SE 3XA3: Software Requirements Specification
Save The Date

Karuka Khurana (khurak1)
Utsharga Rozario (rozariou)
Samarth Kumar (kumars38)
Dhruv Cheemakurti (cheemakd)

February 11, 2022

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	Version	Notes
February 10	1.0	SRS was created
February 10	1.1	Non-functional requirements completed
February 10	1.2	Project drivers completed
February 11	1.3	Project issues completed
February 11	1.4	Functional requirements and project constraints completed (SRS Rev.0 complete)

This document describes the requirements for the **SaveTheDate** project. The template for the Software Requirements Specification (SRS) is a subset of the Volere template (Robertson and Robertson, 2012).

1 Project Drivers

1.1 The Purpose of the Project

Notion is an all-in-one workspace where users can write, plan, collaborate and organize their thoughts, projects, and other information. Notion has a variety of features that allows users to take notes, add tasks, and even create custom page layouts. Alongside its many features, Notion gives users the ability to upload PDF (Portable Document Format) documents onto a page, providing a built-in view of the document, rather than having to download the PDF every time to view it. Currently, many McMaster students skim through multiple course outlines throughout each semester to jot down or remind themselves of important dates. As a result, students are not only spending extra time reading dense pages of course outlines, but also run the risk of missing important dates and deadlines for different courses.

With students taking multiple courses per semester, our team would like to automate the process of sifting through documents for due dates and deadlines. We plan to create easy-to-read dashboards for users to quickly access such information for each respective course they are taking.

1.2 The Stakeholders

1.2.1 The Clients

The clients of this project are the instructor of SFWRENG 3XA3, Dr. Asghar Bokhari and the teaching assistants (TAs) Stephanie Koehl and Abdul Rab Mohammad. The clients will provide information regarding deadlines for weekly deliverables, insight and guidance when needed and assess how well the application meets requirements.

1.2.2 The Customers

The customers for this product are university students. The customers/users have an influence on the requirements of the application and its overall de-

velopment since the purpose of the application is intended to help students organize their dates.

1.2.3 Other Stakeholders

The other stakeholders include the team itself, Group 17, and Notion, as we are trying to integrate our application within Notion. Group 17 is an important stakeholder since the entire development and testing process is dependent on the team.

2 Project Constraints

2.1 Mandated Constraints

2.1.1 Solution Design Constraints

Description: The program shall be written in the React.js and Python programming languages.
Rationale: All developers are familiar with React.js and Python.
Fit Criterion: The software is written in React.js and Python.

Description: The application shall function within Notion.
Rationale: Notion is a popular organization and management tool used by many students. In addition, the existing project implements Notion with all its features.
Fit Criterion: All applications are fully operational within Notion.

2.1.2 Implementation Environment of the Current System

The application will be cloned from the GitLab repository and executed via the IDE of the user's choice such as VS Code. To function, the application will be started on the machine.

2.1.3 Partner or Collaborative Applications

N/A

2.1.4 Off-the-shelf Software

N/A

2.1.5 Anticipated Workplace Environment

The product can be used on any desktop machine that can execute React.

2.1.6 Schedule Constraints

Description: The project must follow the project schedule shown in the SFWRENG 3XA3 course outline.
Rationale: The project needs to follow a predefined plan to ensure the completion of deliverables by their respective due dates and the project by the end of the course.
Fit Criterion: The project will be completed with all the deliverables submitted on time by April 12th, 2022.

2.1.7 Budget Constraints

N/A

2.1.8 Enterprise Constraints

N/A

2.2 Naming Conventions and Technology

Term	Definition
User	The person using the product
SRS	Acronym for Software Requirements Specification; A document that describes what the system will do and the expected performance
React	A front-end JavaScript library for building user interfaces
Tabula-py	Python library for PDF scraping
Pandas	Python library for data manipulation
Cypress	Front-end testing framework
Notion	An organization and project management tool
PDF	Portable Document Format
PC	A personal computer
Scraper	Extracting data from a document or web
OS	An operating system
Clone	Copy the contents from a repository to local storage
GitLab	A type of Repo for version control
Repository	A folder for information
Course Outline	A document that informs all students about the assessments, policies, and grading schemes of a specific course

Table 2: Naming Conventions and Terminology

2.3 Relevant Facts and Assumptions

2.3.1 Facts

- The original repository contains approximately 10,000 lines of code.

2.3.2 Assumptions

- User has already uploaded a PDF using the existing feature to select an upload a PDF to a Notion page.
- Users know how to operate a PC.
- Users understand how to use the features of Notion such as editing a page.
- Users have access to a PC.

3 Functional Requirements

3.1 The Scope of the Work and the Product

3.1.1 The Context of the Work

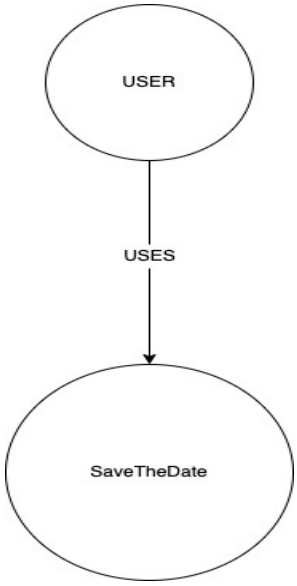

Figure 1: Context Diagram for SaveTheDate

SaveTheDate is a standalone application that does not interact with other systems. It consists of one actor who is the user.

3.1.2 Work Partitioning

Event Number	Event Name	Input	Output
1	Select course name	Mouse/Keyboard	N/A
2	Select page range	Mouse/Keyboard	N/A
3	PDF scraping	Mouse	Deadlines
4	Table editing	Mouse/Keyboard	Changed table

Table 3: Work Partitioning Events

Event Number	Summary
1	The user, through the mouse input and keyboard, can name the course they are choosing to scrape.
2	The user, through the mouse input and keyboard, can write which pages of the PDF they want scraped.
3	The user, through the mouse input, can click a button to scrape the PDF and get the deadlines table in a different Notion page.
4	The user, through the mouse input and keyboard, will be able to edit the deadlines table generated by the PDF scraper.

Table 4: Work Partitioning Event Summaries

3.1.3 Individual Product Use Cases

The use case diagram shown in Figure 2 illustrates how a can interact with our application. To interact with the application, the user has two main cases. The first one is choosing to scrape a document and the second is to view a pre-existing dashboard of deadlines that have already been scraped from a document. To scrape a document, the user has the ability to provide the system with the page range for the document they want to scrape and to specify the course name. The system will then create a table on a new notion page, identify a table within the document and therefore determine

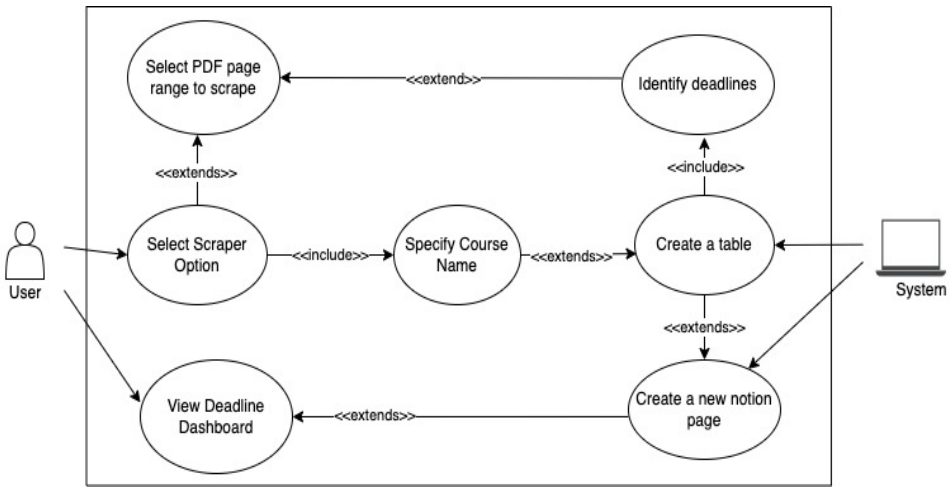

Figure 2: Use case diagram that displays the main functionalities of the application.

the deadlines. This table will be displayed on the dashboard that the user can access.

3.2 Functional Requirements

3.2.1 FR1

Description: The system shall provide the user with a button to begin scraping.

Rationale: The design must be intuitive to use, hence a visual interface for the user.

Priority: High

3.2.2 FR2

Description: The system shall have a way to select the page range they want to scrape.

Rationale: If the user knows the range in which the deadlines might exist in the document, the application can scrape over a specified page range, minimizing processing time.

Priority: High

3.2.3 FR3

Description: The system shall have a way to identify deadline tables in the PDF.

Rationale: It is crucial that the application properly identifies tables with deadlines in a PDF as a PDF might contain a lot of tables irrelevant to deadlines, i.e., tables containing names of professors and their office hours.

Priority: High

3.2.4 FR4

Description: The system shall have a way to structure the output table data.

Rationale: The data obtained from a PDF may not be in a table format when read. Hence, there needs to be a way to structure the data into a table before being output in a Notion page.

Priority: High

3.2.5 FR5

Description: The system shall have a way to create a Notion table in a page.

Rationale: The application must be able to automatically read the data from the PDF and make a page with the read table. Hence, the system must be able to create a Notion table once it has been given the data for the table.

Priority: High

3.2.6 FR6

Description: The system shall display the deadlines as a table in a Notion page.

Rationale: The application should be capable of reading the data and putting it into a Notion table. The table that should be displayed needs to be structured well, in the same if not a better format than that in the PDF.

Priority: High

3.2.7 FR7

Description: The system shall make the table editable.

Rationale: The user might think of changing the deadline in a table, for example, if the due date has been changed by the course instructor. Hence, it is crucial that the table created by the application is editable.

Priority: High

3.2.8 FR8

Description: The system shall ask the user to input the course name/code for the table.

Rationale: Since the name of the course might not be explicitly listed in a PDF, i.e. a PDF with only deadlines or a shared course having several course codes. We required the user to input the course code to include a tag and title with the table.

Priority: High

3.2.9 FR9

Description: The system shall create a new page when the scraper is used for the first time.

Rationale: The idea is to have all the deadlines listed in an easy to access page. Hence a central page for all course deadlines.

Priority: High

3.2.10 FR10

Description: The system shall allow the user to scrape multiple PDFs into one page.

Rationale: To increase functionality of our application, the user must be able to insert multiple deadlines from different PDFs into the same page.

Priority: High

4 Non-functional Requirements

4.1 Look and Feel Requirements

4.1.1 Appearance Requirements

LF1. The application shall be consistent with the Notion theme.

4.1.2 Style Requirements

LF2. The application shall be consistent with the Notion theme.

4.2 Usability and Humanity Requirements

4.2.1 Ease of Use Requirements

UH1. The application must be accessible to the users in an easy-to-access menu.

4.2.2 Learning Requirements

UH2. The users should be able to use the application without prior experience or training.

4.2.3 Accessibility Requirements

UH3. The application must adhere to the same accessibility range as the Notion application.

4.3 Performance Requirements

4.3.1 Speed and Latency Requirements

PE1. The application shall be able to scrape a PDF within a reasonable time.

PE2. The application shall be able to make a Notion page within a reasonable time.

4.3.2 Safety-Critical Requirements

N/A

4.3.3 Precision or Accuracy Requirements

PE3. The application shall be able to identify tables with deadlines with an 85% accuracy.

PE4. The application shall be able to recreate tables with 90% accuracy.

4.3.4 Reliability and Availability Requirements

N/A

4.3.5 Robustness or Fault-Tolerance Requirements

N/A

4.3.6 Capacity Requirements

N/A

4.3.7 Scalability or Extensibility Requirements

N/A

4.3.8 Longevity Requirements

PE5. The application must be functional with existing software and hardware by May 2022.

4.4 Operational and Environmental Requirements

4.4.1 Expected Physical Environment

OE1. The system must not require an Internet connection to function correctly.

4.5 Release Requirements

RR1. The product will have a final release on April 10th, 2022.

4.6 Maintainability and Support Requirements

4.6.1 Maintenance Requirements

MA1. The source code must be fully documented, via commenting and class diagrams.

MA2. The source code must all adhere to the same standard style.

4.6.2 Supportability Requirements

MA3. The project's main repository shall be made public, to allow users to raise issues.

4.6.3 Adaptability Requirements

MA4. The application shall be supported by any machine running Windows 7 or newer, macOS Sierra 10.12 or newer, or Linux Ubuntu 16.04 or newer.

MA5. The application shall be supported by Chrome 98.0.X, FireFox 97.0 and Microsoft Edge 97.0.X.

4.7 Security Requirements

N/A

4.8 Cultural Requirements

4.8.1 Cultural Requirements

CR1. The system shall not allow users to input course names that are culturally offensive/inappropriate.

CR2. The system shall not allow users to input course names that are in languages asides from English.

4.9 Legal Requirements

N/A

4.10 Health and Safety Requirements

N/A

5 Project Issues

5.1 Open Issues

There are no major open issues with the **react-notion-x** repository. A minor issue is the React-Notion collection view calendar is currently unsupported. The last commit was on Jan. 28, 2022.

5.2 Off-the-Shelf Solutions

SaveTheDate is the only open-source PDF deadline scraper that is integrated with Notion webpages. There are other PDF scraping tools like **DocParser** and **Amazon Textract**, however, these are marketed towards businesses that want to scrape PDF data in bulk, and not just deadlines. Furthermore, they are costly to use, and do not produce tables onto a page like Notion.

5.3 New Problems

N/A

5.4 Tasks

Tasks are assigned and scheduled based on the Gantt chart.

5.5 Migration to the New Product

N/A

5.6 Risks

The main risk will be getting familiar with potentially unknown tools and libraries such as **tabula-py** and **pandas** in Python to build the PDF scraper. This will require reading the documentation and referencing existing code which uses these libraries as a starting point. Another risk will be testing, which may be difficult due to a limited amount of test samples for the PDF scraper. Using front-end testing, such as **Cypress**, may additionally be unfamiliar to group members.

5.7 Costs

This project is based on open-source code and uses free software tools such as Python libraries, and React, so there is no monetary cost. There is an estimated time cost of about 50 hours for development, testing, and documentation.

5.8 User Documentation and Training

5.8.1 Documentation

SaveTheDate's repository will contain a README file with a brief description, usage, and installation instructions. There will also be screenshots for user interfacing options. A developer can also see the documented code in the form of an MIS.

5.8.2 Training

By following along an example shown in the README file, the user should understand how to use **SaveTheDate**. Users will be given a reference to

learn the basics of Notion if they are unfamiliar. No specific training is required.

5.9 Waiting Room

Other additions, should there be additional development time:

- Extract deadlines which are not in tables (unstructured)
- YouTube integration, where the user can paste a link and has the option to create a Notion video player for that video.
- Website preview, where hovering over a link shows a small graphics window of that website's home page
- Generating multiple tables in different Notion webpages
- Generating tables of different formats (user can select the output style they want for their deadlines)

5.10 Ideas for Solutions

N/A

References

James Robertson and Suzanne Robertson. *Volere Requirements Specification Template*. Atlantic Systems Guild Limited, 16 edition, 2012.

6 Appendix

This section has been added to the Volere template. This is where you can place additional information.

6.1 Symbolic Parameters

The definition of the requirements will likely call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance.