PROCRASTINATE PROJECT REPORT

NUS MASTER OF TECHNOLOGY
INTELLIGENT REASONING SYSTEM (IRS)

BROUGHT TO YOU BY TEAM PROCRASTINATE

Hou Lu Chiok Weh Alejandro	A0214865Y
Michael Yang ZiChang	A0206039L
Lee Jane	A0214865Y

TABLE OF CONTENTS

EXECUTIVE SUMMARY

In traditional undergraduate and graduate studies, lectures are often perceived as a necessary evil, yet recent research suggests that their effectiveness, especially in STEM fields, is questionable. Many students find extended lecture sessions tedious and unproductive. However, some of the most successful students employ strategies such as effective notetaking and mind mapping to extract key insights from lectures, enhancing their learning experience and overall academic performance.

Recognizing that lectures remain a predominant method of instruction in universities, we propose Procrastinate, a comprehensive framework and application designed to empower students to maximize the value of their lecture time. By leveraging advanced technology, our solution aims to transform recorded lectures into concise summaries and dynamic mind maps, providing students with invaluable resources for review and study.

Using Whisper's transcription capabilities, Procrastinate generates a complete transcript of recorded lectures. Subsequently, employing a BART-based Facebook CNN model, we distil this transcript into succinct text summaries, offering students a comprehensive overview of the lecture's content. To further facilitate comprehension and retention, we employ transcripts to construct visually engaging mind maps, aiding students in organizing and contextualizing key concepts.

Our team envisions that Procrastinate will empower students to extract meaningful insights from their daily lectures, thereby optimizing their learning experience and academic outcomes. By equipping students with efficient tools for review and study, we aim to enhance their engagement in the learning process and promote academic success.

1

BUSINESS JUSTIFICATION

When we embarked on selecting our project, we sought an application that consolidates knowledge and information within the educational technology (ed tech) sector. Given the widespread adoption of OpenAI's ChatGPT, functionalities like summarization and text generation have become more accessible, offering invaluable support to students in their daily studies. However, our exploration uncovered two significant challenges associated with utilizing ChatGPT:

Firstly, when students pose general queries, particularly regarding homework, ChatGPT's responses may lack contextual accuracy, posing a risk of misinformation. Notably, ChatGPT's remarkable ability to craft compelling arguments can make erroneous content appear convincing, leading to unintended consequences for users.

Secondly, certain advanced features of ChatGPT, such as file and image uploading, are restricted behind a paywall, creating a financial barrier for students seeking access to tools that could enhance their learning experience. Our objective is to mitigate or eliminate this obstacle wherever feasible, underscoring the potential of our app as a viable solution.

By addressing these challenges, we aim to foster a more conducive learning environment, empowering students with equitable access to educational resources and facilitating their academic growth.

PROJECT TEAM

PROJECT OBJECTIVE

The primary goal of this project is to develop an application that offers a comprehensive suite of tools, including summarization and mind mapping, to support students in their study endeavors and enhance the effectiveness of their daily learning routines.

Our approach involves leveraging recorded lectures attended by students, extracting pertinent information from the transcripts we generate, and utilizing this data to generate concise notes and dynamic mind maps.

By providing students with organized and visually engaging resources derived from their lectures, we aim to facilitate deeper understanding and retention of key concepts, ultimately enriching their learning experience.

TEAM MEMBERS

Hou Lu Chiok Weh Alejandro

- Project Tech Lead
- Project manager
- System design architect
- Database design
- Integration
- Backend Springboot
- Backend Django
- Database MySQL
- Amazon S3 bucket
- Deployment
- Project video production
- Project report (System design section and Appendix A, B, C)

Lee Jane

- Frontend UI
- Front end model management
- API integration
- Project proposal
- Project Report writing
- User guide

Michael Yang ZiChang

- Whisper transcription model development and testing
- Summarization Model Development
- Automatic Knowledge Base and Mind Map creation
- LDA model development
- Project Report Writing

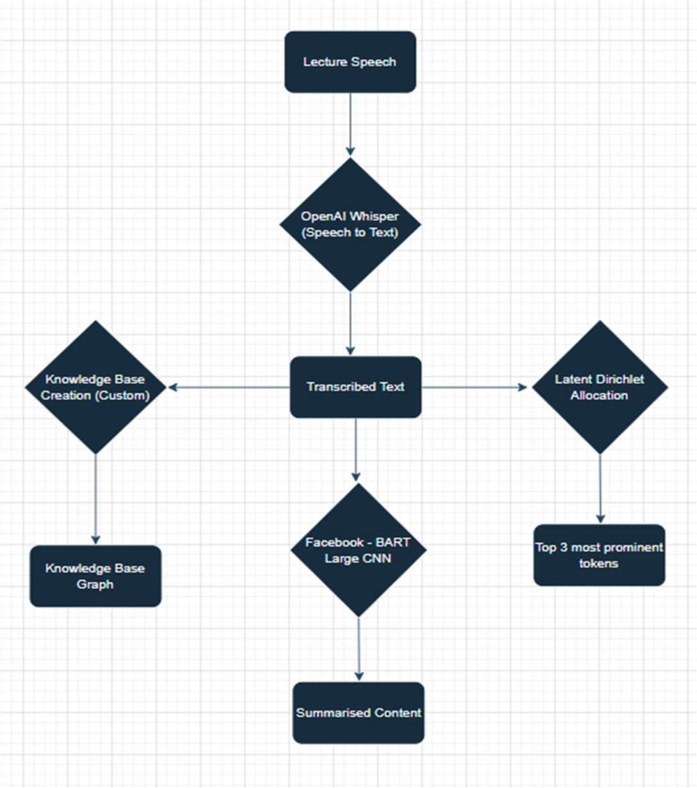
PROJECT SOLUTION

PROJECT DELIVERABLES

APPLICATION FEATURES

The Procrastinate App is a pipeline of frameworks that take in a voice recording (.wav, .mp3, .m4a) file and transcribes it for a variety of analysis purposes. Once we acquire a text file (.txt) format, we take the text file and perform various analysis tasks on it. The main features include text summarization for an overview of the lecture, Knowledge Base mind map for the major topics covered and the Latent Dirichlet Allocation analysis should the student want to research further on each of the topics covered.

We cover the general architecture in the graphic below:

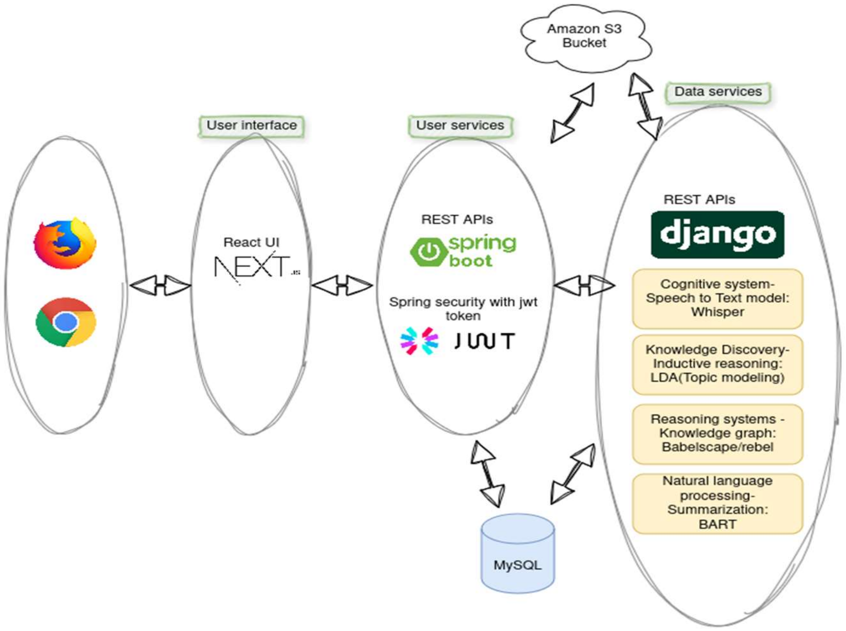

Lecture Speech
OpenAI Whisper (Speech to Text)
Knowledge Base Creation (Custom)
Transcribed Text
Latent Dirichlet Allocation
Knowledge Base Graph
Facebook - BART Large CNN
Top 3 most prominent tokens
Summarised Content

If the user has text data on their lectures, they can use the text data directly in our application to tap into the various features such as summary, Latent Dirichlet Allocation and our knowledge base creation module. Alternatively, the user could upload their voice recording of the lecture and use the Whisper Model to generate a transcript of the lecture and use that as an input for the various features. For the POC, we used the CMU Computer graphics course to test the case where we have the voice recording of the lecture content.

APPLICATION BUSINESS FLOW

Our application is a workflow system that uses speech cognitive systems to perform transcription for the lecture content after a user has sent in their lecture recording audio file. Through a knowledge-based system and a large database from wikipedia, we are able to create a set of relatively compact knowledge graphs by culling entities based on the wikipedia references of each entity that was detected from the extracted text. The text summarization function returns a summarised version of the lecture that is around 30% (Tunable by user) of the original lecture length. We also created a set of LDA analysis tools so the user can search up the top hits within the lecture for further studies after their lecture.

SYSTEM DESIGN, ARCHITECTURE & IMPLEMENTATION


Amazon S3
Bucket
Data services
User interface
User services
React UI
NEXT.js
REST APIs
spring boot
Spring security with jwt token
JWT
REST APIs
django
Cognitive system- Speech to Text model: Whisper
Knowledge Discovery- Inductive reasoning: LDA(Topic modeling)
Reasoning systems - Knowledge graph: Babelscape/rebel
Natural language processing- Summarization: BART
MySQL

Nextjs is our frontend client, springboot is our backend service, taking care of user services and security and django application will be on the data services. On the database side, we used MySQL as our relational database and we also use the Amazon S3 bucket

to store all our files so that it is easily accessible to be uploaded and downloaded by all our services. Finally the 4 models/techniques which we have selectedas the backbone of our system. A speech to text model, Latent Dirichlet allocation, knowledge graph and natural language processing.

USE CASE DIAGRAM

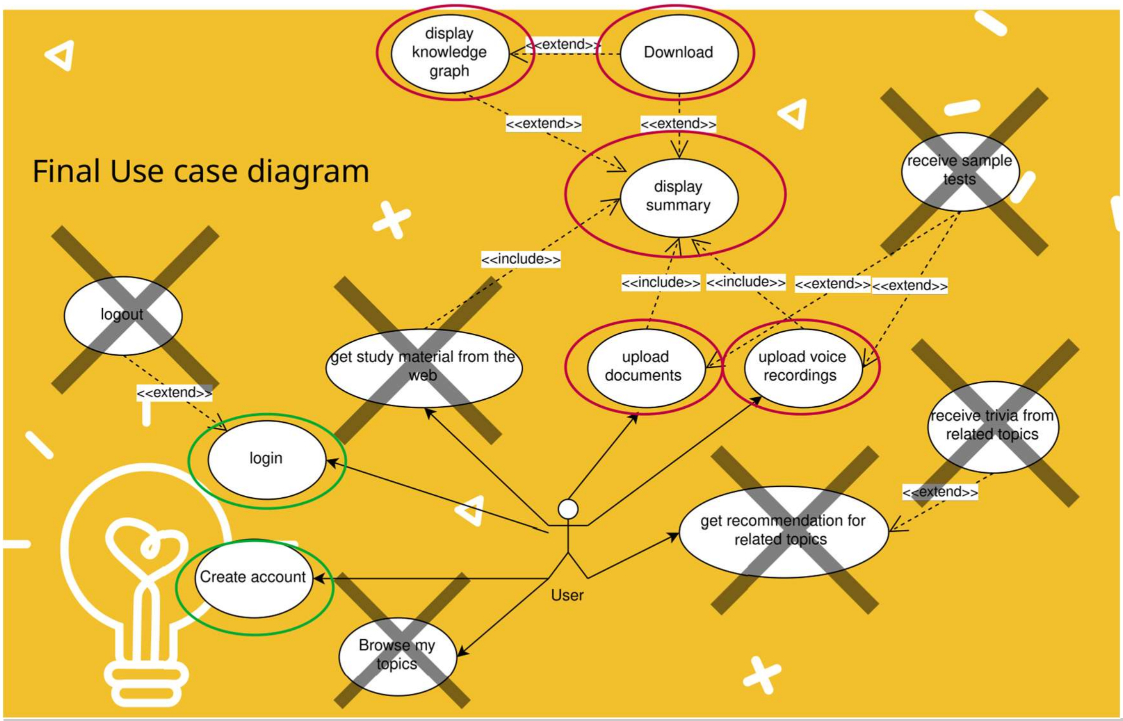


Fig 5.1 - Final Use case diagram

Our use cases were trimmed down as the project progressed in order to meet a realistic deadline. However, plans for the use cases that were dropped were all taken care of in the database where all the relevant data were saved and preserved for their implementation.

Use cases in the red bubble are our core use cases and those in green are enhancements to emulate a right world application.

DATABASE DESIGN (ENTITY DIAGRAM)

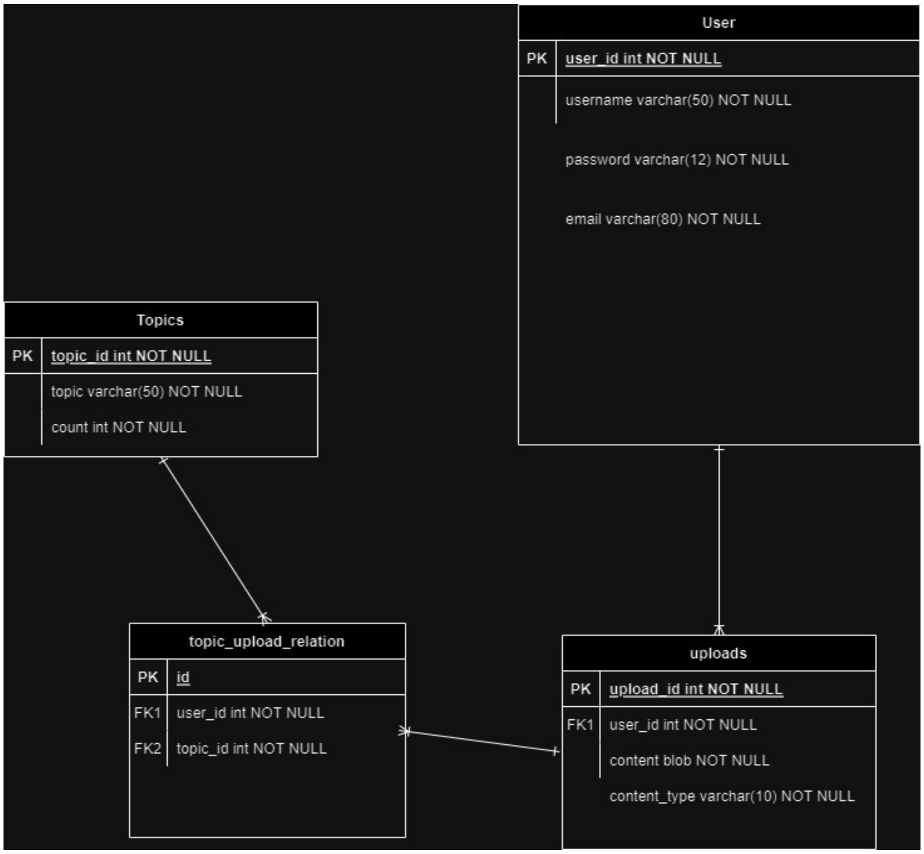
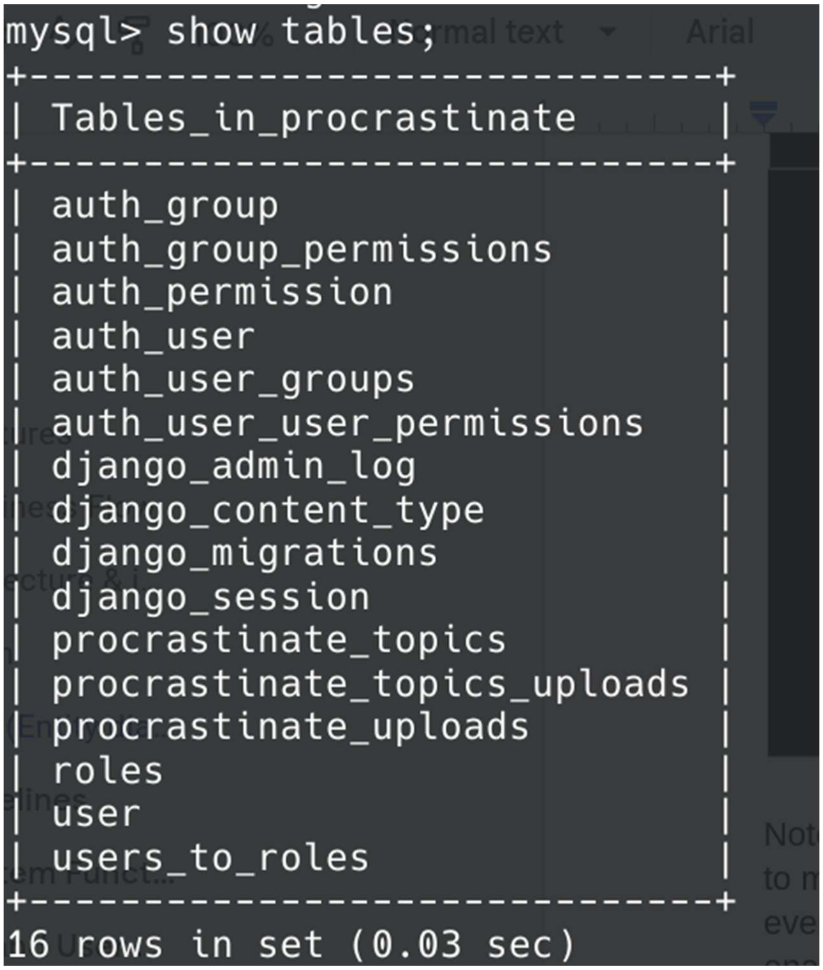


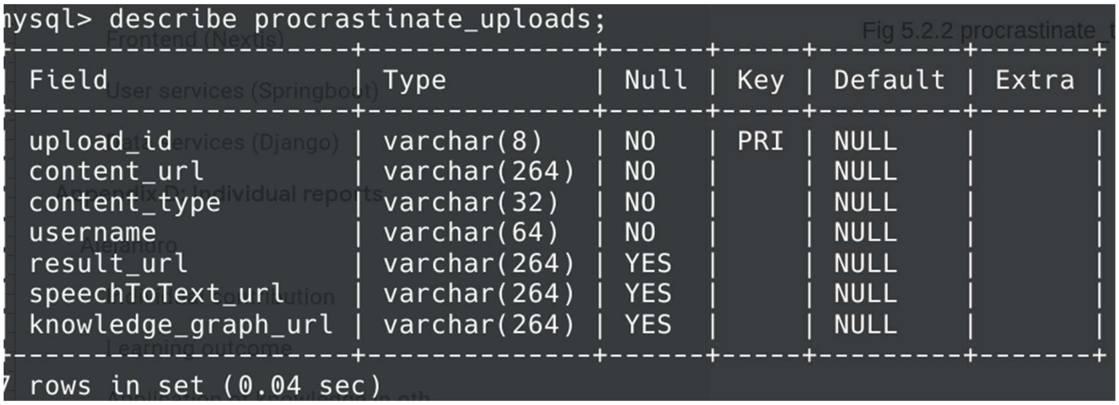
Fig 5.2 Entity diagram

Note that ‘Topics’ and ‘Uploads’ are in a many to many relationship. This is because we want to make every topic unique to not flood the database with thousands of repeated topics. So every Upload can have many Topics and every Topic can belong to many Uploads. This will enable us to do a count of topics across all user’s uploads to know which topics the user is most in need of study material. This will also enable us to, in the future, match users with similar topics using the KNN model to find them study buddies.

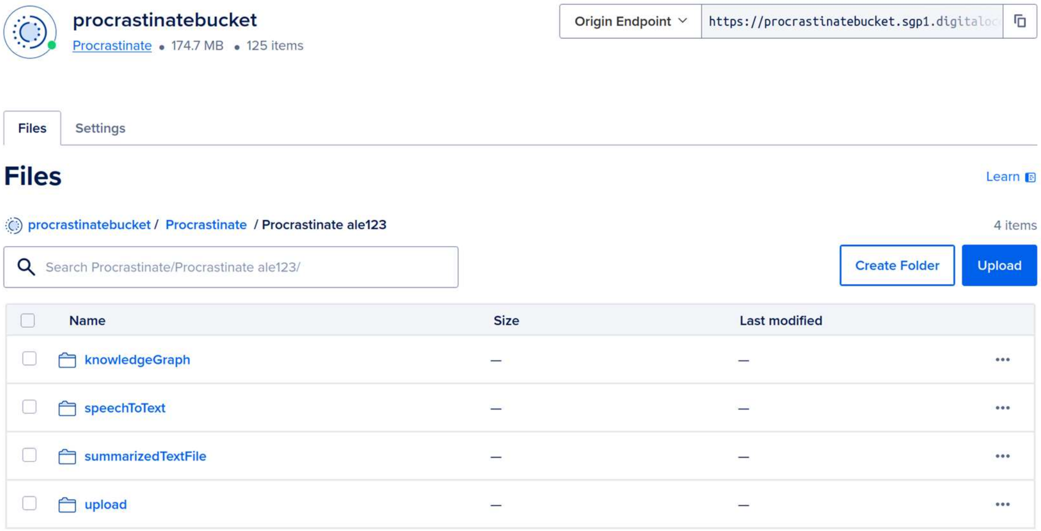


```
mysql> show tables;
+------------------------------+
| Tables_in_procrastinate      |
+------------------------------+
| auth_group                   |
| auth_group_permissions       |
| auth_permission              |
| auth_user                    |
| auth_user_groups             |
| auth_user_user_permissions   |
| django_admin_log             |
| django_content_type          |
| django_migrations            |
| django_session               |
| procrastinate_topics         |
| procrastinate_topics_uploads |
| procrastinate_uploads        |
| roles                        |
| user                         |
| users_to_roles               |
+------------------------------+
16 rows in set (0.03 sec)
```

Fig 5.2.1 Tables



```
ysql> describe procrastinate_uploads;
---------------------+--------------+------+-----+---------+-------+
 Field               | Type         | Null | Key | Default | Extra |
---------------------+--------------+------+-----+---------+-------+
 upload_id           | varchar(8)   | NO   | PRI | NULL    |       |
 content_url         | varchar(264) | NO   |     | NULL    |       |
 content_type        | varchar(32)  | NO   |     | NULL    |       |
 username            | varchar(64)  | NO   |     | NULL    |       |
 result_url          | varchar(264) | YES  |     | NULL    |       |
 speechToText_url    | varchar(264) | YES  |     | NULL    |       |
 knowledge_graph_url | varchar(264) | YES  |     | NULL    |       |
---------------------+--------------+------+-----+---------+-------+
7 rows in set (0.04 sec)
```



procrastinatebucket

Procrastinate • 174.7 MB • 125 items

Origin Endpoint `https://procrastinatebucket.sgp1.digitaloc`

Files Settings

Files

Learn

procrastinatebucket / Procrastinate / Procrastinate ale123

4 items

Search Procrastinate/Procrastinate ale123/

Create Folder Upload

□	Name	Size	Last modified	
□	knowledgeGraph	—	—	•••
□	speechToText	—	—	•••
□	summarizedTextFile	—	—	•••
□	upload	—	—	•••

Fig 5.2.3 Amazon S3 bucket

The S3 bucket is a place where all the files - uploaded document, knowledge graphs, summary output and speech to text are stored. It is easily accessible by all members of the team and also easily downloaded by any of our applications. This is to lighten the workload of our application and prevent encoding, decoding byte arrays for blob type data.

WORKFLOW (SEQUENCE DIAGRAM)

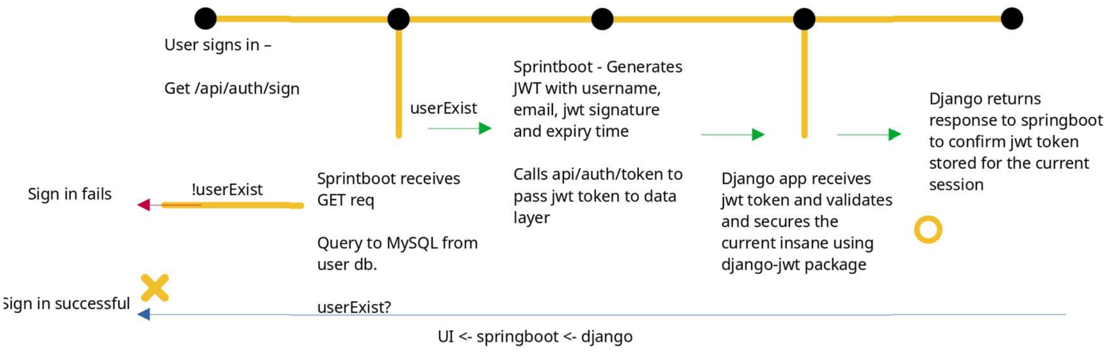
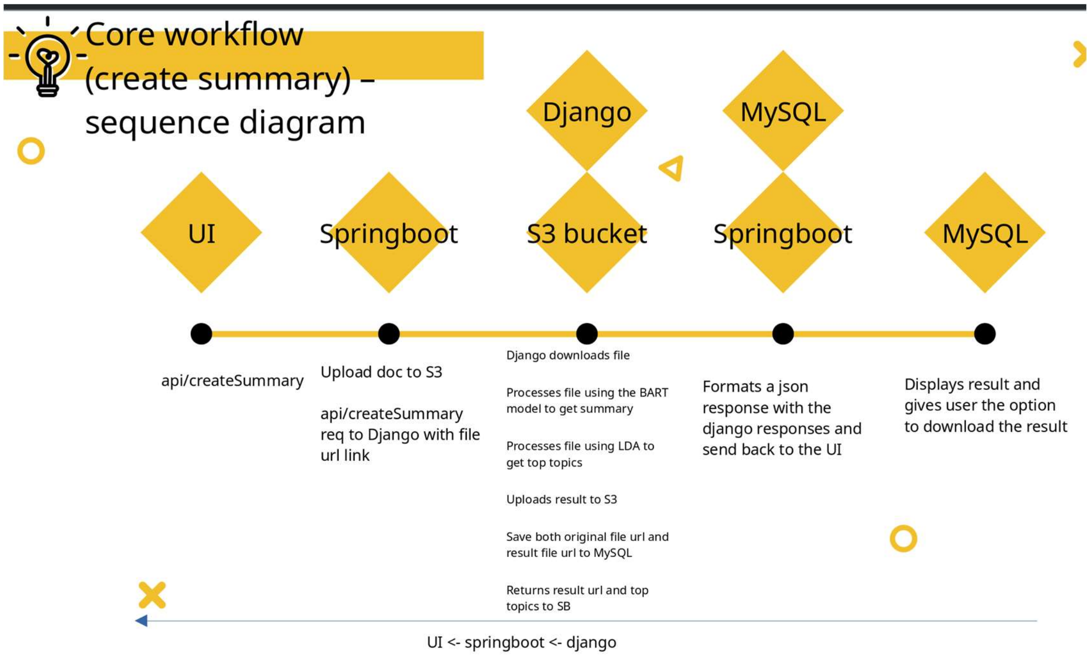

User signs in –
Get /api/auth/sign
userExist
Sprintboot - Generates JWT with username, email, jwt signature and expiry time
Calls api/auth/token to pass jwt token to data layer
Django returns response to springboot to confirm jwt token stored for the current session
Sign in fails
!userExist
Sprintboot receives GET req
Query to MySQL from user db.
userExist?
Django app receives jwt token and validates and secures the current insane using django-jwt package
Sign in successful
UI <- springboot <- django

Fig 5.3.1 Use case sequence diagram (Sign in)

The first and most important workflow with the application starts off with the user signing in from the user interface. That sign in information will be sent over to our springboot application where all the usual validations are made to confirm if the user exists or not. Then generate a JWT (json web token) which will be passed to the Django application. Once the data service gets the jwt token, it will make a response back to springboot and springboot back to the UI to confirm that our session is now secured and locked in for this user. Henceforth the frontend will be able to validate all requests with that JWT.



Core workflow (create summary) – sequence diagram

Django

MySQL

UI

Springboot

S3 bucket

Springboot

MySQL

api/createSummary

Upload doc to S3

api/createSummary req to Django with file url link

Django downloads file

Processes file using the BART model to get summary

Processes file using LDA to get top topics

Uploads result to S3

Save both original file url and result file url to MySQL

Returns result url and top topics to SB

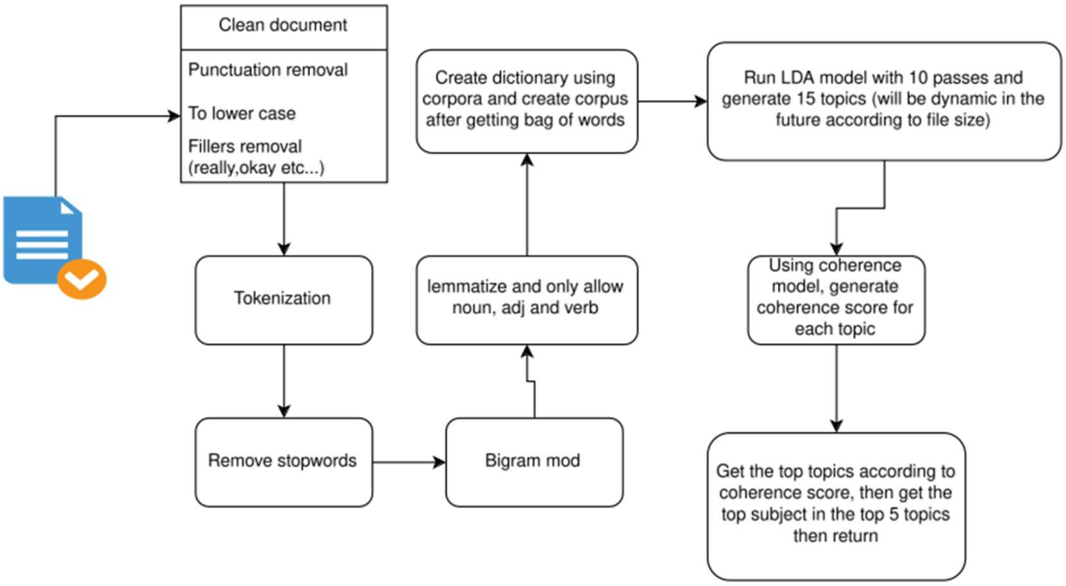
Formats a json response with the django responses and send back to the UI

Displays result and gives user the option to download the result

UI <- springboot <- django

FIG 5.3.2 USE CASE SEQUENCE DIAGRAM (CREATE SUMMARY)

One of our core features is the create summary feature. It begins with the user uploading a document and clicking on the summarise button. The UI will send all the files to springboot where it uploads the file to our S3 bucket, retrieves a url and sends it to our data service for processing. With the url, our data service will then download the file from S3 bucket and this file will go through 2 processes. The BART model from facebook will work its magic to summarise the document while the LDA model will extract the top topics from that document. The results will once again be uploaded to another folder in S3 bucket and all related URLs will be saved into our database. Result url will be returned to springboot then to the UI to be displayed and downloaded.


Clean document
Punctuation removal
To lower case
Fillers removal (really,okay etc...)
Tokenization
Remove stopwords
Bigram mod
lemmatize and only allow noun, adj and verb
Create dictionary using corpora and create corpus after getting bag of words
Run LDA model with 10 passes and generate 15 topics (will be dynamic in the future according to file size)
Using coherence model, generate coherence score for each topic
Get the top topics according to coherence score, then get the top subject in the top 5 topics then return

The LDA model is where we extract the top topics from user uploaded documents. It starts off with our classic text preprocessing techniques. The biggest thing to note here is that we rely heavily on the coherence score to determine which topic we use. From those topics, which subjects we are selecting. In short, we are taking the top subjects (for example, computer science) from the top topics sorted by coherence scores. For future development, these topics will be used in various ways. Study recommender, web scraping for study material like YouTube and test generation.

Lastly, the knowledge graph is created on top of the OpenNER (Named Entity Recognition) framework. Each NER output is structured with the Part of speech tags of the sentence. We take the first entity that is recognized and place it as a head of the argument (Depending on the context of the entity recognized, that might change), we attempt to place relations on the other entity found within the sentence, We then build iteratively based on the building blocks set up from the first few initial records. For each subsequent entity and relation, we check for the presence of the same entity within the knowledge graph in the first place. In order to create a more succinct knowledge graph, we tap into Wikipedia data and if the entity was also mentioned in related entries in Wikipedia, we treat it as a similar entity and merge the node entries together before building the relational edges.

PROJECT PERFORMANCE

A major consideration regarding the day-to-day running of this project would be the computational requirements of the application given ASR and NER are all fairly heavy tasks by themselves. Runtime becomes a concern. For an automatic ASR task on a lecture, the runtime would usually be proportional to the duration of the lecture. That is usually acceptable for students. We can ask them to upload the recordings and ask them to wait for a few hours after getting the respective results that they needed since the application itself is not particularly time sensitive in anyways and real time inference is also not a requirement.

TEST AND POC SETS

For this POC project we tried to apply our methods on various datasets and tasks that we find interesting. We first did ASR on the CMU computer graphics lecture series and used the output generated as input for the summarization, knowledge graph and the LDA analysis. Along with the CMU lecture series, we also used Animal Farm as a Benchmark index to see how long our task takes and whether the system has any major flaws.

OBSERVATIONS

An interesting observation is that the summarization model is a CNN based model built on news dataset, during summarization tasks, some terms might trigger a response from the model to return a set of CNN related tokens and this deviates from the original context of the passage. This prompted us to think about the training source of the model, we initially assumed that the news would have provided a wide coverage for the model to gain an understanding on the context of most terms. On hindsight, perhaps a model trained on education related data (Such as secondary, high school speeches, although quite different, mimics the instructional nature of the speech used in university lectures)

Furthermore, due to the limitation of runtime, there are length limits to the audio file that we feed on the deployed version of the application. This is due to the runtime restriction that machine and online deployments can support. On a local GPU that we ran on, where the runtime and stability can be guaranteed, we do not run into this issue. Therefore, for those who would like to try out this application on a larger set of text and audio file, we recommend running this on a computer that is suitable for large scale inference

CHALLENGES AND FUTURE IMPROVEMENTS

TECHNICAL CHALLENGES

We endeavored to implement a deployed system predominantly situated on the web, a pursuit not devoid of its own technical hurdles and constraints. The endeavor has proven to be a technically demanding one, fraught with challenges encountered in integrating Spring Boot and navigating various hosting platforms. Moreover, our reliance on QWeb has introduced its own set of limitations, as outlined in our previous discussions regarding runtime capacity.

Simultaneously, our exploration has brought to light intriguing constraints stemming from the summarization model, notably concerning token lengths and the CNN backbone's training on news data. This revelation surfaced prominently during the rigorous testing phase of each section, underscoring potential issues and areas warranting further scrutiny.

FUTURE IMPROVEMENTS

The 2 important improvements that we would consider given enough time and resource would be first to consider the capacity and plan accordingly when designing the application. With a larger set of tasks, we would have to allocate more resources and runtime to support the operations and that is true for the web deployments as well. The second point would be on the training data that we would use on the summarization model (As we have learnt how intimately connected the summary model is trained on versus how it is behaving during a deployment situation), we would have probably chosen to retrain or use a pretrained model that is trained based on education related data. (Such as summary writing in high schools or perhaps other forms of educationally related tasks).

APPENDIX A

PROJECT GUIDELINES

DATE OF PROPOSAL

05/05/2024

PROJECT TITLE

ISS Intelligent Reasoning Systems project - Procrastinate: A student's best studying companion.

CLIENT

NUS ISS

25 Heng Mui Keng Terrarce, Singapore

BACKGROUND/ AIMS/ OBJECTIVES

Procrastinate aims to ease a student's workload by condensing study material, from speech to text, text to summary, topics extraction and text to mind map.

REQUIREMENTS OVERVIEW

- SQL database knowledge
- Amazon s3
- Programming on Springboot, Django and NextJs
- BART model from facebook (Text summarization)
- LDA model (Language processing)
- Knowledge based model (Knowledge graph formation)
- Whisper model (speech to text)
- Frontend and backend deployment

RESOURCE REQUIREMENTS

Hardware:

- CPU

Software:

- Springboot (Java, Maven)
- Django (Python)
- Python libraries
- NextJs (Node.Js HTML, CSS)

PROJECT GUIDELINES

NUMBER OF STUDENTS

1) Hou Lu Chiok Weh Alejandro - Main SWE, lead
2) Jane Lee - Frontend developer
3) Yang Yichang - ML model implementation and algo optimization

TEAM FORMATION & REGISTRATION

Team Name: Procrastinate

Project Title: Procrastinate

Team Member 1

Name: Hou Lu Chiok Weh Alejandro

Matriculation Number: A0230492L

Contact (Mobile/Email): 90685997 / e0696906@u.nus.edu

Team Member 2

Name: Yang ZiChang Michael

Matriculation Number: A0206039L

Contact (Mobile/Email): 97816496 / e0425962@u.nus.edu

Team Member 3

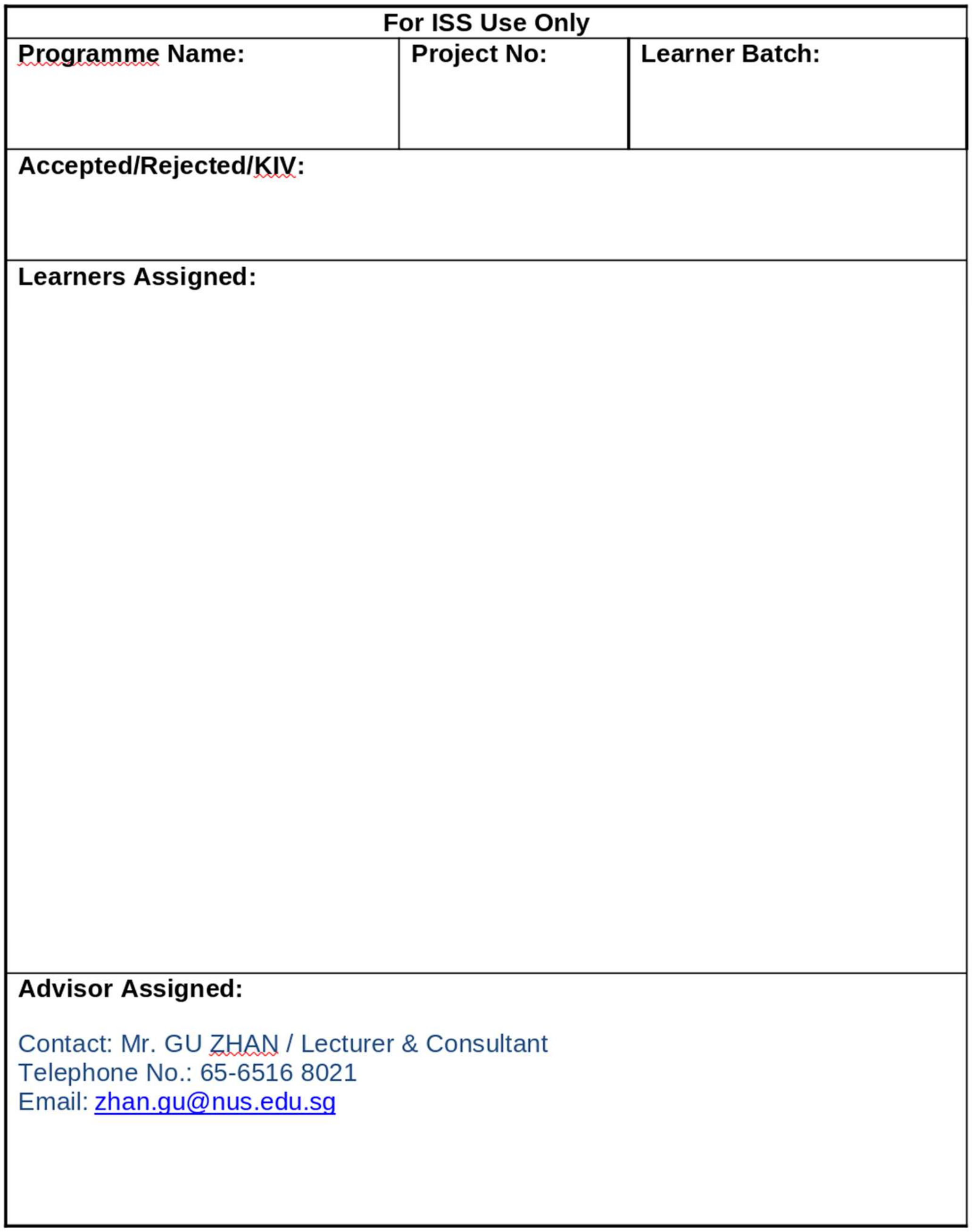
Name: Lee Jane

Matriculation Number: A0214865Y

Contact (Mobile/Email): 97712063 / e0533359@u.nus.edu

APPENDIX A

PROJECT GUIDELINES



For ISS Use Only		
Programme Name:	**Project No:**	**Learner Batch:**
Accepted/Rejected/KIV:		
Learners Assigned:		
Advisor Assigned: Contact: Mr. GU ZHAN / Lecturer & Consultant Telephone No.: 65-6516 8021 Email: zhan.gu@nus.edu.sg		

APPENDIX A

MAPPED SYSTEM FUNCTIONALITIES

Machine Reasoning

Knowledge extraction

The application takes in the user's upload document. With that doc it extracts relationships and processes them to form a network with appropriate properties.

Topic modelling

Documents from the user will also go through text preprocessing, then it will pass through an LDA model where it will create topics, get coherence scores, and according to the scores pick up the top subject from each topic. This is to gather the topics from the user's upload document to find out in general what the user is looking to learn.

Reasoning system

Knowledge representation

With the knowledge extracted, the app will them create a knowledge graph and represent to the user as a mind map.

Cognitive System

Speech to text function

The application calls upon Open AI's whisper model when user's input as a voice/sound file. Returning a full transcript to the user.

APPEN
DIX
B

APPENDIX C

INSTALLATION AND USER GUIDE

USING THE APPLICATION

There are 2 ways to use the procrastinate application. The simple and not so simple way.

The first way:

https://procrastinate-frontend.vercel.app/

Username: ale123

Password: sunny

That's it. Our application is live on the web so it's available to use anytime, anywhere (Or until our team runs out of money to host them)

RUNNING THE APPLICATION ON LOCALHOST

As our application is run by 1 frontend app and 2 microsoft, user will need to install 3 different environments

Frontend (Nextjs)

Requirement: Nodejs, Npm

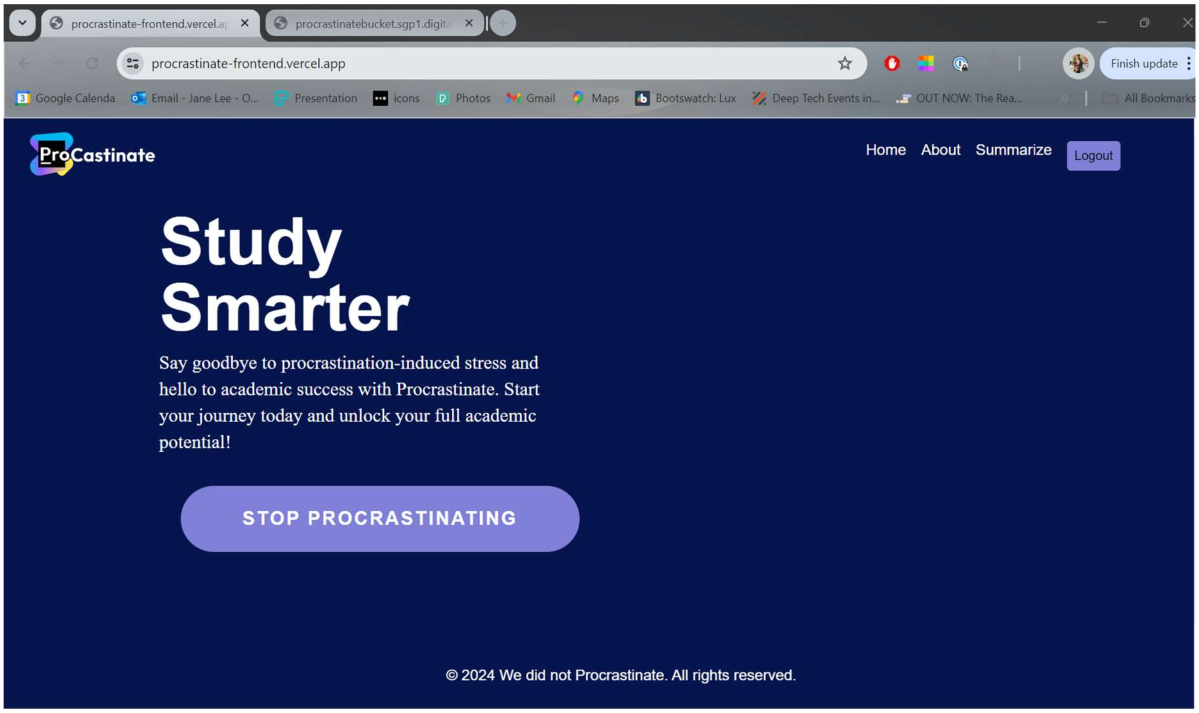
1) CD INTO PROCRASTINATE-FRONTEND
2) RUN NPM INSTALL
3) RUN NPM RUN DEV
4) GO TO LOCALHOST 3000

APPENDIX C

INSTALLATION AND USER GUIDE

UI WORKFLOW

1. Upon entering the website, users will first be directed to the landing page



Study Smarter

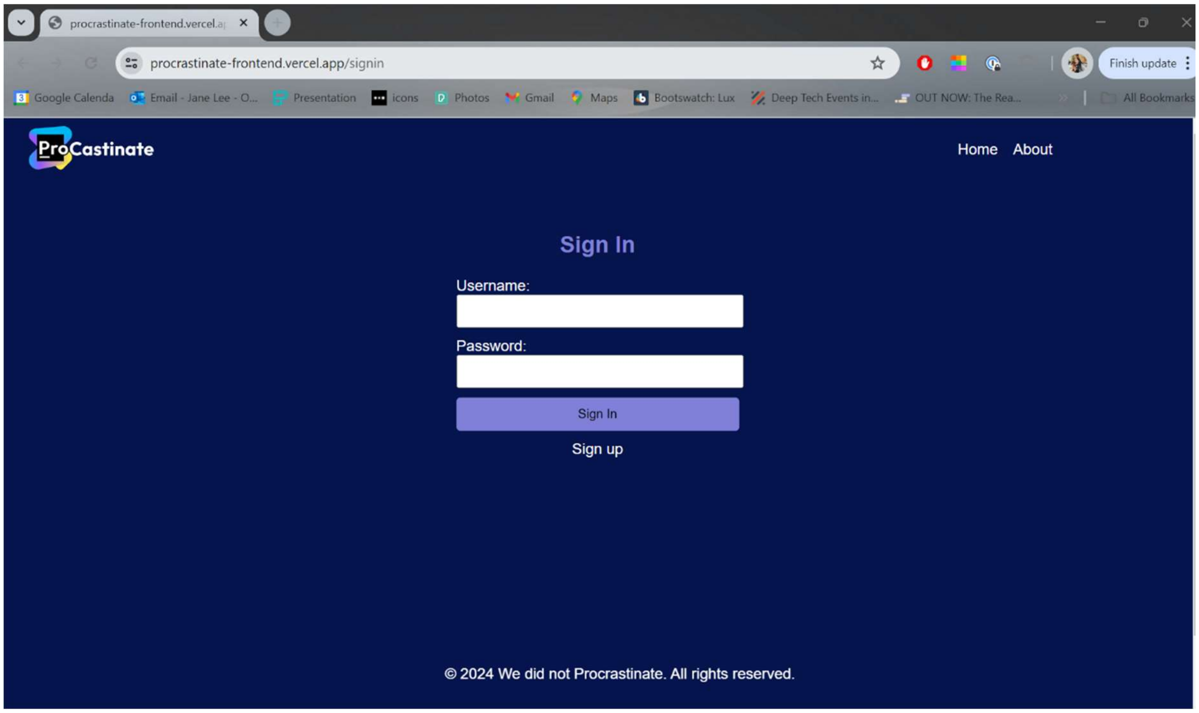
Say goodbye to procrastination-induced stress and hello to academic success with Procrastinate. Start your journey today and unlock your full academic potential!

© 2024 We did not Procrastinate. All rights reserved.

2. When the "Stop Procrastinating" button is selected, if user is not signed in,
   a. Users will be prompted to do so
   b. On the same page, users can also opt to sign up

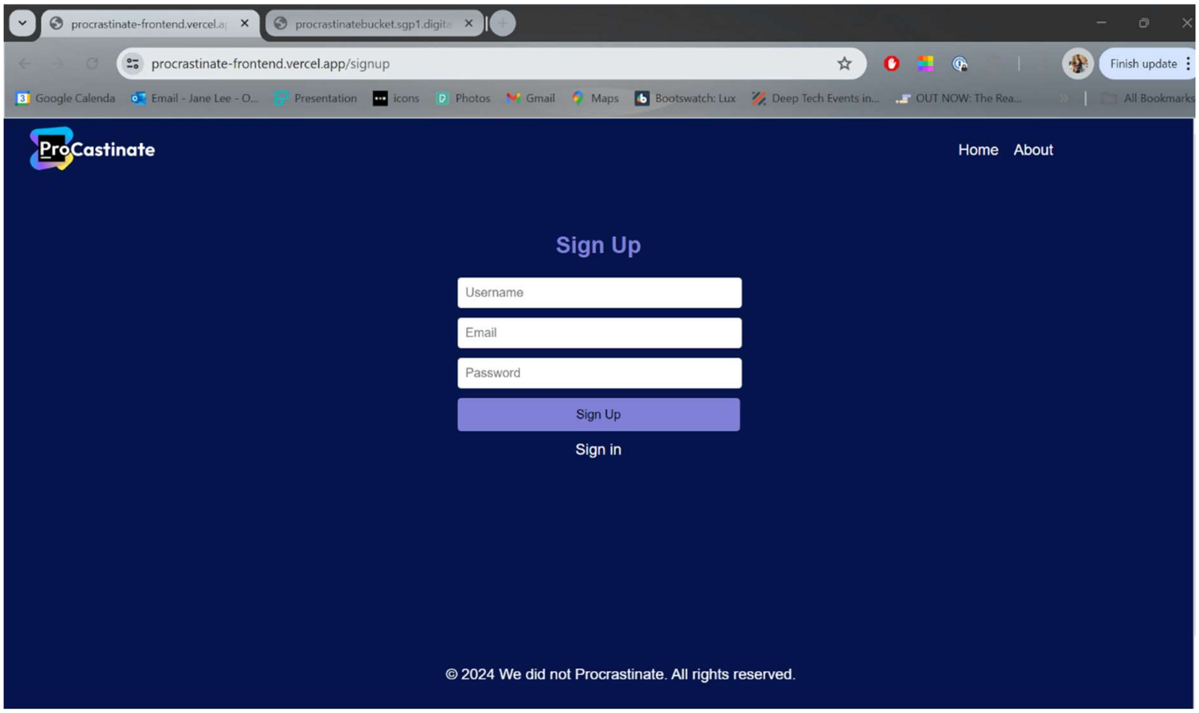
APPENDIX C

INSTALLATION AND USER GUIDE



Sign In

© 2024 We did not Procrastinate. All rights reserved.

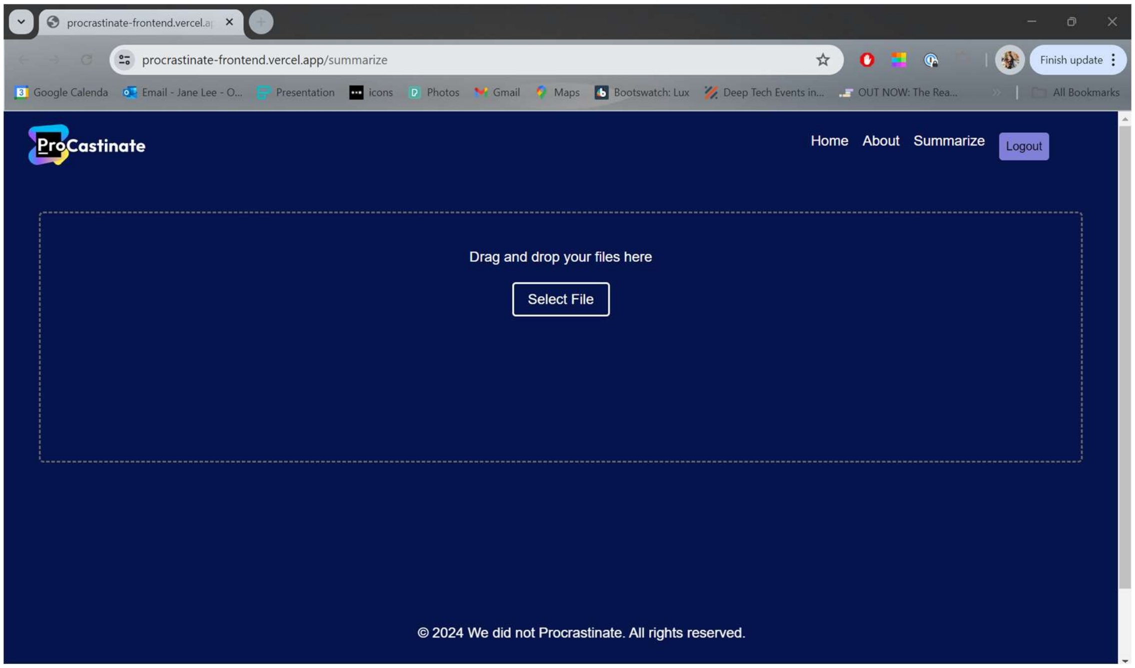


Sign Up

© 2024 We did not Procrastinate. All rights reserved.

INSTALLATION AND USER GUIDE

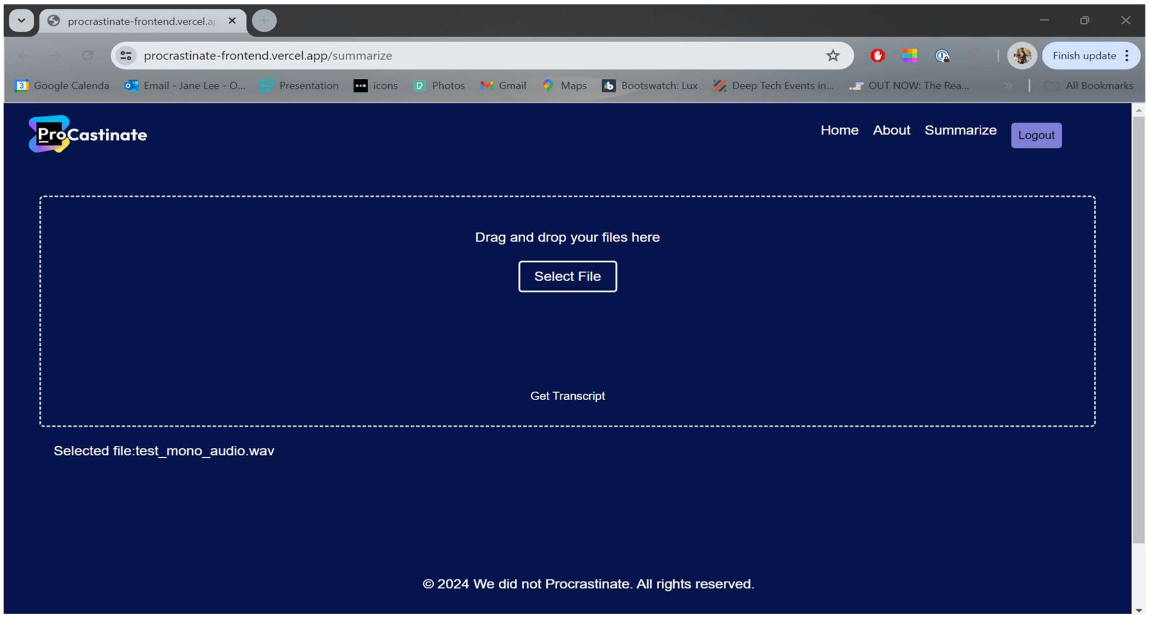
3. After the user signs in, they will be redirected to the summarise page. On this page, the user can upload audio or text files.



Drag and drop your files here

© 2024 We did not Procrastinate. All rights reserved.

a. If audio file is selected, the user will have the option to "Get transcript"

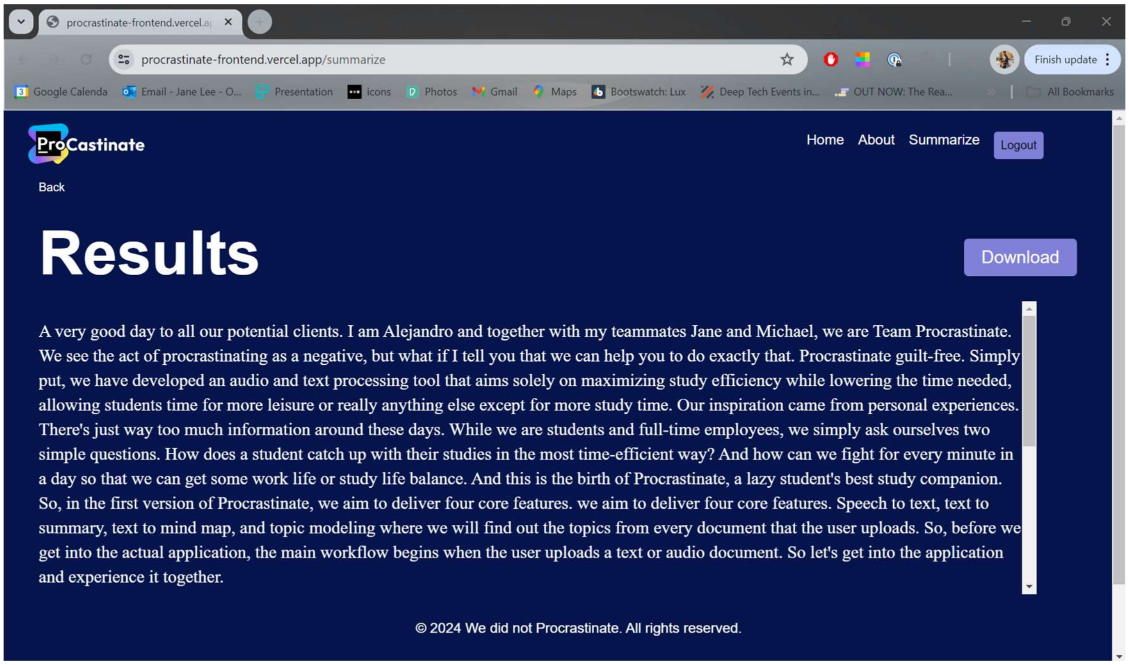


Drag and drop your files here

Selected file:test_mono_audio.wav

© 2024 We did not Procrastinate. All rights reserved.

b. After file is selected and the "Get transcript" button is select, the user will be able to view and download the transcribed text.

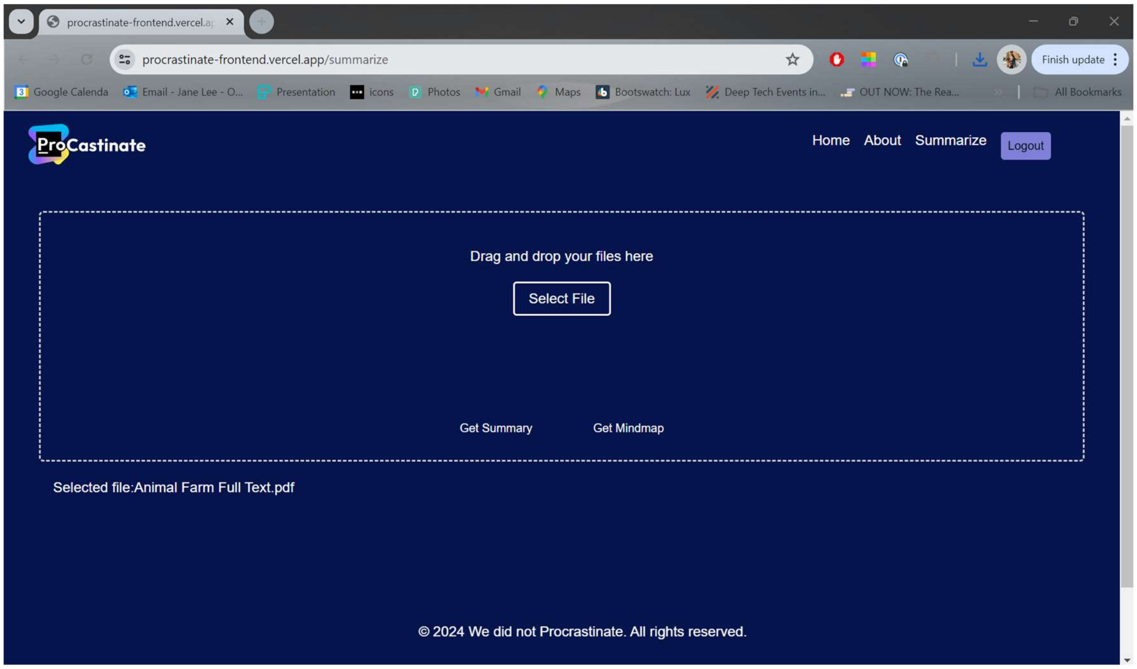


Results

A very good day to all our potential clients. I am Alejandro and together with my teammates Jane and Michael, we are Team Procrastinate. We see the act of procrastinating as a negative, but what if I tell you that we can help you to do exactly that. Procrastinate guilt-free. Simply put, we have developed an audio and text processing tool that aims solely on maximizing study efficiency while lowering the time needed, allowing students time for more leisure or really anything else except for more study time. Our inspiration came from personal experiences. There's just way too much information around these days. While we are students and full-time employees, we simply ask ourselves two simple questions. How does a student catch up with their studies in the most time-efficient way? And how can we fight for every minute in a day so that we can get some work life or study life balance. And this is the birth of Procrastinate, a lazy student's best study companion. So, in the first version of Procrastinate, we aim to deliver four core features. we aim to deliver four core features. Speech to text, text to summary, text to mind map, and topic modeling where we will find out the topics from every document that the user uploads. So, before we get into the actual application, the main workflow begins when the user uploads a text or audio document. So let's get into the application and experience it together.

© 2024 We did not Procrastinate. All rights reserved.

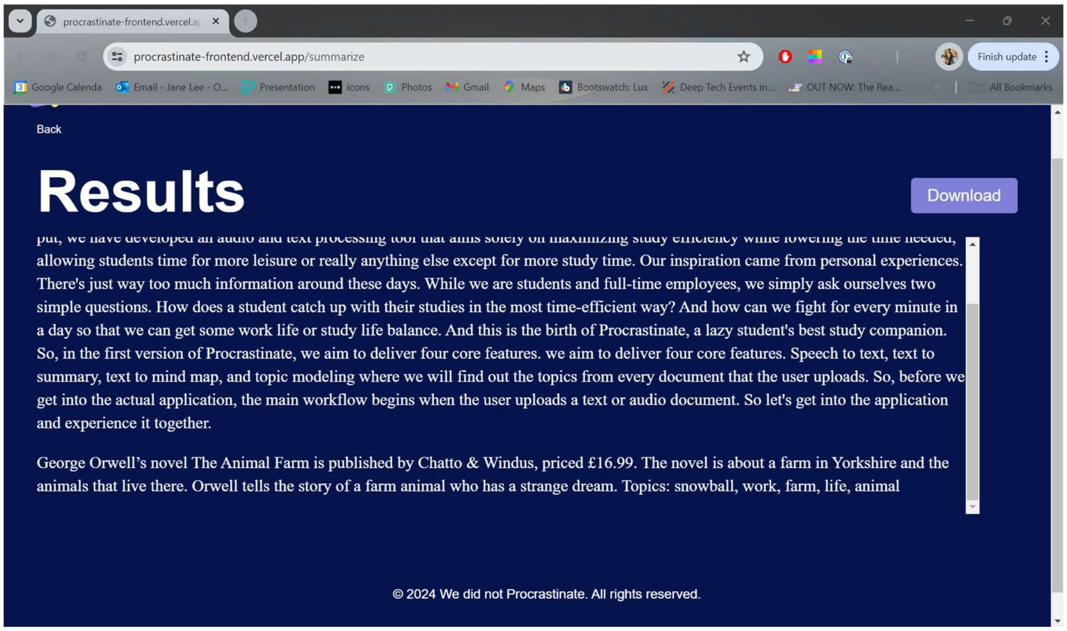
c. If a text file is selected, the user will be given the option to "Get Summary" or "Get Mindmap".



Selected file:Animal Farm Full Text.pdf

© 2024 We did not Procrastinate. All rights reserved.

i. If "Get Summary" is select, the user will be able to see the summarised text along with the topics



Results

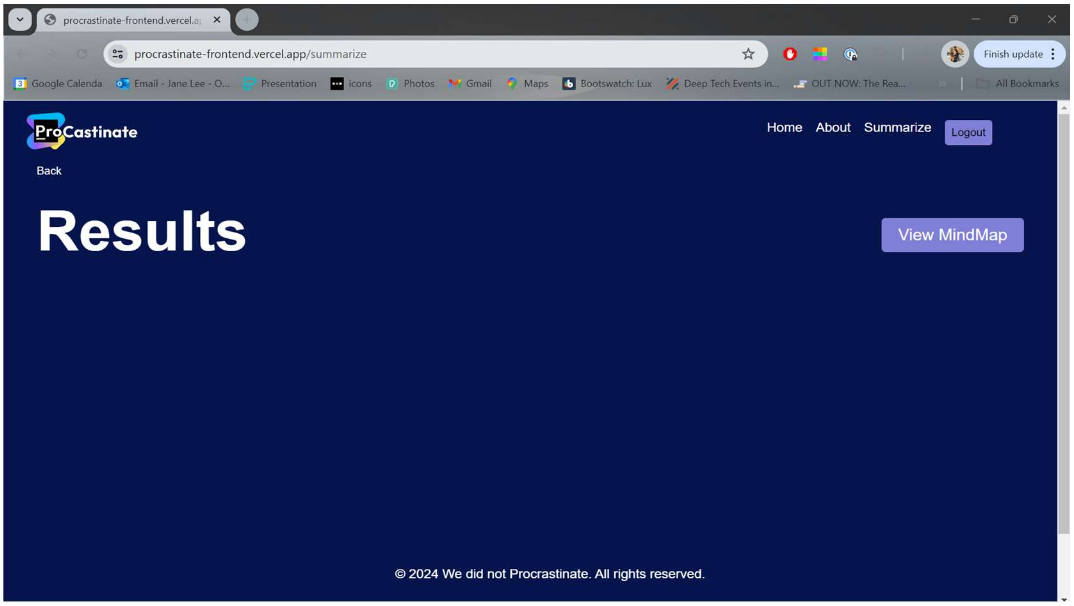
put, we have developed an audio and text processing tool that aims solely on maximizing study efficiency while lowering the time needed, allowing students time for more leisure or really anything else except for more study time. Our inspiration came from personal experiences. There's just way too much information around these days. While we are students and full-time employees, we simply ask ourselves two simple questions. How does a student catch up with their studies in the most time-efficient way? And how can we fight for every minute in a day so that we can get some work life or study life balance. And this is the birth of Procrastinate, a lazy student's best study companion. So, in the first version of Procrastinate, we aim to deliver four core features. we aim to deliver four core features. Speech to text, text to summary, text to mind map, and topic modeling where we will find out the topics from every document that the user uploads. So, before we get into the actual application, the main workflow begins when the user uploads a text or audio document. So let's get into the application and experience it together.

George Orwell's novel The Animal Farm is published by Chatto & Windus, priced £16.99. The novel is about a farm in Yorkshire and the animals that live there. Orwell tells the story of a farm animal who has a strange dream. Topics: snowball, work, farm, life, animal

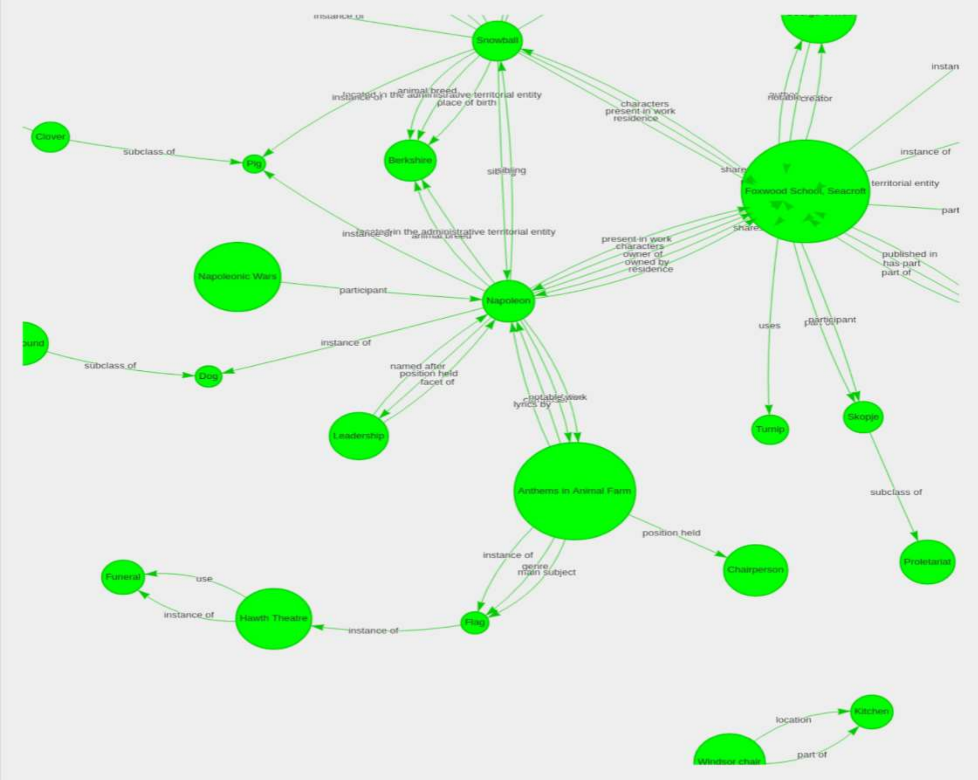
© 2024 We did not Procrastinate. All rights reserved.

ii. If "Get MindMap" is selected, the user will be given the option to "View MindMap" . This option will redirect the user to a separate page where the Mind Map will be displayed

INSTALLATION AND USER GUIDE



© 2024 We did not Procrastinate. All rights reserved.



APPEN
DIX
C

INSTALLATION AND USER GUIDE

USER SERVICES (SPRINGBOOT)

Requirement:

- *Maven*
- *Java SDK / JDK*

1) Set 3 environment vars
    a) SPRING_DATASOURCE_PASSWORD="sunnydays"
    b) SECRET_KEY="cZz/NH0k4SBxT28lmYg6sleWlhDIQzPew4DwNzW3dso"
    c) ACCESS_KEY="DO00HDHNTKHM9TFQFW6F"
2) Cd into procrastinate-backend
3) Run mvn spring-boot:run

DATA SERVICES (DJANGO)

Requirement:

Python 3.10 and above

Pip Install requirement.txt

1) Set 3 environment vars
    a) db_password="sunnydays"
    b) SECRET_KEY="cZz/NH0k4SBxT28lmYg6sleWlhDIQzPew4DwNzW3dso"
    c) ACCESS_KEY="DO00HDHNTKHM9TFQFW6F"
2) Cd into procrastinate_data_processor
3) Run python manage.py runserver

NOTE: Upload audio files MAX runtime <= 3mins

The UI is dynamic and simple. Therefore we trust our users to be able to cruise around our MVP with ease. In any case, do refer to the demo/marketing video for a complete walkthrough if needed.

APPENDIX C

INSTALLATION AND USER GUIDE

APPEN
DIX
C

INDIVIDUAL REPORTS

ALEJANDRO

INDIVIDUAL CONTRIBUTION

I was the tech lead for this project due to my idea being chosen and also the one with the most software engineering background. To start, I hosted weekly project standups and discussions that were completely needed to catch up on each member's work status, roadblocks and also new ideas if any. I also wrote the sprint document and assigned work to every member from the team.

I came up with the full system architecture, use case diagrams, sequence diagrams and entity diagrams for our relational database.

On the technical side, I worked on every part of the project. All 3 projects, Nextjs, Springboot and Django were set up by me. While the frontend, beyond initial set up and minor tests were mostly handled by Jane, I was fully working on both the Django and Springboot applications. On the Django side, Michael did most of the work on the ML models and algo to be used by the application while I integrated them (with some rewrites/enhancements for some models) into the application.

Application security using Spring security and JWT was also set up by me. While it was not the most pressing matter for this particular project, I felt that it was important to execute our project as close to real world implementation as possible.

I also created and deployed both our databases, MySQL and Amazon S3 bucket. All database related matters like connection to the application and maintenance were done by me.

Deployment of all 3 applications were also done by me after fully integrating all aspects of the application. The Nextjs application was hosted on Vercel. Springboot was deployed on Heroku and Django on a ngrok server.

ALEJANDRO

LEARNING OUTCOME

Most of what I learned from this project is from the technical side. I never knew implementing ML models needed to have so many considerations. The biggest thing is time.

This is because running a model without high-end GPU and CPU will take a very long time. So many things can go wrong the longer the processing time is. Network interruptions, system crashing and server timing out. Heroku has a 30 seconds limit while waiting for a response and that became a big problem during deployment. There was not enough time to fix that hence bandage type solution was used to overcome it, but in the future when I am planning a project that has AI or ML elements, I will definitely look into this first thing during system design.

Also, python applications that have ML come with a lot of libraries, causing the application to be very large in size. This means most cloud hosting services may not be able to deploy it. This happened to our Django application. While normal web applications will never have this problem, this problem is very real for python projects with ML capabilities. If I had known this at the start, I would spin up a different Django/Flask project for every single model implementation so that I can easily host all of them on Heroku. This can even help with concurrency problems that are very hard to overcome even with multithreading if not given a powerful system/server. This would mean that all my applications can be on heroku or any cloud hosting service with minimum lag time and not risk being too large to be deployed.

Lastly, I think as a tech lead I shouldn’t assume my team’s ability in any aspect. For example, there were instances where git was misused maybe due to a lack of understanding, causing files to be missing, branches to be corrupted or wrongly created. What I should have done was to establish a proper workflow on git, maybe even write a script so that this kind of time consuming mistakes don’t happen. I also cannot assume that my team will want to work day and night for a certain project and should never plan my sprints with that assumption. It’s fair to not want to code and research during all free time and I should have given way less work in hindsight. My expectations were properly adjusted from this project and I know what I should or shouldn’t do for future projects.

APPLICATION OF KNOWLEDGE IN OTHER AREAS/WORKPLACE

I can definitely apply whatever I did in this project in my workplace. Most SWE jobs in 2024 will cross paths with ML/AL or intelligent systems in one way or another. This is especially true in my workplace where we deal with large amounts of data. The immediate use case I can

APPN
DIX
D

ALEJANDRO

think of is regarding dynamic thresholds, which is a big deal in my company. The only way to set dynamic thresholds is by machine learning, with all the data available in our database, it'll be a waste to not run it through a hybrid system to determine the appropriate threshold for a particular time.

Generally speaking, text processing is useful in all corporate settings. Whether it is meeting notes or document processing, having the knowledge to extract topics, relationships and sometimes summarization can be very useful. Yes we can call on the openAI api for this, but as a tech company, there's always a time where we have to do our own implementation. It could be security issues, it could be special cases where openAI do not have the knowledge on or very specific instances or highly abstract scenarios. Using a tool on the internet is nothing new, but being able to craft one that caters specifically to your needs is a mark of a good SWE.
Lastly, I can now say I have coded a full stack application using 3 different frameworks, 2 different databases, 2 different ORMs and 4 different languages.

APPEN DIX D

JANE

MAIN CONTRIBUTION

My primary contribution to the project centered on developing the user interface (UI) and seamlessly integrating it with the backend APIs. The UI, constructed using Next.js, embodies modern development practices and is enhanced with TypeScript, ensuring robustness and maintainability. TypeScript's benefits include static typing, which catches errors early in the development process, facilitating smoother collaboration and reducing debugging time. Throughout the development process, I gained valuable experience in managing data flow across different components, optimizing performance, and enhancing user experience. The codebase was meticulously organized, with a focus on modularity and maintainability. Each feature was compartmentalized into modular components, with distinct files for Pure UI (TSX) files, logic processing (TS) files, models and interfaces, styled components, and string management. This approach ensured clarity and ease of maintenance, allowing for efficient collaboration and scalability as the project evolved.

To further improve maintainability, API-related files were consolidated within a dedicated folder, encompassing logic files, API file paths, and models mapped to response and request bodies of the API calls. Every component that utilises the UI does so by calling functions from this centralised folder. Additionally, an authentication utils file was created to manage all authentication-related functions, streamlining authentication processes across the application.

INDIVIDUAL CONTRIBUTION

In my role, I spearheaded the design and implementation of the UI, focusing on creating an intuitive and visually appealing interface that aligns with user expectations and project requirements. This involved collaborating closely with the backend team to ensure seamless integration of frontend and backend functionalities, as well as conducting thorough testing to identify and address any potential issues.

LEARNING OUTCOME

One of the key learning outcomes from this project was mastering the intricacies of frontend development, particularly in the context of web applications. By working extensively with Next.js and TypeScript, I honed my skills in building responsive and interactive UIs while adhering to best practices and design principles. Additionally, I gained valuable insights into

JANE

the importance of effective communication and collaboration within cross-functional teams, as well as the significance of user-centric design in delivering successful software solutions.

Throughout this project, I underwent a transformative journey in frontend development, particularly within the dynamic realm of web applications. Embracing the Next.js framework, I immersed myself in crafting responsive, interactive user interfaces that transcend conventional boundaries. Next.js became my trusted companion, EMPOWERING me with its server-side rendering prowess and facilitating seamless integration with TypeScript for enhanced code quality and maintainability. Moreover, my proficiency in version control and collaborative development blossomed through GitHub, where I honed my skills in Git workflows and pull requests, fostering a culture of teamwork and continuous improvement.

Throughout the project, one notable challenge revolved around the passing and management of data across different components. Despite the initial difficulty posed by the modularization of each component, which made debugging data management more complex, this obstacle ultimately became a significant learning opportunity. Wrestling with data flow intricacies provided invaluable insights into frontend data management. By navigating through these challenges, I developed a profound understanding of effectively managing data within frontend applications. This hands-on experience not only sharpened my problem-solving skills but also deepened my appreciation for the nuances of frontend development.

In addition to frontend development skills, this project provided an invaluable opportunity to explore and apply concepts from machine reasoning, reasoning systems, and cognitive systems. Through the integration of these advanced technologies, I gained insights into their practical applications and their potential to enhance user experiences in educational settings.

Overall, the project's exploration of machine reasoning, reasoning systems, and cognitive systems underscored the potential of these technologies to revolutionize educational tools and platforms. By harnessing machine learning algorithms, knowledge extraction techniques, and natural language processing capabilities, the project

JANE

APPLICATION OF KNOWLEDGE IN OTHER AREAS/WORKPLACE

The knowledge and experience gained from this project have proven invaluable in my professional development and have equipped me with a versatile skill set that I can apply across various domains and industries. Whether it's developing custom web applications, optimizing existing UIs for better performance, or collaborating with backend developers to deliver integrated solutions, my proficiency in frontend development and UI design has opened up opportunities for me to contribute meaningfully in different roles and projects.

One significant takeaway from debugging the UI is the holistic understanding it provides of system architecture and functionality. Through diagnosing and resolving UI-related issues, I gained insights into how different components interact within the system, as well as the flow of data throughout the application. This broader perspective has been instrumental in tackling complex challenges and making informed decisions in diverse technical environments.

Moreover, the problem-solving skills honed through debugging have proven transferable to various scenarios outside the realm of UI development. Whether it's troubleshooting software bugs, optimizing system performance, or devising innovative solutions to technical challenges, the analytical reasoning and critical thinking cultivated during this project have empowered me to navigate complex problems with confidence and efficiency.

Furthermore, the proactive mindset fostered through iterative debugging processes has influenced my approach to continuous improvement and innovation in the workplace. By consistently seeking out opportunities to identify and address inefficiencies, streamline processes, and enhance user experience, I have been able to drive positive change and contribute to the success of projects and initiatives across different teams and organizations.

In summary, the lessons learned from debugging the UI have transcended the confines of this project, shaping my professional growth and enabling me to make meaningful contributions in a wide range of technical roles and environments.

APPENDIX D

MICHAEL (YANG ZICHANG)

INDIVIDUAL CONTRIBUTION

I was chosen as the Machine Learning/AI subject matter expert for this project due to my exposure as an AI Engineer and Data Science graduate. While I am unfamiliar with the SWE side of the house (That is covered by Alejandro in the team), I contributed by choosing the right model for the various requirements that are needed by the project. Implementing the model logic for knowledge graph creation while understanding the characteristics of structures of lectures and the subtleties in knowledge entities in lectures.

I am also occasionally involved in debugging certain python codes and behaviours that my teammate works on. One of the more notable issues is the interchangeability between unix and windows paths. The solution I created involved calling the cwd command that taps into local absolute paths.

Lastly, I am also the host of the github project. This being the first time I am involved in a software related project (Most of my work I do involves R&D and local projects), I am involved in most of the code review and work processes.

MICHAEL (YANG ZICHANG)

LEARNING OUTCOME

One of the biggest takeaways from this project is to learn to integrate the knowledge I have as an AI Engineer with the Software Engineer and UI engineers in the team. Most of the time as an Research Engineer, my job involves getting to a very specific outcome that is required by the team, I may not have an upstream or downstream, therefore there isn't that much need and thoughts put into integrating software solutions (that is to say I can virtually write whatever I want and get away with it as long as the code at the end of the day works as intended by the user whenever they need it running). However, in this project, the integrated layer and sophisticated nature of deployment means that I will need to document my code and attempt to make it understandable by Alejandro who is integrating my work into the ML workflow software in our tech stack. Jupyter notebooks are useful for that as I am intimately familiar with writing well documented markdowns.

One of the difficulties that I learnt from while working on this project is the usage of git. Due to the large and sophisticated nature of the project, I need to learn to create more systematic commits and merges which Alejandro has set up for the team. I was able to learn some valuable lessons in managing a git project through the multiple commits, merges and branch management duties.

I think one of the things that Alejandro has told me that rings true for me is, no matter what the model or task is, do not just think about how the model output is going to look like and whether they make sense, also think about how they make sense in the larger context of the workflow. Ie: How is the person going to receive it (From a UI/UX perspective), how is the model output going to be served etc. Thinking about these bigger picture questions helps to scope the work and give a better target for me to shoot towards as I develop the models and algorithms in order to produce more usable data for my team members to serve and consume.

MICHAEL (YANG ZICHANG)

APPLICATION OF KNOWLEDGE IN OTHER AREAS/WORKPLACE

One of the largest applications of knowledge that I can see myself using in my workplace is the ability to work with Software engineers and thinking and planning on a larger scale beyond just my data skills. Having data skills and the maths foundation is great, but as an AI Engineer it is extremely easy and tempting to get lost amidst the large set of models and theories out there. It is important for us to get reminded of the need to get integrated work together with the software engineers in our midst so that we can work together to serve something. As my work inches closer towards deployment, learning to work in such a manner really helps us to serve the right product to our customers.

Crossing paths with a SWE and a UI/UX engineer helps me to put into perspective on how some of the work that ML does translates to directly usable products to the User.

Lastly, I'll say something about ChatGPT. I would not be surprised if by the time the project comes out one of the questions that many people ask would be "Why not just feed it to ChatGPT and let it decide what to return, it is quite good at this point." There's really a good point being brought up here, ChatGPT has been a really mature technology that many tech staff finds indisposable. The amount of data for the model to gain an intuitive understanding of language is beyond doubt. However, as an engineer, I believe there's value in being able to create pipelines and figure out I/Os and understand how the models behave as compared to relying on Chat. Furthermore in the environment where I work, offline deployments are a MUST. Knowing that these systems can be created and implemented in an offline environment (barring the cloud deployment) really helps to make the skill set extendable in my workplace.

APPENDIX D