SHARDA UNIVERSITY

GREATER NOIDA

School of Engineering & Technology
Department of Computer Science & Engineering

Software Engineering and Testing Methodologies Lab

(CSP357)

SUBMITTED BY:

SURYA RAJ (2022484252)

CLASS: - CSE – D

Experiment Number	Date	Name of Experiment	Signature

Software Requirements Specification for
Snap Script AI: Transforming Images to Text with Precision

Prepared by

SURYA RAJ (2022484252)

Sharda University

Table of Contents

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

- # Introduction

1.1 Purpose

This document provides a detailed description of the SnapScript website. The purpose of SnapScript is to convert image text to normal text using Tesseract OCR, generate PDF notes using the Gemini API, and provide a chatbox for user interaction. The website will be developed using HTML, JavaScript, CSS, and Bootstrap, and will be hosted on GitHub.

1.2 Scope

The SnapScript website will allow users to upload images, extract text from these images, and convert the text into a downloadable PDF file. Additionally, users can interact with a chatbox for assistance and queries.

1.3 Definitions, Acronyms, and Abbreviations

OCR: Optical Character Recognition

- PDF: Portable Document Format

- API: Application Programming Interface

- HTML: Hypertext Markup Language

- CSS: Cascading Style Sheets

- JS: JavaScript

1.4 References

Tesseract OCR: [https://github.com/tesseract ocr/tesseract]

- **Overall Description**

2.1 Product Perspective

SnapScript is an independent web application designed to provide OCR and PDF generation functionalities. It will be hosted on GitHub and will leverage the Gemini API for note generation.

2.2 Product Functions

Upload images and extract text using Tesseract OCR.

Convert extracted text to a formatted PDF using the Gemini API.

Download the generated PDF.

Interact with a chatbox for assistance and queries.

2.3 User Classes and Characteristics

General Users: Individuals who need to convert images to text and generate PDFs. Administrators: Users who manage the website content and handle user queries through the chatbox.

2.4 Operating Environment

The SnapScript website will be accessible on modern web browsers (Chrome, Firefox, Safari, Edge) and hosted on GitHub Pages.

2.5 Design and Implementation Constraints

The website must be responsive and user-friendly.

The application must ensure data privacy and security.

The chatbox should provide real-time interaction

2.6 User Documentation

Online help documentation and tutorials.

FAQ section on the website.

2.7 Assumptions and Dependencies

Users will have a stable internet connection.

The Gemini API will be available and functioning

- **Specific Requirements**

3.1 Functional Requirements

The UI will be eye catching with a basic color scheme which will provide a relaxing and a motivational theme while searching for the prompts.

3.1.1 Image Upload

Users can upload image files (JPEG, PNG).

The system will display a preview of the uploaded image.

3.1.2 Text Extraction

The system will use Tesseract OCR to extract text from the uploaded image.

Extracted text will be displayed on the webpage for review.

3.1.3 PDF Generation

Users can click a button to generate a PDF from the extracted text.

The system will use the Gemini API to format the text into a PDF.

The generated PDF will be available for download.

3.1.4 Chatbox

A chatbox will be available for user queries and assistance.

Real-time messaging will be supported.

3.2 Non-Functional Requirements

3.2.1 Performance Requirements

The text extraction and PDF generation should be completed within a reasonable time frame (e.g., under 30 seconds for average-sized images).

3.2.2 Usability Requirements

The website should have a clean, intuitive interface.

Instructions should be clear and concise.

3.2.3 Reliability Requirements

The system should handle up to 100 concurrent users.

The website should have an uptime of 99.9%.

3.2.4 Security Requirements

User data should be encrypted during transmission.

Access to the administrative functions should be secured with authentication.

a) Website for Content Writing: The system will interact with the user and will get

b) the prompt for the Content that has to be printed.

c) Database: The application will interact with a database to store user account.

d) API: An API is developed to redirect the prompt for the data.

- **System Features**

4.1 Image to Text Conversion

Description: Allows users to upload an image and converts the image text to normal text.

Stimulus/Response: User uploads image → System extracts text and displays it.

Functional Requirements:

FR1.1: The system shall accept image files in JPEG and PNG formats.

FR1.2: The system shall display the extracted text on the webpage.

4.2 PDF Generation

Description: Converts extracted text into a PDF document.

Stimulus/Response: User clicks "Generate PDF" → System generates PDF and provides a download link.

Functional Requirements:

FR2.1: The system shall use the Gemini API to format the text into a PDF.

FR2.2: The system shall provide a link to download the generated PDF.

4.3 Chatbox

Description: Provides a chat interface for users to interact with the system.

Stimulus/Response: User types a message → System responds in real-time.

Functional Requirements:

FR3.1: The system shall support real-time messaging.

FR3.2: The system shall allow administrators to manage chat interactions.

• External Interface Requirements

5.1 User Interfaces

Home Page: Image upload button, preview section, extracted text display, PDF generation button, and chatbox.

Chatbox Interface: Real-time chat functionality for user assistance.

5.2 Hardware Interfaces

No specific hardware interfaces are required.

5.3 Software Interfaces

Tesseract OCR: For text extraction.

Gemini API: For PDF generation.

5.4 Communications Interfaces

The website will use HTTPS for secure communication.

• Other Non-Functional Requirements

6.1 Performance Requirements

The system should be able to handle text extraction and PDF generation within 30

seconds for an average image size.

6.2 Security Requirements

All user data should be encrypted during transmission.

The chatbox should have proper user authentication.

6.3 Software Quality Attributes

Usability: The interface should be user-friendly and intuitive.

Reliability: The system should be reliable and have minimal downtime.

Maintainability: The code should be well-documented for easy maintenance.

Appendix A: Glossary

1. API (Application Programming Interface): A set of routines, protocols, and tools for building software and applications.
2. CSS (Cascading Style Sheets): A style sheet language used for describing the presentation of a document written in HTML or XML.
3. GitHub: A web-based platform used for version control and collaborative software development.
4. HTML (Hypertext Markup Language): The standard markup language used for creating web pages.
5. OCR (Optical Character Recognition): Technology used to convert different types of documents, such as scanned paper documents, PDF files, or images captured by a digital camera, into editable and searchable data.
6. PDF (Portable Document Format): A file format used to present documents independently of application software, hardware, and operating systems.
7. Tesseract OCR: An open-source OCR engine for various operating systems.
8. Gemini API: An API service used for generating notes from text.
9. JS (JavaScript): A programming language commonly used to create interactive effects within web browsers.
10. Bootstrap: A free and open-source CSS framework directed at responsive, mobile-first front-end web development.

Appendix B: To Be Determined (TBD) List

1. TBD1: Detailed requirements for the chatbox's real-time interaction capabilities.
2. TBD2: Specific security measures for encrypting user data during transmission.
3. TBD3: Authentication mechanisms for accessing administrative functions.
4. TBD4: Detailed integration process with the Gemini API for PDF generation.
5. TBD5: Exact performance benchmarks and testing protocols for OCR and PDF generation times.