested courses

Response: System displays list of courses that user needs/may like to take.

* This functional requirement depends on the action generated by other functional requirements (inputting major/minor and completed courses.)

#### 3.2.4.3 Functional Requirements

* Point and click UI for selecting feature

### 3.2.5 Push appointment reminders to students and advisors.

**ID: FR5**

#### 3.2.5.1 Description and Priority

Since the user is logging in with their Google account, the user can opt-in for appointment reminders sent to their email the morning before the appointment.

#### 3.2.5.2 Stimulus/Response Sequences

Stimulus: Trigger set to go off morning of the appointment.

Response: Email will be sent to users gmail with appointment reminder information.

#### 3.2.5.3 Functional Requirements

* When making an appointment, we’ll need a check box for sending reminders.
* Program will have to send an email the morning of the appointment with reminder information.

##### 

### 

### 3.2.6 Creating appointment availability

**ID: FR6**

#### 3.2.5.1 Description and Priority

The professor user should be able to set dates and time sessions where they are available for meetings as well as setting a time limit for the meeting. For example, the professor would be able to set each slot as 15 minutes as well as limit how many sessions each student can have. This is a high(9) priority function.

#### 3.2.5.2 Stimulus/Response Sequences

Stimulus: The user enters information via the UI to specifying, dates, time blocks, meeting time limits, and meeting

amount limits they would like.

Response: The software will create a calendar where available appointments become viewable to the students and

they can select a meeting time.

Stimulus: A student user has selected and confirmed an appointment.

Response: The appointment time block on the professors calendar will be shown as confirmed (by the change in

color) and show the students name and student ID.

#### 3.2.5.3 Functional Requirements

* The user and professor must enter/ select valid appointment times.
* The UI will be designed so that the professor cannot enter invalid times or dates with the use of drop down menus.
* If the error occurs that a student user requires more meetings than are allowed by the professor then they will be prompted to privately email the professor.
* If a student cannot make any time slot provided the student user will be prompted by the system to privately email the professor to work out a suitable appointment time.
* The software must provide a fully functional calendar system for both users.

### 3.2.7 Deletion of appointments

**ID: FR7**

#### 3.2.5.1 Description and Priority

Both the professor and student user should be able to delete appointments, this will be a high (6) priority feature.

#### 3.2.5.2 Stimulus/Response Sequences

Stimulus: A user deletes an appointment.

Response: The appointment will be deleted from the student and professor calendar.

#### 3.2.5.3 Functional Requirements

* To execute the use case the calendaring system must work.
* The professor and student appointment calendars must be able to sync, specific technical requirements are TBD.

##### 

### 3.2.8 Creating a mock schedule.

**ID: FR8**

#### 3.2.8.1 Description and Priority

Both the professor and student user should be able to make mock schedules, being able to send the mock schedule to the advisor or to the student. This will be a high (7) priority feature.

#### 3.2.8.2 Stimulus/Response Sequences

Stimulus: A user selects courses from the catalogue for the next semester.

Response: The courses are organized into a schedule template.

#### 3.2.8.3 Functional Requirements

* The course selection catalog must work.
* Must have a function that takes the course number and time slot data from the course catalogue and converts it to a weekly calendar view.

### 3.2.9 Students should only see availability for advisors in their department.

**ID: FR9**

#### 3.2.9.1 Description and Priority

In account settings, an option will allow them to select their major/minor. Doing this will only show advisors in their department. This prevents the student from creating appointments with advisors from the wrong departments. This feature has medium priority (5).

#### 3.2.9.2 Stimulus/Response Sequences

Stimulus: A student select their major/minor in account settings.

Response: When making an appointment or communicating with the advisor, the program will only show advisors in the correct department.

#### 3.2.9.3 Functional Requirements

* Must have a setting that allows the user to select their major/minor.
* Must have a function that takes students selected major/minor and filters through the complete list of advisors, only showing the correct advisors for the specific major/minor.

### 

### 3.2.10 Students will be able to select classes they have already taken.

**ID: FR10**

#### 3.2.10.1 Description and Priority

In account settings, an option will allow the student to select and save any classes that they have already taken. This will filter the classes needed for the students major and exclude any classes that have been marked as taken. This will be a high (7) priority feature.

#### 3.2.10.2 stimulus/Response Sequences

Stimulus: A student select classes that they have already taken in settings.

Response: When a student is searching for classes, the classes that have been marked as taken will not show up in the course search.

#### 3.2.10.3 Functional Requirements

* The course selection catalog must work.
* Must have the ability to save completed courses for the student, that allows the program to reference when searching for classes.

### 3.2.11 Advisors will be able to see appointments scheduled for a day/week.

**ID: FR11**

#### 3.2.11.1 Description and Priority

The advisor will be able to see an organized calendar view of appointments scheduled within a specified period of time. He/she will be able to click on appointments to display additional information like the appointment meeting. This requirement has a medium priority (6).

#### 3.2.11.2 Stimulus/Response Sequences

Stimulus: The advisor requests to view his/her schedule

Response: The program generates a list of appointments for the selected time frame.

Stimulus: The advisor toggles the time frame

Response: The program regenerates the list of appointments in the selected view.

Stimulus: The advisor clicks on an appointment.

Response: The program displays meeting information like time and reason.

#### 3.2.11.3 Functional Requirements

* Multiple calendar views for each time structure in the advisor would like to view his/her schedule
* Communication between students and advisor to populate advisor schedule with created appointments
* Point and Click UI for navigating through the task

### 3.2.12 Data will be saved between app sessions.

**ID: FR12**

#### 3.2.12.1 Description and Priority

On both the advisor and student end, important information like course selections and appointment schedules will be saved each time the user closes the application. This will be stored in a NoSQL database located on the Firebase Server. This has a very high priority (10).

#### 3.2.12.2 Stimulus/Response Sequences

Stimulus: User closes the application

Response: The program writes any new/modified information to the designated database.

#### 3.2.12.3 Functional Requirements

* Database creation and functionality with fast and automatic read/write time
* Firebase implementation will simplify this task

### 3.2.13 Modification of appointments

**ID: FR13**

#### 3.2.13.1 Description and Priority

Both the professor and student user should be able to modify appointments, this will be a high (6) priority feature.

#### 3.2.13.2 Stimulus/Response Sequences

Stimulus: A user modifies an appointment.

Response: The appointment will be modified on the student and professor calendar.

#### 3.2.13.3 Functional Requirements

* To execute the use case the calendaring system must work.
* The professor and student appointment calendars must be able to sync, specific technical requirements are TBD.

## 3.3 Use Cases

### 3.3.1 Use Case #1

|  |  |
| --- | --- |
| **Use Case Name** | Advisor sets up appointments |
| **Reference** | Section 3.2.6, FR 6 |
| **Trigger** | The user clicks to add new appointments. |
| **Precondition** | The Web is displayed with grids for searching |
| **Basic Path** | 1. The Reader chooses how to search the Web site. The choices are by Author, by Category, and by Keyword. 2. If the search is by Author, the system creates and presents an alphabetical list of all authors in the database. In the case of an article with multiple authors, each is contained in the list. 3. The Reader selects an author. 4. The system creates and presents a list of all articles by that author in the database. 5. The Reader selects an article. 6. The system displays the Abstract for the article. 7. The Reader selects to download the article or to return to the article list or to the previous list. |
| **Alternative Paths** | In step 2, if the Reader selects to search by category, the system creates and presents a list of all categories in the database.   1. The Reader selects a category. 2. The system creates and presents a list of all articles in that category in the database. Return to step 5.   In step 2, if the Reader selects to search by keyword, the system presents a dialog box to enter the keyword or phrase.   1. The Reader enters a keyword or phrase. 2. The system searches the Abstracts for all articles with that keyword or phrase and creates and presents a list of all such articles in the database. Return to step 5. |
| **Postcondition** | The selected article is downloaded to the client machine. |
| **Exception Paths** | The Reader may abandon the search at any time. |
| **Other** | The categories list is generated from the information provided when article are published and not predefined in the Online Journal database. |

*Table 1: Use case 1*

### 3.3.2 Use Case #2

|  |  |
| --- | --- |
| **Use Case Name** | Advisors can view appointments |
| **Reference** | Section 2.2.1, Section 7.1 |
| **Trigger** | The Reader assesses the Online Journal Website |
| **Precondition** | The Web is displayed with grids for searching |
| **Basic Path** | 1. The Reader chooses how to search the Web site. The choices are by Author, by Category, and by Keyword. 2. If the search is by Author, the system creates and presents an alphabetical list of all authors in the database. In the case of an article with multiple authors, each is contained in the list. 3. The Reader selects an author. 4. The system creates and presents a list of all articles by that author in the database. 5. The Reader selects an article. 6. The system displays the Abstract for the article. 7. The Reader selects to download the article or to return to the article list or to the previous list. |
| **Alternative Paths** | In step 2, if the Reader selects to search by category, the system creates and presents a list of all categories in the database.   1. The Reader selects a category. 2. The system creates and presents a list of all articles in that category in the database. Return to step 5.   In step 2, if the Reader selects to search by keyword, the system presents a dialog box to enter the keyword or phrase.   1. The Reader enters a keyword or phrase. 2. The system searches the Abstracts for all articles with that keyword or phrase and creates and presents a list of all such articles in the database. Return to step 5. |
| **Postcondition** | The selected article is downloaded to the client machine. |
| **Exception Paths** | The Reader may abandon the search at any time. |
| **Other** | The categories list is generated from the information provided when article are published and not predefined in the Online Journal database. |

*Table 2: Use case 2*

### 3.3.3 Use Case #3

|  |  |
| --- | --- |
| **Use Case Name** | Delete Appointments |
| **Reference** | Section 2.2.1, Section 7.1 |
| **Trigger** | User clicks delete appointments. |
| **Precondition** | The Web is displayed with grids for searching |
| **Basic Path** | 1. The Reader chooses how to search the Web site. The choices are by Author, by Category, and by Keyword. 2. If the search is by Author, the system creates and presents an alphabetical list of all authors in the database. In the case of an article with multiple authors, each is contained in the list. 3. The Reader selects an author. 4. The system creates and presents a list of all articles by that author in the database. 5. The Reader selects an article. 6. The system displays the Abstract for the article. 7. The Reader selects to download the article or to return to the article list or to the previous list. |
| **Alternative Paths** | In step 2, if the Reader selects to search by category, the system creates and presents a list of all categories in the database.   1. The Reader selects a category. 2. The system creates and presents a list of all articles in that category in the database. Return to step 5.   In step 2, if the Reader selects to search by keyword, the system presents a dialog box to enter the keyword or phrase.   1. The Reader enters a keyword or phrase. 2. The system searches the Abstracts for all articles with that keyword or phrase and creates and presents a list of all such articles in the database. Return to step 5. |
| **Postcondition** | The selected article is downloaded to the client machine. |
| **Exception Paths** | The Reader may abandon the search at any time. |
| **Other** | The categories list is generated from the information provided when article are published and not predefined in the Online Journal database. |

*Table 3: Use case 3*

### 3.3.4 Use Case #4

|  |  |
| --- | --- |
| **Use Case Name** | Advisor can view the students in their advising list. |
| **Reference** | Section 2.2.1, Section 7.1 |
| **Trigger** | The Reader assesses the Online Journal Website |
| **Precondition** | The Web is displayed with grids for searching |
| **Basic Path** | 1. The Reader chooses how to search the Web site. The choices are by Author, by Category, and by Keyword. 2. If the search is by Author, the system creates and presents an alphabetical list of all authors in the database. In the case of an article with multiple authors, each is contained in the list. 3. The Reader selects an author. 4. The system creates and presents a list of all articles by that author in the database. 5. The Reader selects an article. 6. The system displays the Abstract for the article. 7. The Reader selects to download the article or to return to the article list or to the previous list. |
| **Alternative Paths** | In step 2, if the Reader selects to search by category, the system creates and presents a list of all categories in the database.   1. The Reader selects a category. 2. The system creates and presents a list of all articles in that category in the database. Return to step 5.   In step 2, if the Reader selects to search by keyword, the system presents a dialog box to enter the keyword or phrase.   1. The Reader enters a keyword or phrase. 2. The system searches the Abstracts for all articles with that keyword or phrase and creates and presents a list of all such articles in the database. Return to step 5. |
| **Postcondition** | The selected article is downloaded to the client machine. |
| **Exception Paths** | The Reader may abandon the search at any time. |
| **Other** | The categories list is generated from the information provided when article are published and not predefined in the Online Journal database. |

*Table 4: Use case 4*

…

## 3.4 Non-Functional Requirements

Non-functional requirements may exist for the following attributes. Often these requirements must be achieved at a system-wide level rather than at a unit level. State the requirements in the following sections in measurable terms (e.g., 95% of transaction shall be processed in less than a second, system downtime may not exceed 1 minute per day, > 30 day MTBF value, etc.).

### 3.5.1 Performance

### 3.5.2 Reliability

### 3.5.3 Availability

### 3.5.4 Security

<Example: The server on which the Online Journal resides will have its own security to prevent unauthorized *write*/*delete* access. There is no restriction on *read* access. The use of email by an Author or Reviewer is on the client systems and thus is external to the system. The PC on which the Article Manager resides will have its own security. Only the Editor will have physical access to the machine and the program on it. There is no special protection built into this system other than to provide the editor with *write* access to the Online Journal to publish an article.>

### 3.5.5 Maintainability

### 3.5.6 Portability

## 3.5 Design Constraints

<Specify design constrains imposed by other standards, company policies, hardware limitation, etc. that will impact this software project. Example, the software is required to have a login screen based on company policies.>

## 3.6 Logical Database Requirements

<Will a database be used? If so, what logical requirements exist for data formats, storage capabilities, data retention, data integrity, etc?>

## 3.7 Other Requirements

<Catchall section for any additional requirements that did not belong to the previous sections. If there are none, exclude this section>

# 4. Analysis Models

<List all analysis models used in developing specific requirements previously given in this SRS. Each model should include an introduction and a narrative description. Furthermore, each model should be traceable the SRS’s requirements.>

## Sequence Diagrams

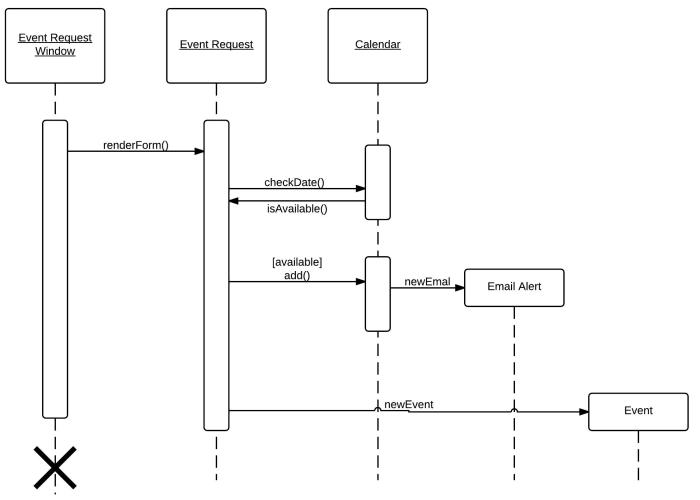


Figure 1: Data Flow Diagram Example 1

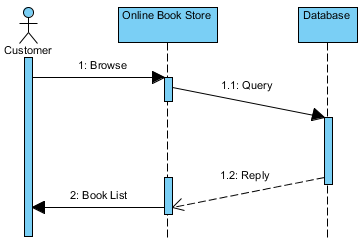


Figure 2 Data Flow Diagram Example 2

<At least one sequence diagram should be included for each requirement or use case.>

# 5. Change Management Process

<Identify and describe the process that will be used to update the SRS, as needed, when project scope or requirements change. Who can submit changes and by what means, and how will these changes be approved.>

# References

# Appendices

<Appendices may be used to provide additional (and hopefully helpful) information. If present, the SRS should explicitly state whether the information contained within an appendix is to be considered as a part of the SRS’s overall set of requirements. Example Appendices could include (initial) conceptual documents for the software project, marketing materials, minutes of meetings with the customer(s), etc.>

## A.1 Appendix 1

## A.2 Appendix 2