The Book Assembler

Analysis Document: Software Requirement Specification

Author: Kyrylo Krysko

CONTENTS

1 INTRODUCTION

1.1 APPLICATION PURPOSE

To specify the requirements for the Book Assembler application. Once approved by client it will be the source for Design Document.

1.2 APPLICATION SCOPE

Here we providing high-level description of the Book Assembler application

1.2.1 USER WORKFLOW

In order to understand the software requirements we need to understand the workflow of book creation activity. The process of book developing includes such **phases**:

- Concept,
- Design,
- Layout,
- Printing.

During **concept** phase the Book Author creates the draft version of the book that consist of a page sketches. Once concept is done each page is saved as a jpeg file and sent to Illustrator.

At the **design** phase illustrator creates the final illustrations and return them back to author for review and storage. Author place illustrations into the same folder with pages. Author decides when page is finalized and publishes the finalized version (record the approved version to the database).

Once all Illustrations are finalized, they being send to **layout** artist who creates a Print PDF file for **printing**. This file is sent to a printing facility to produce the circulation of the book.

The book development process is not linear, each page goes through multiple rounds of revisions, and hence each page file has multiple versions.

The pages that Author sending to Layout Artist does not contain version component in the file name, which allows automatic update of all pages to the final in Layout Software.

1.2.2 APPLICATION DESCRIPTION

Client (book author) wants to get a system that will automate: managing of pages and versions, sending approved pages to layout artist, creation of PDF file containing all pages. System should be able to handle any amount of projects (meaning that one book is a one project).

The Book Assembler will ask user to create new or select existing project. Within the selected project Assembler scans the folder with pages and record each page to the database relying on page name (see details in section 6: Data Requirements). The list of all existed versions will be shown in UI as a table where for each page user can also see the published version, sent version and page description.

User can select the page from the list and see the page image in UI. The system will display the published version. If nothing was published, the first (01) version will be shown.

User can observe previous or next versions of selected page pressing “+” or “-“ buttons.

User can record (publish) the current displayed version as approved by pressing “Publish Current Version” button.

User can sent all or selected pages to a layout directory with “Sent Published versions” button. System will copy image files without the version component in the name to a predefined folder. Those images would be used at layout phase.

1.3 DEFINITIONS AND ACRONYMS

Here we will define any technical terms and acronyms used in the document.

Term	Definition
The Book	Paperback printed book that consist of pages with illustrations
Book Author	Person who come up with the book idea, develop overall structure and detailed description of the content. Book Author draws sketches of all pages and sends them to illustrator.
Illustrator	Person who takes the sketch of each page and produce the final illustration.
Layout Artist	Person who takes all final illustrations and assembles a digital version of the book (PDF) for printing in a printing house.
Page	The basic entity of the book, each page correspond to one page file on disk. Each file could have multiple versions.
Sketch	Page characteristic, rough version of a page drawing that shows what should be on the illustration.
Illustration	Page characteristic, refined sketch, the final version of the page that would be printed.
Publishing of a Page	Recording to the database which version of the page is a final illustration
Sending a Page	Each published version can be sent to layout artist. We should keep tracking of sent versions.
Preview PDF	The PDF file of the book for users observation produced by Book Assembler
Print PDF	The PDF file of the book for printing produced by Layout Artist
System Analyst	Software Development manager person, $ 75 per hour rate
Developer	Software Developer, $ 50 per hour rate

2 FUNCTIONAL REQUIREMENTS

This section includes the requirements that specify all the fundamental actions of the Book Assembler system.

Use Case Name: Launch Book Assembler
Use Case Description: User double-click the Book Assembler launcher and application starts: The Book Assembler main window appears showing the current project name and data (the last opened project should be set as current). If there is no previous projects, user should be able to create a new one.

Use Case Name: Create a new project
Use Case Description: User press “Create/Set Project” button. File dialog appears where user can select a root project folder. If root folder is empty (the project is new without any data) the system should create <project root>/pages/”jpg” and “pdf” folders.

Use Case Name: Set different project as current
Use Case Description: User press “Create/Set Project” button, selects folder with existing project, project data should be shown in UI.

Use Case Name: Add page file
Use Case Description: Use adds new page to the project page folder and press “Reload Pages” button. List of new pages should be shown in UI.

Use Case Name: Preview page
Use Case Description: User selects the page in the list of pages, Book Assembler shows the published page version in UI.

Use Case Name: Preview next page version
Use Case Description: If page is selected, and hence shown in UI, user press “+” button, Book Assembler shows next version (currently shown + 1) of the page. If page is not selected, the warning message shown in the status bar: “Please, select page to explore versions”. If next version does not exists, the warning message shown in status line “Page <page number> version <version> does not exists!” and no image shown in preview.

Use Case Name: Preview previous page version
Use Case Description: If page is selected, and hence shown in UI, user press “-” button, Book Assembler shows previous version (currently shown - 1) of the page. If page is not selected, the warning message shown in the status bar: “Please, select page to explore

versions". If previous version does not exists, the warning message shown in status line "Page <page number> version <version> does not exists!" and no image shown in preview.

Use Case Name: Publish current page version
Use Case Description: If page is selected, user press "Publish Current Version" button. Current version of page is registered in database and shown in "Published" sell in the page list. If page is not selected, the warning message shown in UI: "Select page to publish!"

Use Case Name: Generate preview PDF
Use Case Description: User enter the PDF file version in "pdf version text field", press "Generate PDF file" button. Book Assembler creates PDF file from all existing pages using published versions. The preview PDF file is created in <book project folder>/pages/pdf

Use Case Name: Copy files to layout folder
Use Case Description: User press "Sent Published Versions" button, Book Assembler copies the published versions of each page to the layout folder <book project folder>/pages/layout. During the copy process each page file is renamed to remove version component from the name (0000_03.jpg becomes 0000.jpg). If "SEL" check box is ON, only selected pages are copied to layout folder.

Use Case Name: Enter/change page description
Use Case Description: User double click the Description sell and enter text. If description already been entered before, the text is shown after double click. Description stored in the data base.

Use Case Name: Change project folder
Use Case Description: User selects in main menu: Edit > Set Project to change project location. This information is stored in the setting file.

Use Case Name: Launching documentation
Use Case Description: User selects in main menu: Help > Documentation to launch default web browser with html user documentation for Book Assembler.

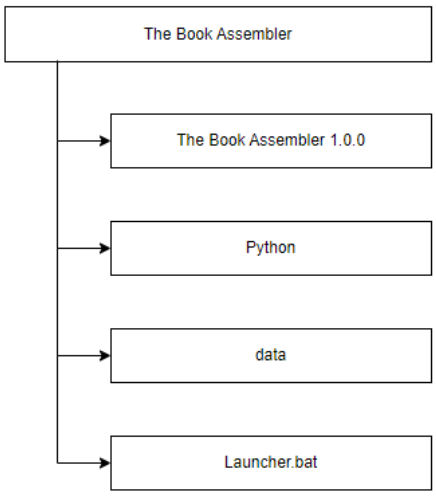
3 NON-FUNCTIONAL REQUIREMENTS

3.1 INSTALLATION AND UPDATES

The Book Assembler is meant to be used by one user only on one computer, so we can skip creation of installation package that includes all dependencies. We can deliver the zip file with application that includes:

- Application Folder, name should include the app version
- Python folder with necessary modules
- Data folder, contains SQLite database and settings JSON files.
- Application launcher, pointing to the latest version.

Unzipping application ZIP to any folder on PC or in the cloud should be enough to run application with a launcher. To update application with a new version the folder with a new version should be placed to App Dir and launcher needs to be updated to point to a new version.



3.2 USER INTERFACE

There is no necessity to provide modern high-end user friendly UI. The UI should contain:

3.3 DATA REQUIREMENTS

The input data should be valid. For any projects page files should be placed in the <project root>/pager/jpeg folder and has proper name: <**page name>_<2 digits page version>.jpg**

The Book Assembler should scan the folder with input data (page images) and read files that meets naming conventions, other files can be ignored.

Existing pages will be recorded to the database. Database should be able to store **published** and **sent** versions of each page

3.4 ENVIRONMENT

The Book Assembler should support the One Drive access and store all data in the cloud. The application itself could be either installed on the user PC (Windows 10 OS) or on the cloud as well.

3.5 DOCUMENTATION

The Book Assembler should include user documentation.

4 TEST PALN

4.1 INTRODUCTION AND OVERVIEW

An overview of the test plan, its purpose, and a brief description of the software under test.

Objectives:

- Verify that the application meets all functional requirements.
- Validate data input, processing, and output.
- Ensure the application is user-friendly and error-free.

Test scope: developer will test all use cases listed in functional requirements. Using test project data provided by client we will ensure that all functions are working correctly. Next developer will update project data to introduce possible user errors like naming files incorrectly to ensure that application is still functions.

Test level:

- Unit testing: check individual functions and components.
- Acceptance testing: Ensuring the application meets user needs and requirements.

Test Environment:

- Python version: 3.7
- Operating systems: Windows
- Database: SQLite.

Test resources: Testing will be done by developer. The test project data will be provided by client, which includes project folder with a list of pages.

4.2 TEST DELIVERABLES

Test deliverables are the artifacts or documents generated during the testing process. These deliverables help in tracking the progress of testing, provide evidence of testing activities, and serve as reference material for future projects. For the Book Assembler Application, the following test deliverables will be produced:

4.2.1TEST PLAN DOCUMENT

A comprehensive document outlining the testing approach, objectives, scope, resources, schedule, and other relevant details.

4.2.2 TEST CASES

Detailed descriptions of each test scenario, including the input data, expected results, and the purpose of the test. These cases will cover various functionalities and components of the application, such as creating new projects, publishing file versions etc.

Test Case Name: Launch Book Assembler.
Test Case Description: User double click launcher file in the application install directory.
Test Case Input: Application install directory.
Test Case Expected Results: Main window appears.

Test Case Name: Create a new project.
Test Case Description: With the Book Assembler launched, user click “Create/Set project”. File Open Dialog appears, user select a root folder for a new project: “C:/Users/kko8/OneDrive/projects/master/CIS575/projects/workbook”
Test Case Input: The Book Assembler.
Test Case Expected Results: The Book Assembler shows current project name and path in UI.

Test Case Name: Add pages to a new project.
Test Case Description: User copy pages from test project (provided by client) to a new project. With the Book Assembler launched, user click “Reload Pages” button.
Test Case Input: New pages copied to <root project>/pages/jpg folder.
Test Case Expected Results: The Book Assembler shows list of pages in UI. The new pages should be added to database tables.

Test Case Name: Set different project as current
Test Case Description: User click “Create/Set project” button and select another project in File Dialog.
Test Case Input: Another project with data
Test Case Expected Results: The Book Assembler shows list of pages for a new project in UI

Test Case Name: Add new pages to an existing project.
Test Case Description: User copy new pages from test project (provided by client) to a current project. With the Book Assembler launched, user click “Reload Pages” button.
Test Case Input: New pages copied to <root project>/pages/jpg folder.
Test Case Expected Results: The Book Assembler updates list of pages with new pages in UI

Test Case Name: Preview page
Test Case Description: User selects the page in the list of pages
Test Case Input: Project with pages.
Test Case Expected Results: The Book Assembler shows page version 01 image in UI

Test Case Name: Add page description
Test Case Description: User selects the page in the list of pages, double click description field and enter text.
Test Case Input: Project with pages.
Test Case Expected Results: The Book Assembler shows page description text in UI, corresponding database record created.

Test Case Name: Preview previous/next page version
Test Case Description: User selects the page in the list of pages, and press +/- buttons
Test Case Input: Project with pages.
Test Case Expected Results: The Book Assembler shows the +1/-1 page version images in UI.

Test Case Name: Publish current page version
Test Case Description: User selects the page in the list of pages, press +/- buttons to set desired version, press “Publish Current Page Version”.
Test Case Input: Project with pages.
Test Case Expected Results: The Book Assembler shows the published version in UI for selected page. Database record should be created.

Test Case Name: Sent pages to Layout
Test Case Description: User clicks ”Sent Published Versions” button.
Test Case Input: Project with pages.
Test Case Expected Results: The Book Assembler copy all pages of published versions to “to_layout” folder. Page version should be cut from the file name. If page is unpublished, use 01 version.

Test Case Name: Generate PDF file
Test Case Description: User clicks ”Generate PDF” button.
Test Case Input: Project with pages.
Test Case Expected Results: The Book Assembler creates a PDF file in “pdf” folder. PDF should contain images of published versions. If page is unpublished, use 01 version.

4.2.3 TEST DATA SET

Collections of data used as input for the test cases. Test data sets should represent various scenarios, including typical use cases, boundary cases, and edge cases to ensure thorough testing of the application. Client will provide a test project with all necessary page files. It should include at least one set of updated data, e.g. set of new pages and existing pages with a new versions.

4.2.4 TEST RESULTS AND REPORTS

Documents that summarize the outcome of the test cases, including pass/fail status, defects discovered, and other relevant information.

4.2.5 DEFECT REPORTS AND RESOLUTION

Detailed descriptions of the defects or issues identified during testing, including steps to reproduce the problem, severity, priority, and any additional relevant information.