SOFTENG 750

Project Proposal

Team - Fearless Foxes
Ben Lowthian
Jia Tee
Jordan York
Matan Yosef
Matthew Jakeman
Ojas Madaan

Introduction

The application proposed is a web-based PDF editor which allows for collaborative annotation of documents in real-time. The editor will focus on supporting text and mathematic based annotations.

The motivation for this project comes from a lack of a free, web-based PDF editing software in the current market. Although there are many powerful programs, they either do not contain any collaborative features, are desktop based or remain behind an expensive subscription model.

Additionally, this project emerged from the idea to allow for annotation of course and lecture notes with instructors and peers. Being students ourselves, we want to have the best experience possible when taking notes in class to achieve well. By all personally related to the project's need, we were keen to tackle the task.

Related Work

Across the market, there are several applications which allow for PDF annotation. However, many lack several features and seem to 'miss' important functional requirements of an all-round free PDF annotation software. A list of the some commonly used PDF annotation software (according to this article), alongside their shortcomings, are as follows:

- **Drawboard PDF:** Originally designed as a desktop-based app - the web-based solution is quite heavy and slow, and does not natively support maths annotations.
- **Acrobat:** Originally a desktop application which supports collaboration, powerful annotations. However, it is known to be quite expensive.
- **OneNote:** Features collaboration and powerful annotation including mathematical annotation, however one of the key shortcomings is that PDFs are imported as images, resulting in a loss of structure in the document that cannot be re-exported.
- **Lumin PDF:** A lot of similarities - collaboration, pdf editing, but does not natively support maths annotations.

There is no lightweight, "no strings attached" PDF annotation tool. All of the existing solutions come with caveats and are not in-line with modern web-first trends. We believe there is a clear gap in the market both in terms of technology and business for the collaborative PDF annotation tool outlined above.

Requirements

The core project requirements and features can be separated into must-haves and should-haves. This is what we aim to implement within the project timeline:

Must Haves (Minimum Viable Product)	**Should Haves (Target Product)**
Single page PDF annotation ● Load PDF ● Annotate PDF ● Annotation is persisted Annotation types ● Text box annotation Collaboration ● Editing sessions ● Multiple users, one document ● Annotations appear in real time File management ● User login system ● File upload ● 'Your files'	Multi-page PDF annotation ● Support for multi page PDF documents Workspaces ● Grouped collections of documents ● Per-workspace permissions Annotations ● Maths annotations (using LaTeX or MathML) ● Rich text support in text annotations ● Wet/dry/super-wet ink (stroke simplification)

Additionally, we have defined could-haves and nice-to-haves in terms of additional goals if the project is running ahead of schedule:

Could Haves (Stretch Goals)	**Nice to Haves**
"Real" PDF Editing ● Use PDF annotations instead of overlaid graphics ● Exporting as a flattened PDF	Notion integration ● Linking and referencing ● Import/export Utilise AI through ● Stroke recognition ● Text analysis ● Generate flashcards from the pdf

Technology

What technologies and other material will be used to complete your project? This should include all relevant content from the lectures, but also material that is *not* taught in class. It is important that you demonstrate the ability to carry out independent learning, so please do research and apply libraries / frameworks / technologies other than just what is taught, and / or delve deeper into the tech that is introduced.

Frontend Technologies

React

React is the technology of choice for the frontend. It is an industry standard library for building web UIs. It is the frontend framework the team is most familiar with and provides sufficient functionality .

pdf.js

The **pdf.js** library, developed by Mozilla for the Firefox browser, was chosen for PDF rendering. It is a mature, well tested, actively developed library which can performantly render PDF files to a canvas element and is well suited for our use-case. It is released under the Apache 2.0 licence which additionally reduces barriers to adoption.

In terms of integration with React, we considered the **react-pdf-js** library but are tending towards using pdfjs directly for lower level control of the rendering pipeline.

Inking

For inking support, the **'Touch events'** web API will enable us to receive pressure sensitive events from styluses such as the Apple Pencil and Surface Pen. This will be integrated with the canvas element to draw above the PDF document. As the MVP does not require true PDF annotation support, we will store and render ink annotations as overlays.

Tailwind CSS

Tailwind CSS allows for a class-based styling system as opposed to typical CSS styling. We have elected to use Tailwind since members have experience with the technology, and with such a short timespan, it makes design quick, easy, and highly readable, and tailwind lets us do that.

Backend Technologies

Express & NodeJS

We looked to use a similar technology stack for frontend and backend, and as such opted for a JavaScript-based backend. We chose Express and NodeJS, as it will make it easy for us to get a web service up and running for our backend. Additionally, both projects have mature TypeScript support.

TypeScript

TypeScript provides type safety over JavaScript, greatly improving code quality and development efficiency. The team generally has experience working with typescript through previous projects and internships, so it was a good fit for the backend.

Socket.io

For bidirectional and low-latency communication, we will use socket.io to communication and persist annotations from frontend to backend..

MongoDB

We have opted for MongoDB as our database store for this project. It is a NoSQL database which utilises JSON-like documents to store information. Using a non-relational database provides us with greater flexibility while developing the application, and has less rigidity over the data that can be stored.

Project management

The project management strategy we will use is an incremental agile strategy. With this in mind, we aim to have 1 week sprints to complete the project, and are looking to have 8 total sprints in order to complete development. Alongside this management strategy, we will use Jira for issue and work tracking. This decision comes from the team's experience with the tool, and possible integration potential. Slack will be used as the main form of asynchronous communication, with GitHub, Jira, or Notion integrations created as deemed appropriate.

In terms of version control, the strategy will consist of one 'main' branch alongside feature branches which are based on the issue, as created in Jira. For example, if a developer is working on a ticket 'FRONT-12' to implement a login component, I will create a branch called FRONT-12-login-component. Similarly, to distinguish what commits are related to what issue, the team will use the issue number as the start of the commit message. For example, "FRONT-12: added form inputs" is a valid commit message.

Upon merging, pull requests into main will need to be reviewed by at least one other developer from the appropriate team (i.e. backend or front-end).

Risks in team projects

There are numerous risks that may occur in a student team project. We have outlined some common risks and mitigations strategies as follows:

Risk	Example	Mitigation Method	Monitoring Plan
A team member suddenly becomes unavailable	A team member has a family emergency to attend to	Matan to document work done, meeting notes, etc. So that other team members can continue their work	Rotate through team members to take minutes in meetings to ensure that progress updates are recorded. Also, all team members will report their progress in weekly meetings.
File(s) get corrupted	A team member is working on the project when the file corrupts	Regularly commit work to branch on GitHub so it can be easily sourced again	The front-end lead (Matan) and back-end lead (Ojas) will ensure that each side of the project follows sound Git principles, and hold members accountable if they do not.
Communication becoming too slow or reactive	A team member needs clarification on a task to proceed and is not getting responses in time to be able to make progress	If you are pinged in a message, respond within 24 hours	Regular meetings are scheduled so if a team member does not respond there is still plenty of opportunity to communicate.
Crucial app component is delayed and blocks others progress	The API for PDF retrieval is not implemented, so the frontend annotation work is blocked	Design components to have minimal coupling, and mock or stub with fake data to unblock work	The team will collectively enforce this during code reviews and ensure that there are clear API boundaries between modules.
Disagreements or differences in opinion on project direction	The team is split between the app being dark or light themed	Vote on the decision to be made	Upon disagreement, Matan will organise a meeting, listen to both sides and vote on the issue.