Project Report - Team 41

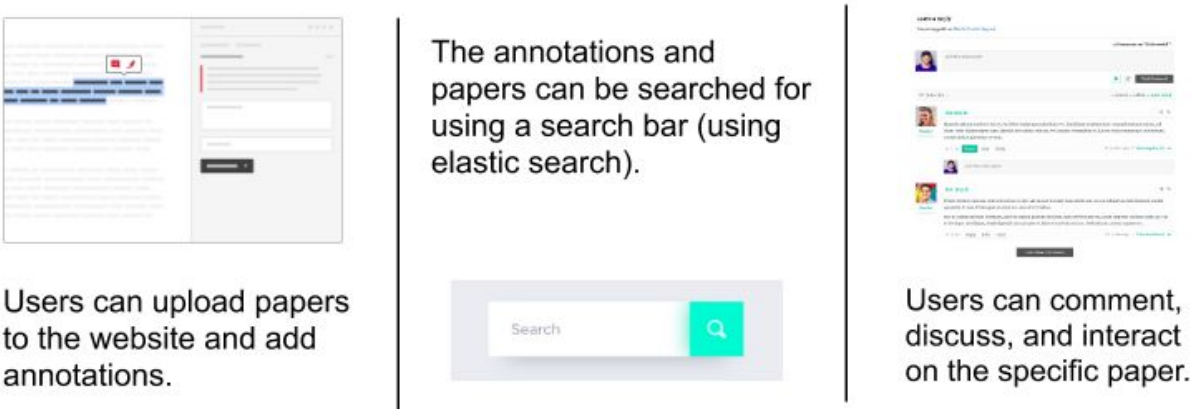
Team members

Manvith Reddy (2018101057)
Shanmukh Karra (2018101111)
Karsh Tandon (2018101034)
Mrinal Khubchandani (2018101053)

TA Mentor: Vijayraj Shanmugaraj
Client: Descign (Dr. Vishal Gupta, Shivani S, Bharath J)

Brief Problem Statement

An important aspect of collaboration is to read and discuss published literature. Current tools force researchers to do this in isolation. We are proposing to build a prototype ‘Reddit for Research’ where users can read and discuss interesting research in a seamless manner which is embedded in their daily workflow.



Features required

1. Allow opening a document (e.g. PDF) in a web-browser
2. Integrate an open-source annotation and commenting tool
3. Allow multiple users to read, discuss and comment on the document
4. Intuitive UI to enable end users to have a seamless experience

Profile of Users

The users of this platform will primarily be researchers, who will use the platform to share and collaborate over research papers/other literature. However, regular people can use it as well, to upload regular articles/PDFs, and then have discussions over the same.
Keeping the target audience in mind, and considering the platform is meant to collaborate over research-work, the platform doesn't have any complex-to-use features.

Task delegation

- Document upload, PDF Viewer: Karsh, Mrinal
- PDF Viewer transition: Manvith, Shanmukh
- Search (ML + Elasticsearch): Manvith
- Docker/Kubernetes/AWS: Shanmukh
- Building the platform (authentication, search, frontend, categories): Manvith, Shanmukh
- Hypothes.is integration: Manvith, Shanmukh, Karsh, Mrinal
- Data science pipeline tool testing: Manvith, Shanmukh (miscellaneous)

Team and Client Communication

- In the first half of the project, teammates met and communicated with the client regularly in-person at the Descign office located at CIE IIIT-H.
- Meetings with the client were held on a weekly basis and communication was mostly done with the engineering team at Descign.
- Due to the coronavirus pandemic, the second half of the project was primarily conducted online
- Meetings were held on Google Hangouts bi-weekly, and information was shared on social channels such as WhatsApp (in the presence of the TA mentor)
- Documentation was shared primarily through Google Drive.

Development Environment

- Git/GitLab for collaboration
- Express.js and Node.js framework for backend (Node.js v10.19.0 was installed using apt, and Express.js v4.17.1 was used)
- MongoDB for database layer (v4.2.5)
- JavaScript - Angular8 framework for frontend components (Angular v8.0.6)
- Elasticsearch for search (v.7.6.2)
- Flask and Python3 for ML-Search demo webapp
- Jupyter, Kubernetes, Nuclio.io, and Kubeflow for data science pipelines
- Google Docs, Pages - project documentation

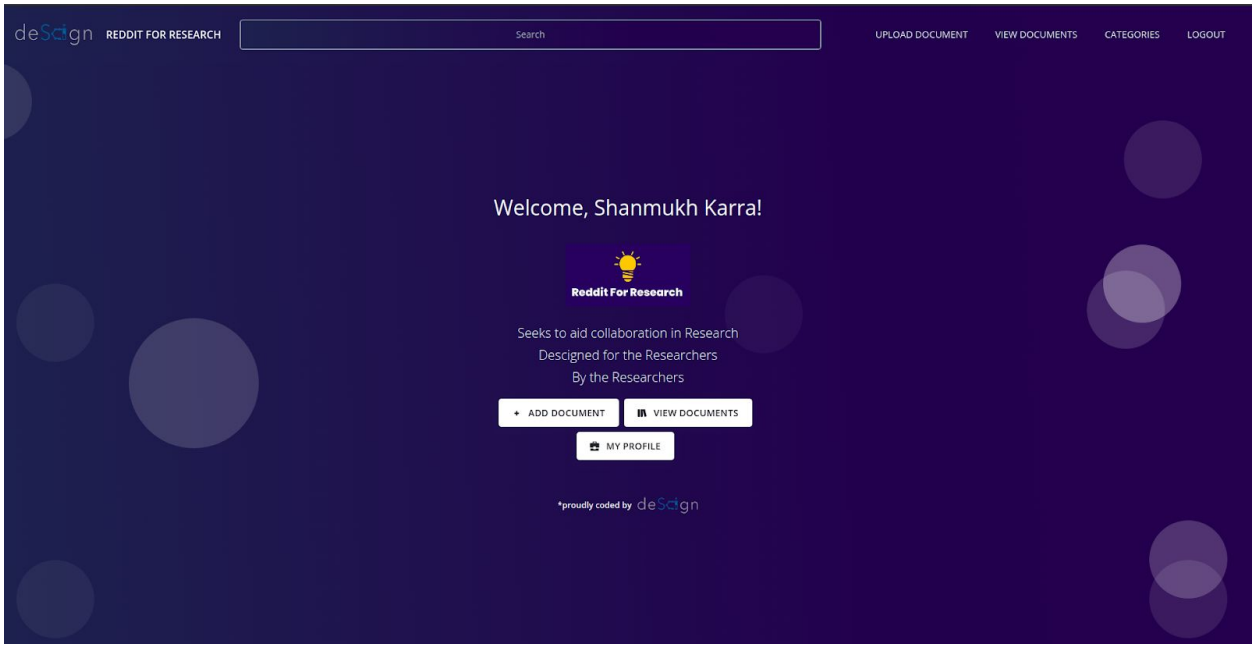
Software Requirements Specification

- User Management (secure registration and login)
- Upload document (store in backend)
- PDF Viewer (view documents using inbuilt PDF viewer)
- Annotations (highlight particular subsections of a document and annotate them)
- Commenting (comment on an uploaded document)
- Highlighting (highlight particular subsections of a document without annotating)
- Reply (reply to other annotations/highlights/comments, support **nested** replies)
- Edit (edit annotations/highlights/comments)
- Delete (delete annotations/highlights/comments)
- Search (ability to search through different documents and different annotations/highlights/comments)
- User interface (webapp must have a clean responsive UI, and must be easy to navigate)
- Profile (user profile that contains information about the user along with all his posts)

Installation instructions

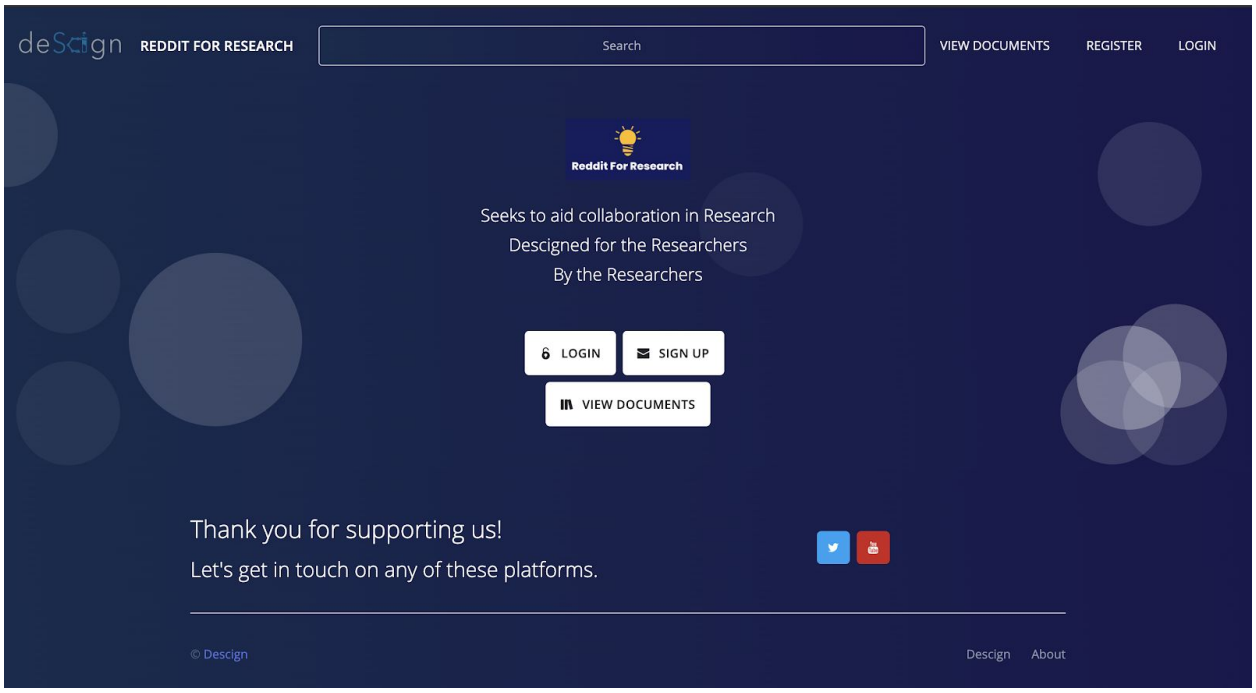
1. Install MongoDB (instructions: https://docs.mongodb.com/manual/installation/). Ensure it has installed successfully
2. Install node.js and npm (instructions: https://nodejs.org/en/download/)
3. Install elasticsearch (instructions: https://www.elastic.co/guide/en/elasticsearch/reference/current/deb.html) Run it using 'sudo systemctl start elasticsearch.service'
4. Install mongo-connector and elastic2-doc-manager using pip
5. Run the following commands to set up MongoDB and Elasticsearch (step 5 can be also completed by following the instructions from this article, but be careful to run 5c as mentioned here https://medium.com/@xoor/indexing-mongodb-with-elasticsearch-2c428b676343):
    a. sudo mongod --dbpath /var/lib/mongodb --replSet rs0 (command should stay running)
    b. Open the mongo shell, and type in 'rs.initiate()'
    c. mongo-connector -n Reddit-for-Research.documents -m 127.0.0.1:27017 -t 127.0.0.1:9200 -d elastic2_doc_manager (command should stay running)
6. Clone the Reddit-for-Research code, which has the src/ folder with all the frontend code, and the backend/folder with all the backend/database code. Then run the following commands to get the backend and frontend running
    a. cd Reddit-for-Research
    b. npm install; npm start
    c. cd backend; npm install; npm start (**OR** nodemon server.js, where nodemon is a package that can be installed using npm)
7. The webapp should be successfully up and running. Register an account, and upload a document. Run the 'curl localhost:9200/_cat/indices' command to check if Elasticsearch has successfully indexed the Reddit-for-Research database. If it doesn't show, please redo step 5 carefully

Reddit for Research preview

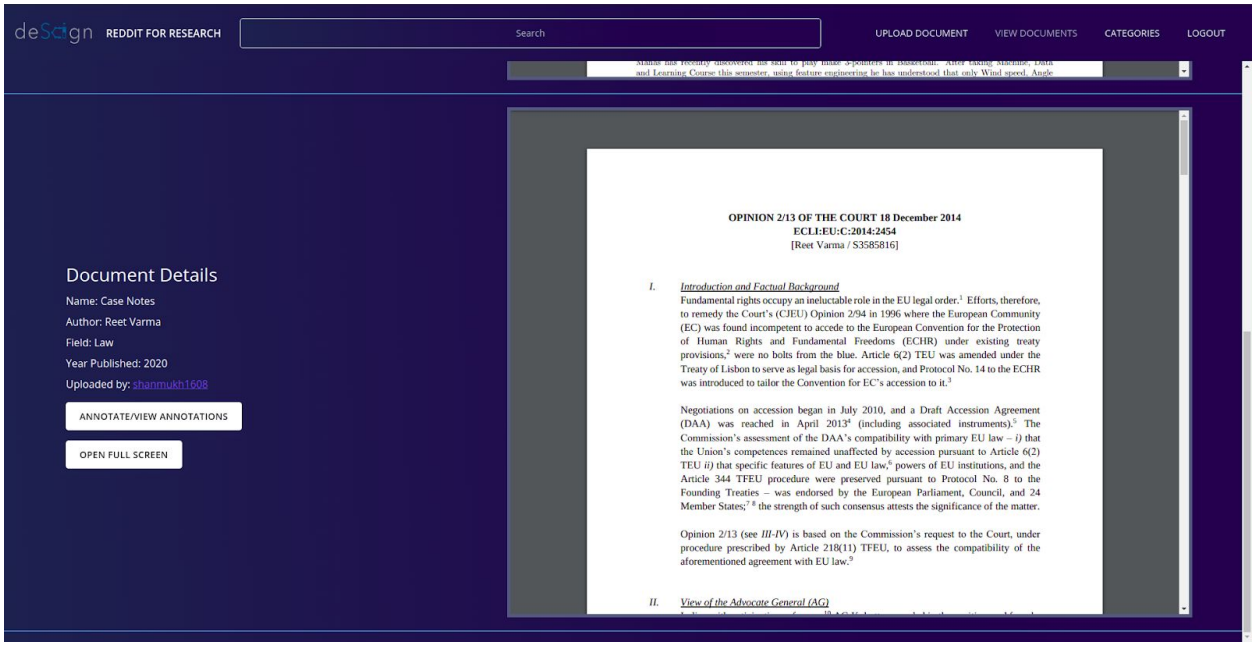


Home page (Logged in)

The home page is the central hub of Reddit for Research. Our aim was to give the end user the liberty to travel to all the important sections of our product right from the home page. There are 2 views to the homepage, the one above which is displayed if the user is logged in, and the one below which is displayed if the user is a Guest.



Home page (Logged out)

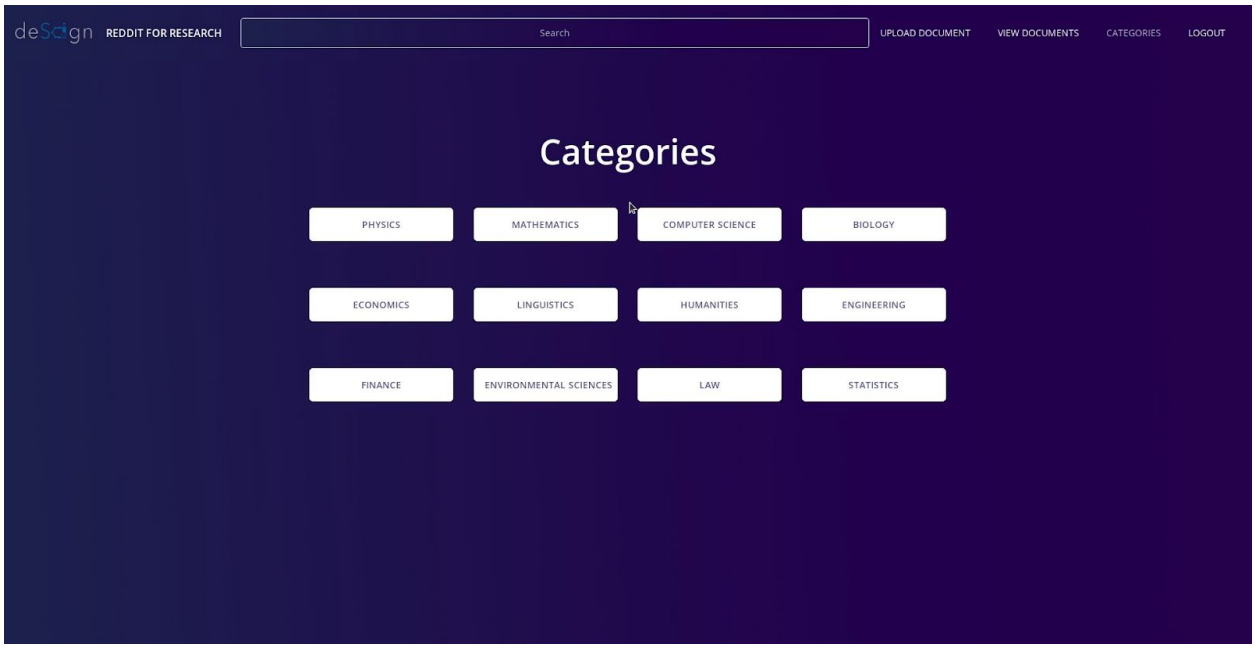


Document view

The “View Documents” page enables any user (Logged in/out) to view the documents in the reverse chronological order of upload. We display the documents to Logged out users as well which is a conscious design choice.

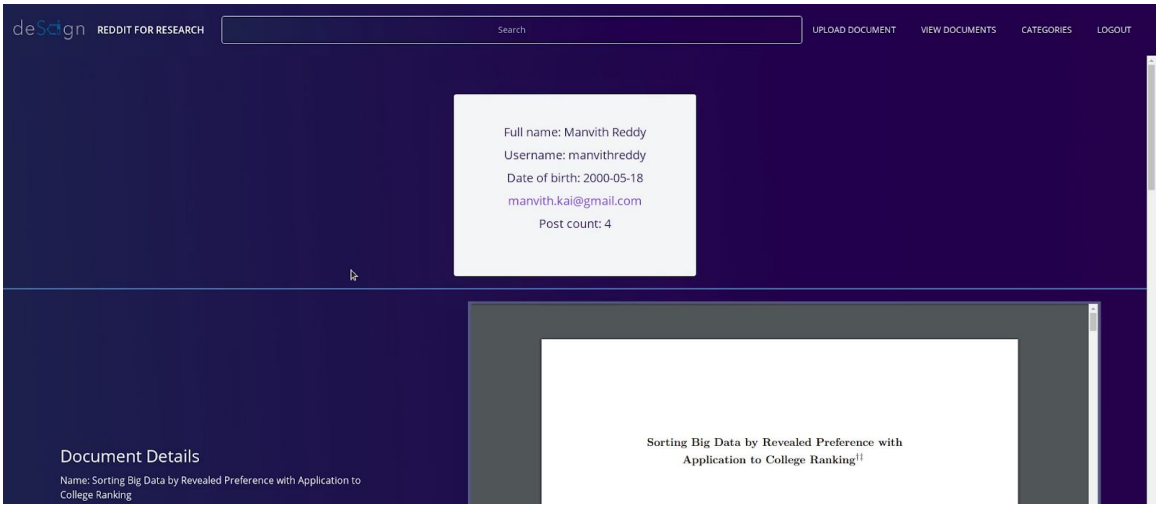
Search Bar and advanced options

Using ElasticSearch and its API's, we enable the end user to a plethora of search options. A simple search involves fuzzy string matching with great speed. The user also has the option for an "Advanced Search" which enables the user to search through multiple fields as well as search through documents of a given time period simultaneously.



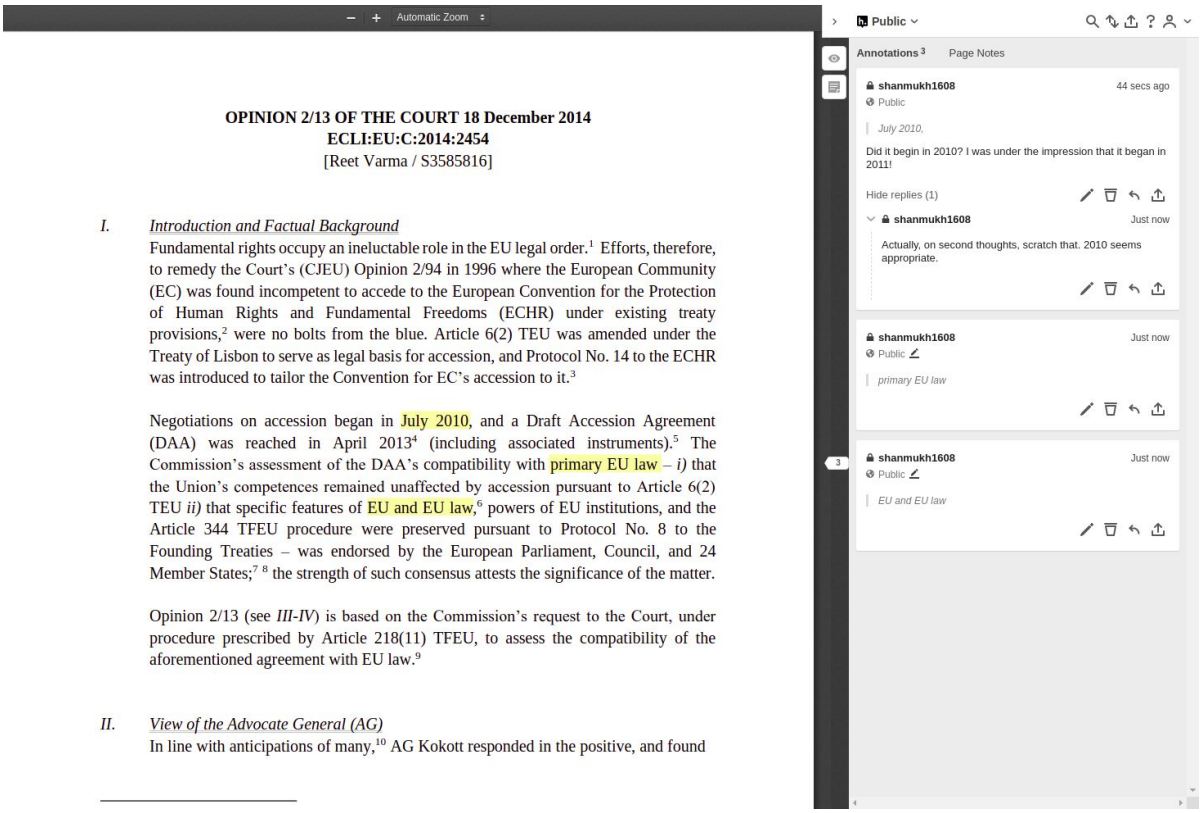
Categories

The categories page enables the user to browse the various fields of research at the ease of a click. The backend of this functionality is also run through ElasticSearch to enable sub categories to feature through the main category. (For example: Mechanical Engineering would show up in Engineering).



Profile

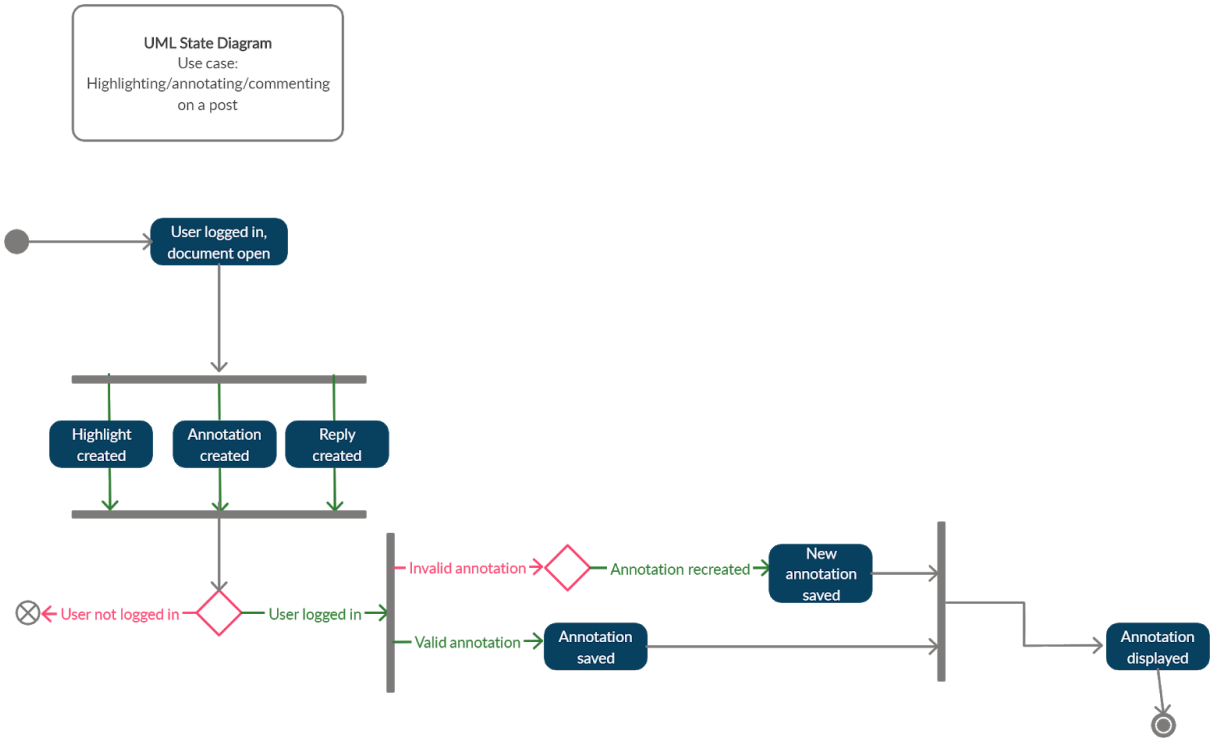
The profile page displays information about every user, his email address for contact and his post count. It also displays the documents in the reverse chronological order of upload.



Headline feature - annotations and highlights, with nested replies

Our most integral feature that ties our project together is our Hypothesis annotator embedded into our PDF viewer. This allows researchers to annotate and highlight and part of the research paper. The annotator also allows comments and replies for each annotation/highlight, enabling the communication between researchers which we strived to provide to our users. It also has an ElasticSearch Engine for searching through annotations and highlights. The stream is a standalone mode of the Annotator. It listens to the service on the websocket, and displays annotations in real time as they arrive.

The diagram below is a UML state diagram which displays the flow of the product all the way from logging in to creating a highlight/annotation or comment. The product is built in such a uniquely simple way that all these three features follow a similar flow till they reach their intended functionality state.



UML State Diagram of Highlights, Annotation and Commenting

More features and the entire functionality demonstration can be found in our demo video, which has been submitted alongside this report.

Project Journey

Initial Phase - Understanding Scope of Work

In the initial phase, time was spent:

- Understanding the scope of the project and what features were to be developed - This was probably the most integral phase of the project because the time we put in to achieve an air-tight SRS exponentially decreased our work in the coming weeks and made this project achievable.
- Understanding the aim of the project and the intended users - A great deal of time was spent in achieving our principal goal, designing a product that the end user would find easy to use and collaborate. Our main user base would be researchers from all domains and that's why we ensured the platform doesn't have any complex-to-use features.
- Developing ideas related to implementation - The design documentation was crafted with great detail to ensure that when we went on to build the product, we left as few holes as possible and had a seamless coding phase of the SDLC.
- Distributing workloads and understanding strengths of each team member - Our entire product was built heavily based on the "Separation of Concerns" design principle. At every layer of abstraction, issues were spread amongst the team based on his strengths. This made every teammate accountable and responsible for his work, leading to a healthier development environment.

Release 1

During the road to R1, the work done by the team was divided into different domains. Karsh and Mrinal were handling the File Upload + Annotations Webapp; Manvith was working on ML search and Elasticsearch; Shanmukh was working on Docker, Kubernetes, and AWS deployment. For all of us, most of these domains were extremely new and it was the first time that we worked on such a large scale project while collaborating as a team. Although it took us a week or two to familiarize ourselves with the respective frameworks, once we

began working it all started falling into place. We followed a bottom-up integration approach with each of us having to build the core elements of what would together be our final product for R2. The deliverables for R1 were:

- Reddit-for-Research webapp
    - Document upload
    - PDF Viewer
- ML Search Flask webapp
    - Uses Birch algorithm and hierarchical clustering
    - Tested on open source technical Qualcomm dataset
- Elasticsearch MERN-stack webapp
    - Uses Elasticsearch APIs with configurable options
    - Tested on research paper dataset
- Dockerized MEAN-stack webapps deployed on AWS

The R1 deliverables were submitted and presented in the first week of March, and were approved by the client and the TA mentor.

Release 2

The development for R2 began in the second week of March. Manvith and Shanmukh were assigned work on data science pipelines, while Karsh and Mrinal were expected to make progress on the primary project.

Manvith had the task of using KubeFlow to explore the possibilities of using a Jupyter Hub notebook with large datasets and running the pipelines in an automated fashion. Using nuclio.io, Shanmukh explored the same concept but using serverless function tech for compute. This research was conducted alongside the primary project to evaluate different tools' viability for the client for future use.

Karsh and Mrinal began exploring the concept of integrating Hypothes.is into our Document Upload webapp built in R1. However, due to the abrupt shutting down of university, followed by the nationwide lockdown, there was a change in plans. The onus was on Manvith and Shanmukh to see it through.

Integrating Hypothes.is for the PDF viewer turned out to be our hardest challenge because of the lack of support Hypothes.is had for PDF viewers. The 2 of us worked on this problem for 2 weeks and explored a plethora of options and at one point were on the verge of giving up on Hypothesis. However, we managed to successfully integrate Hypothesis with the PDF.js viewer, solving one of our biggest tasks.

The focus was then shifted to integrating the Elasticsearch APIs which was easily achievable since we were only required to make minor modifications to our R1 submission. Following the successful completion of these tasks, work was put into building an application with a solid frontend, and then ensuring rigorous testing to attain product acceptance.

The entire Reddit-for-Research webapp with all its functionalities forms the list of deliverables for R2:

- Document upload
- PDF Viewer
- Secure authentication
- Annotations
- Highlighting
- Commenting
- Replies (nested)
- Editing
- Deletion
- Search with advanced options
- User Interface
- Profile pages
- Categorization of documents based on field chosen

Challenges Faced

As the project lasted over 3 months, we faced several challenges of varying levels of intensity throughout the project timeline:

- Lack of experience working in a professional development environment
- Difficulties collaborating on large-scale product

- Adapting to the agile-scrum methodology
- Understanding different frameworks and software (Angular, Elasticsearch, Docker, Flask, MongoDB, etc) as per client requirement
- Understanding Machine Learning algorithms like Birch and Hierarchical Clustering.
- Comparing different annotation tools, search implementations, and cloud service providers
- Integrating Hypothes.is into the project
- Shanmukh was sick and underwent surgery.
- Karsh and Mrinal dropped out of the project after college shut down in March.
- Nationwide lockdown impacted the way we worked and communicated (amongst the team as well as with the client)

Crucial Learning

The project taught us a lot, as we contended with different issues throughout the development and testing phases.

- Learnt several frameworks/software/tools (Angular8, MEAN-stack, Docker, Kubernetes, Flask, Elasticsearch, Git, Postman, Selenium, Loadrunner)
- Understanding the importance of Design and Documentation and its importance in the software development life cycle.
- Manvith learned a lot of machine learning in the process of building his search.
- All of the team members learnt a lot about the world of web development and feel ready to take on a role as full-stack developers.
- We learned the importance of meeting deadlines for a proper workflow.
- We learned how the agile methodology enabled us to catch up on unfinished work while keeping up with new challenges.
- Understanding requirements and developing projects to meet requirements
- Interacting with clients and the importance of consistent communication

Unfortunately, not all the experiences were pleasant, and we have also experienced that:

- Collaboration is difficult, especially with varying levels of commitment from different members of the team towards the project.
- The coronavirus pandemic took away the comfort of working together in the same physical space. This led to us resetting communication methods and some slowdown of work from the initial momentum we had. However, we soon overcame this and were able to get the job done.