Software Requirements Specification

for

F.U.M.L. (F.U.M.L. Uses Machine Learning)

**Version 1.0 approved**

**Prepared by Trevor Brooks, Michael Fulghum, Michael Goetz, Felix Potter & David de Sousa**

**F.U.M.L.**

**November 17, 2021**

**Table of Contents**

[**Introduction**](#_heading=h.3znysh7) **1**

[**Purpose**](#_heading=h.2et92p0) **1**

[**Document Conventions**](#_heading=h.tyjcwt) **1**

[**Intended Audience and Reading Suggestions**](#_heading=h.3dy6vkm) **1**

[**Product Scope**](#_heading=h.1t3h5sf) **1**

[**References**](#_heading=h.4d34og8) **1**

[**Overall Description**](#_heading=h.2s8eyo1) **2**

[**Product Perspective**](#_heading=h.17dp8vu) **2**

[**Product Functions**](#_heading=h.3rdcrjn) **2**

[**User Classes and Characteristics**](#_heading=h.26in1rg) **2**

[**Operating Environment**](#_heading=h.lnxbz9) **2**

[**Design and Implementation Constraints**](#_heading=h.35nkun2) **3**

[**User Documentation**](#_heading=h.1ksv4uv) **3**

[**Assumptions and Dependencies**](#_heading=h.44sinio) **3**

[**External Interface Requirements**](#_heading=h.2jxsxqh) **3**

[**User Interfaces**](#_heading=h.z337ya) **3**

[**Hardware Interfaces**](#_heading=h.3j2qqm3) **4**

[**Software Interfaces**](#_heading=h.1y810tw) **4**

[**Communications Interfaces**](#_heading=h.4i7ojhp) **4**

[**System Features**](#_heading=h.2xcytpi) **4**

[**Fillable PDF Creation**](#_heading=h.1ci93xb) **5**

[**Website HTML Functionality**](#_heading=h.3whwml4) **5**

[**Website Account Creation and Access**](#_heading=h.kojyy1epbsfw) **6**

[**Fillable PDF Email Messaging**](#_heading=h.cgyi85aravg3) **6**

[**Other Nonfunctional Requirements**](#_heading=h.2bn6wsx) **7**

[**Performance Requirements**](#_heading=h.qsh70q) **7**

[**Safety Requirements**](#_heading=h.3as4poj) **7**

[**Security Requirements**](#_heading=h.1pxezwc) **7**

[**Software Quality Attributes**](#_heading=h.49x2ik5) **8**

[**Business Rules**](#_heading=h.2p2csry) **8**

[**Other Requirements**](#_heading=h.147n2zr) **8**

[**Appendix A: Glossary**](#_heading=h.3o7alnk) **8**

[**Appendix B: Analysis Models**](#_heading=h.23ckvvd) **8**

**Revision History**

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

The software product under the name of “F.U.M.L.” is a software package. This document details the entire software package including its front-end and back-end functions and components exhaustively. (Fulghum)

## Document Conventions

Normal text that serves as content and does not need any special emphasis will not have any special inflections such as bolding, italicizing, and underlining. All headers are prefixed with a numeric identifier; a numeric identifier includes: a positive integer, a numeric identifier with a period and then a positive integer. Each period in a numeric identifier represents the start of a subsection.

Filenames are written in two double quotes, use the forward slash as a separator, and are referenced from the root of the official software’s source code directory. Terminal commands are to be placed between two double quotes. (Fulghum)

## Intended Audience and Reading Suggestions

The intended audience for this SRS are developers that are interested in software recognition, and documentation writers. The rest of this SRS contains information on how our program runs and the type of functions the user can use when interfacing with the program. Start with the overall description, and continue on with what pertains to your needs. (Brooks)

## Product Scope

Many people still rely on inefficient paper forms to collect information and existing PDF form conversion tools are costly and limited. The F.U.M.L. software is an easily accessible and free web app that allows users to upload PDFs and have them converted to fillable forms through the recognition of fields and boxes which they can then download. The website hosts ads to pay for its upkeep and generate profit. F.U.M.L. seeks to solve the cost issues of current form recognition software as well as the inconsistencies of missing or incorrect fields. (Goetz)

## References

There are no references at the moment. (Fulghum)

# Overall Description

## Product Perspective

Our product originated from the idea of providing a cheaper alternative than other current services. This product seeks to replace systems that are already in place, specifically other document editing softwares that can interact with PDFs. Ours would not wholly replace these other programs, but rather provide a compelling option. (Brooks)

## Product Functions

This product serves as a tool for PDFs to make them able to be fillable in places where the product deems appropriate. The function for this product are as follows:

* Allow for a file as input of the specific file format PDF
* PDFs of varying sizes and page formats should work just as well as more commonly used formats
* Given a PDF, a fillable version should be provided as output in return.

(de Sousa)

* Users should be able to navigate the website and its pages without incident or irregularities
* Users should be able to create and access their accounts on the website
* If logged in, users should have the option to receive emails of their fillable PDFs that the software returns

(Goetz)

## User Classes and Characteristics

The two most important user classes are small businesses and individuals. A small business refers to a business that does not yet have a practical use for complex software infrastructure to perform its software needs. An individual refers to a person that uses the software for personal reasons only that may or may not be used in commercialization. For small businesses and individuals, it is expected that most if not all of the features of the software will be used. This software product is not intended to be used by large businesses that already have a need for a complex software infrastructure to satisfy their computational and actuarial needs. (Fulghum)

## Operating Environment

The software is composed of two parts: the front-end website and the back-end server. The front-end website is built on Flask and an SQL database. Due to the user accessing the app through the web, their operating system will not have any bearing on their experience. The different versions of the software are not a concern for end users as well, as the website is a singular entity that can be updated and accessed and due to the separation of the front- and back-end. (Goetz)

## Design and Implementation Constraints

Lawful policies regarding the handling and usage of personal information such as email address and password have to be obeyed before the software is deployed. The operating system must be able to fetch an implementation of all of the required python packages from either the PyPI repository or another service as specified in the list of python packages in the file named “/requirements.txt.” The environment should have python 3.9 or higher installed if not the latest. Every invocation to a python module in the official source code must be performed at the root of the source code (the current working directory must also, therefore, be at the root) and executed via a command of the form “python3 -m <module\_name>” for defined behavior. (Fulghum)

## User Documentation

There is no need for user documentation as the software is a web application. The design is intuitive enough that any tutorials to properly operate the software are unnecessary; the aim of the website is to be immediately accessible by a new user. (Goetz)

## Assumptions and Dependencies

All third-party, non-standard python packages and modules are assumed to be well tested and to work in the future; this includes availability of the packages, security and stability, and cross-platform support. The server at which the software is intended to be deployed is assumed to have the ability to host multiple Flask sessions at a time to support the needs of multiple users at a time. A sufficient amount of RAM and secondary storage is assumed to be present at the server that will allow it to function in the presence of a few multiple users at least.

The software package depends on python 3.x at least, pip for the said python version, and the following PyPI packages by name: numpy, matplotlib, opencv-python, PyMuPDF, boxdetect, reportlab, and PyPDF2. (Fulghum)

# External Interface Requirements

## User Interfaces

3.1.1 Error messages

Error messages will occur when information is not put in correctly in text boxes.

3.1.2 Navigational Components

Menu, Icons, breadcrumbs, and text pagination will help the user navigate

the website.

3.1.3 Color and Typography

Helps increase readability, scannability, and readability

(Potter)

## Hardware Interfaces

Given our product is a website and solely deals with document conversion there is no hardware component to our product other than the device used to access our website. (de Sousa)

The website has no hardware interface requirements. (Potter)

## Software Interfaces

The software package, mainly made of python modules and packages, is intended to be executed from the user or another process via the terminal or a functionally equivalent mechanism; this is the primary way the Flask sessions ought to be started by another process. SQLite databases are maintained by the software package locally and are intended to store actuarial information about the service’s users securely. This software package is not designed to be highly extensible but is geared towards server deployment. (Fulghum)

## Communications Interfaces

Our product has the ability to send emails, and connects to the internet utilizing the Flask package to connect to a port. Since emails can be used, the obvious case of email security arises, and users must be cautious to not open emails from unknown sources. no-reply@fuml.com is the email that the users will receive emails from. At some point, we could also transfer to a more secure server and implement HTTPS. (Brooks)

# System Features

* The website will allow for a file as input of the specific file format PDF and rejects any file that is not of that type.
* PDFs of varying sizes and page formats should work just as well as more commonly used formats *ei. letter,* and as such return a PDF which has correctly positioned and functioning fillable components.
* Given an input of the PDF filetype the website should return a file of the same format, PDF, but now adjusted to contain fillable areas (where deemed appropriate by a box detection algorithm) and should be presented as follows:
  + All checkboxes are able to be checked and unchecked and in a reasonable position relative to where it should be.
  + All textboxes are able to be written in with the appropriate font size and in a reasonable position relative to where it should be

(de Sousa)

* The website should provide an intuitive and accessible interface for users, requiring only common technological knowledge to operate.
* Users should be able to create and access secure accounts that are checked for validation and verified through email.
* Users should be able to receive emails with their fillable PDFs if they are logged in and have selected the option.

(Goetz)

## Fillable PDF Creation

4.1.1 Description and Priority

The main function of the website will take in a PDF, scan it for check-boxes and signature areas, and return a version of that PDF that will highlight those fillable zones for the user. This is a high priority function, since without this service there wouldn’t be much of a point of the website. (Brooks)

4.1.2 Stimulus/Response Sequences

The user will have a clickable option that will take them to a page that will then allow them to upload their PDF for conversion. (Brooks)

4.1.3 Functional Requirements

Fillable PDF creation on the server must be able to handle any page size and aspect ratio. Any attempt at such creation must be able to report errors in its business logic, as well as errors propagated from the environment, to the function callers if needed; any error should have strong exception safety: if there is a failure or an error, no side effects are caused. PDF creation should have little to no side effects on the server if possible; effectively, there should be no coupling outside of the core logic via external state (Fulghum).

## Website HTML Functionality

(Goetz)

4.1.1 Description and Priority

This feature entails the basic functionality of the website for users. This is a high priority, as a website with poor functionality would dissuade potential users.

4.1.2 Stimulus/Response Sequences

Users should be able to click and access the different .html pages of the site. Each and every button and link should lead to a page on the site. There should not be any “dead” elements that serve no purpose.

4.1.3 Functional Requirements

REQ-1: Buttons should perform their intended function and redirect to the correct page.

REQ-2: Text input boxes should be accessible and represent clearly to the user what they are exactly entering into the software.

REQ-3: Certain pages should not be accessible to users that are not logged in; they should be redirected to a 404 error page with a way to return to the homepage.

## Website Account Creation and Access

(Goetz)

4.1.1 Description and Priority

This feature entails the ability of users to create and access accounts on the website. This is a medium priority in the sense of account creation itself; however, account security is a higher priority for protecting sensitive user information. The security in question comes from the encryption and hashing of user data and the password requirements during the sign up process.

4.1.2 Stimulus/Response Sequences

When logged out, users should have an option to create an account or log in. If they choose to sign up, their input for their username and password should be checked before successful account creation. If they want to sign in, they should be able to use either their email or username with their password. If a user logs out, they should be able to log back in or log into a different account in the same session.

4.1.3 Functional Requirements

REQ-1: Logged out users should be able to create an account that checks for empty usernames and strong passwords. They should not be able to circumvent any of the account creation safeguards and are to receive information upon unsuccessful account creation of what requirements they need to meet.

REQ-2: Users should be able to sign in with either their username or email and password. The SQL server should accurately store the correct ciphertext versions of user data to ensure integrity.

REQ-3: Logging out should update the Flask session accordingly and treat the user as a logged out user afterwards.

REQ-4: After a period of idle time, the user should automatically be logged out and informed why.

## Fillable PDF Email Messaging

(Goetz)

4.1.1 Description and Priority

This feature entails the sending of an email from the website to the user that contains their fillable PDF result. Among each of the functional requirements, this is of the lowest priority, as users will always have the option to download their most recently uploaded PDF through the website.

4.1.2 Stimulus/Response Sequences

When logged in, by default a new user account will have the option to receive PDFs in their emails ticked off. If they access the option in settings and flip it on, the next time they upload a PDF the website should send an email with the attached fillable PDF afterwards.

4.1.3 Functional Requirements

REQ-1: The user should be able to toggle on and off whether they wish to receive the email of their fillable PDF.

REQ-2: The email should contain a concise, friendly message with the proper attachment of the same PDF the user downloaded from the website.

# Other Nonfunctional Requirements

## Performance Requirements

The main performance requirement will be the processing of the file and converting it to the desired PDF fillable form. Since different files will have different sizes, this process could be potentially shorter or longer. Another smaller performance requirement will be the email system. This is also tied into the PDF size, but could also take extra time to send to the user’s email. (Brooks)

## Safety Requirements

As our product could be used at both a personal and commercial level, it is imperative that the users, whether they be individuals or those representing a company, keep usernames and passwords secret, and to not open any false emails that could be potential malware or phishing links. (Brooks)

## Security Requirements

5.3.1 Authentication

Authentication ensures that each user that is is who they claim to be when

using the website.

5.3.2 Data Protection

Data Protection ensures that the service requested has not been tampered

with. Encryption and password hashes help secure data integrity and privacy.

Prevent multiple forms with the same name

Ensures that forms with the same name will have an additional modifier to prevent the same name and result in one being deleted.

Timed logout

We make sure the user will be logged out after a certain time so no unauthorized personnel can access their account with them unnoticed. (Potter)

## Software Quality Attributes

The main focus of this product is to provide a fast and easy way to convert regular PDFs into their fillable counterparts. With that comes ease of use. We have a very clear and simple interface for file selection; speed and conversion time is quite timely given the scale. Our product has been rigorously tested for reliability. Our initial scope was achieved and even expanded, with our stretch goals being met. (de Sousa)

## Business Rules

Any user that wishes to use the hosted software product can do so by visiting the website. Any qualifying user may create an account to allow the service to send emails and perform other administrative tasks. (Fulghum)

# Other Requirements

The software should be able to analyze any reasonable document page size to allow for international standards such as the A4 document size. (Fulghum)

# Appendix A: Glossary

(Potter)

**PDF:** portable document format that allows user to share and print out document in a non editable format

**requirement:** A statement of a customer need or objective, or of a condition or capability that a product must possess to satisfy such a need or objective.

**user:** A customer who will interact with a system either directly or indirectly (for example, using outputs from the system but not generating those outputs personally). Also called *end user*.

# Appendix B: Analysis Models

(Fulghum)

Diagram between user and server over https.

