



Dissertation on

“Automatic Generation of Tutorials from a text file”

Submitted in partial fulfilment of the requirements for the award of degree of

**Bachelor of Technology
in
Computer Science & Engineering**

UE17CS490A – Capstone Project Phase – 2

Submitted by:

Shrutiya M	PES1201700160
Tejaswini A	PES1201700740
Shubha M	PES1201701540
Kritika Kapoor	PES1201701868

Under the guidance of

Prof. N S Kumar
Computer Science Professor
PES University

January - May 2021

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING
PES UNIVERSITY**

(Established under Karnataka Act No. 16 of 2013)
100ft Ring Road, Bengaluru – 560 085, Karnataka, India



PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)
100ft Ring Road, Bengaluru – 560 085, Karnataka, India

FACULTY OF ENGINEERING

CERTIFICATE

This is to certify that the dissertation entitled

‘Automatic Generation of Tutorials from a text file’

is a bonafide work carried out by

**Shrutiya M
Tejaswini A
Shubha M
Kritika Kapoor**

**PES1201700160
PES1201700740
PES1201701540
PES1201701868**

in partial fulfilment for the completion of seventh semester Capstone Project Phase - 2 (UE17CS490B) in the Program of Study - Bachelor of Technology in Computer Science and Engineering under rules and regulations of PES University, Bengaluru during the period Jan. 2021 – May. 2021. It is certified that all corrections / suggestions indicated for internal assessment have been incorporated in the report. The dissertation has been approved as it satisfies the 8th semester academic requirements in respect of project work.

Signature
Prof. N S Kumar
Designation

Signature
Dr. Shylaja S S
Chairperson

Signature
Dr. B K Keshavan
Dean of Faculty

External Viva

Name of the Examiners

Signature with Date

1. _____

2. _____

DECLARATION

We hereby declare that the Capstone Project Phase - 2 entitled “**Automatic Generation of Tutorials from a text file**” has been carried out by us under the guidance of Prof. N S Kumar, Professor and submitted in partial fulfilment of the course requirements for the award of degree of **Bachelor of Technology in Computer Science and Engineering** of **PES University, Bengaluru** during the academic semester January – May 2021. The matter embodied in this report has not been submitted to any other university or institution for the award of any degree.

PES1201700160

Shrutiya M



PES1201700740

Tejaswini A



PES1201701540

Shubha M



PES1201701868

Kritika Kapoor



ACKNOWLEDGEMENT

I would like to express my gratitude to Prof. N S Kumar, Department of Computer Science and Engineering, PES University, for his continuous guidance, assistance, and encouragement throughout the development of this UE17CS490B - Capstone Project Phase – 2.

I am grateful to the project coordinators, Prof. Silviya Nancy J and Prof. Sunitha R, for organizing, managing, and helping with the entire process.

I take this opportunity to thank Dr. Shylaja S S, Chairperson, Department of Computer Science and Engineering, PES University, for all the knowledge and support I have received from the department. I would like to thank Dr. B.K. Keshavan, Dean of Faculty, PES University for his help.

I am deeply grateful to Dr. M. R. Doreswamy, Chancellor, PES University, Prof. Jawahar Doreswamy, Pro Chancellor – PES University, Dr. Suryaprasad J, Vice-Chancellor, PES University for providing to me various opportunities and enlightenment every step of the way. Finally, this project could not have been completed without the continual support and encouragement I have received from my family and friends.

ABSTRACT

In the present scenario, it has become a major necessity to create easy-to-access, easily-understandable and less time-consuming resources/tutorials. Our application proposes to create a full-fledged tutorial, provided an input in the form of a text file (txt/pdf). We have adopted a generator-subscriber model with role specific duties. The generator is provided with the option to generate tutorials and the subscriber can view/access the tutorials.

The input provided by the generator is analysed to identify headings, subheadings and paragraphs as a hierarchy. Additionally, the uploaded material is summarised and a concise presentation with audio voiceover, to enhance comprehension of the user, is presented for quick learning.

Furthermore, to ensure user interaction, assessments in the form of multiple-choice questions are dynamically created from the uploaded material. The scores of the assessment are instantly provided and the subscriber's performance is recorded to show progress. The application is further extended to handle multimedia such as images in the input text files and mathematical formulae. In order to increase the usability of the application and make it accessible to a wider population, the application is made language agnostic. The features of language agnosticism will be majorly concentrated on Indian languages, Kannada and Hindi.

A complete user-friendly web interface with a catalogue of automatically generated tutorials is provided for easy use of all the features and hassle-free learning.

TABLE OF CONTENTS

Chapter No.	Title	Page No.
1.	INTRODUCTION	01
2.	PROBLEM DEFINITION	03
3.	LITERATURE SURVEY	06
	3.1 Tutorial generation from related materials	
	3.1.1 Automatic Generation of Tutorial Systems from Development Specification	
	3.1.2 Automatic Generation of Contents Models for Digital Learning Materials	
	3.2 Study material generation from Text files	
	3.2.1 Effective Classroom Presentation Generation Using Text Summarization	
	3.2.2 A Semantic Machine Learning Approach to Automatic PPT Generation	
	3.2.3 Enhancing Automatic PPT Generation Technique through NLP for Textual Data	
	3.2.4 SlidesGen: Automatic Generation of Presentation Slides for a Technical Paper Using Summarization	
	3.3 Question Answer Generation	
	3.3.1 Enhanced Automatic Question Creator – EAQC: Concept, Development and Evaluation of an Automatic Test Item Creation Tool to Foster Modern e-Education	
	3.4 Multimedia Handling	
	3.4.1 Video Generation from Text	
	3.4.2 TiVGAN: Text to Image to Video Generation With Step-by-Step Evolutionary Generator	
	3.5 Language Agnosticism	
	3.5.1 A Study on Abstractive Summarization Techniques	

4.	PRODUCT SURVEY	12
	4.1 Text to Video Products	
	4.1.1 Raw Shorts	
	4.1.2 Viomatic	
	4.1.3 Lumen5	
	4.2 Text to PPT Products	
	4.2.1 Google Slides	
	4.2.2 Convertio	
	4.3 Assessment Generation Products	
	4.3.1 Typeform	
	4.3.2 Intel Smart Question Generation using NLP	
	4.3.3 QuestGen AI	
	4.4 Language Agnosticism Products	
	4.4.1 INLTK	
	4.4.2 Indic NLP Library	
	4.4.3 Hindi Text Summarisation	
	4.5 Multimedia Handling	
	4.5.1 PyMuPDF	
	4.5.2 Text2img API	
5.	PROJECT REQUIREMENTS SPECIFICATION	15
6.	HIGH LEVEL DESIGN	21
7.	IMPLEMENTATION	28
8.	INSTALLATION GUIDE	33
9.	USER MANUAL	35
10.	USER STATISTICS	39
11.	CONCLUSION AND FUTURE WORK	43
	REFERENCES/BIBLIOGRAPHY	46

LIST OF TABLES

Table No.	Title	Page No.
5.1	Software Requirements Table	19
6.1	ER Table	25
10.1	Feedback Comments Table	42

LIST OF FIGURES

Figure No.	Title	Page No.
6.1	Use Case Diagram	22
6.2	Master Class Diagram	23
6.3	Component Diagram	24
6.4	ER Diagram	24
6.5	Activity Class Diagram	26
9.1	Home Screen	37
9.2	Upload Document Interface	37
9.3	View Assessment Interface	38
9.4	List of Existing Tutorials Interface	38
9.5	Tutorial Interface	39
9.6	Take-up Assessment Interface	39
10.1	Average Feedback Metrics	40
10.2	Subscriber distribution	41
10.3	Average Scores of Pre-Generated Tutorials	41

CHAPTER-1

INTRODUCTION

A tutorial system is essential for a wide variety of user groups majorly involving teachers-students, mentors-trainees and many others. The documents and information available to the users is in huge volume, thus making it impossible to create a handwritten summary or an easy to interpret tutorial within a short time period.

The application built enables automatic generation of tutorials given an input text file, overcoming the problem of information overload and manual work that needs to be done otherwise. This helps the end user to easily understand the basic concepts of the topic clearly. End users can actively interact with the application, navigating from one topic to another, through a well-built web interface unlike the traditional trying to listen and grasp concepts in an otherwise time-consuming verbal comprehension.

The web application supports two roles for the end users. The 'generators' upload the documents onto the application and are responsible for tutorial generation. The 'subscribers' access the tutorials created and attempt the assessments based on topics read. The tutorial created can be accessed on the website with easy navigation tools or downloaded as a concise presentation with a voice over. Attempts have been made to create a concise presentation from a text file which concentrates on summarisation of the text through NLP techniques. This was followed by a subtopic generation phase to create a well-structured tutorial.

In order to generate the assessments for every subtopic of the tutorial, QuestGen.ai module was used to provide most appropriate objective type questions. The application was also scaled to handle multimedia content to make the tutorial more presentable. Language agnosticism was attempted by extending the application to handle Kannada and Hindi text files.

The application built proves to be a user-friendly platform for generators to make their content reach a large number of users and the subscribers to master new skills by referring to a well-structured easy to understand tutorial.

CHAPTER-2

PROBLEM DEFINITION

Quality teaching and student comprehension has always been an intensive yet important procedure, which is constantly being improved upon using many comprehensive and creative ideas. This is done to create an immersive learning experience for the learner, enhancing the understandability of the topic being highlighted upon. In recent times, the traditional teaching has been replaced by the use of technology aided tools like demos, presentations, etc to provide an interactive learning experience increasing the comprehensiveness of the attentive listener. These tools are everywhere around us, not only in the traditional teacher-student scenario, but also for a company to explain the timeline to its employees, for the explanation of a user manual for a software and many more.

In order to learn a new topic, a tutorial system is the most apt method that can be incorporated. A tutorial system makes the understandability of the topic easier than a normal verbal conversation. It's an aided tool towards direction and presentation of the content in an efficient manner, aiming for one goal - accurate audience grasp of the concept. For this very purpose, many tutorial generation tools were launched with the help of technology, where the presenter can customize a tutorial, he/she plans to put forward which can be viewed by the respective audience, like raw shorts and viomatic. As an improvement to this, online platforms like TutorialPoint and W3Schools provide full-fledged tutorials on already existing, popular and interesting topics, which a user can browse through to get a grasp. A common limitation of all these tools, is that they all present a tutorial on pre-generated content with no element of automation.

This is the problem we aim to solve through our research for this project is to automatically generate a full-fledged tutorial given the only input of a text file, acceptable in txt or pdf format. The use domain of the application will be massive, stretching over many areas. Split into two roles, a tutorial generator and a tutorial subscriber, anyone can generate a tutorial pertaining to their own topic at a

short notice because the entire procedure is automated. A subscriber, after subscribing to a tutorial, can view a navigable tutorial. The subscriber can access a concise presentation with a detailed voice over.

The content of the tutorial is presented with the text split into subtopics with easy navigation from one subtopic to the next. To evaluate learner comprehension and completion, assessments are provided at every step, again automatically generated on the text of the tutorial, and they are made mandatory so that the learner's progress could be tracked and his/her understanding of the concept could be evaluated. The tutorial generator is alerted with the scores of the subscriber, so that the progress of the audience could be monitored, also acting as a feedback tool, providing input to enhance the quality of the tutorial. Extending this idea, all featured and popular tutorials are displayed on the dashboard of the learner to pique interest, and can be viewed by the learner.

In this project, we have created an application that achieves all the purposes mentioned above, aiming at an interactive presentation of a new topic via a tutorial consisting of multiple features which is fully automatic. A user friendly, hassle free web interface available to everyone, is the output of this project aiming at an audience who wishes to learn any new topic. This application involves many ideas like automatic subtopic segregation via analysis of document structure, employment of NLP techniques to generate summary of the input document, tools to create presentations and automatic assessment generation tools.

Furthermore, the application processes multimedia input involving images and improves the understanding of the concept for the subscriber.

In order to provide a wider range of input ,language agnosticism has been attempted through extending the application for Kannada and Hindi languages.NLP techniques have been used to process the input files and generate an appropriate navigable tutorial.

We have explored different methods to create this application, providing a compilation of the most efficient techniques to create a fault tolerant, entirely automatic, tutorial generation website.

CHAPTER-3

LITERATURE SURVEY

Our project involves several components such as Text summarisation, PPT generation, Video tutorial generation, inherent Q/A system and progress tracker. There have been separate attempts at building some of the components in various manners. Several papers have been published pertaining to this.

3.1 Tutorial generation from related materials

This section details the survey of tutorial generation from related materials such as diagrams and learning materials.

3.1.1 Automatic Generation of Tutorial Systems from Development Specification [1]

In this paper, automatic tutorial generation was targeted at software as a step-by-step demo on how to use a particular software. The input to this system was use case diagrams, sequence diagrams and test data provided by developers and the generated tutorial was directly woven into the software. The system architecture consisted of five main steps - Adding extra description to sequence diagrams, Extracting function name from use case diagrams, Extracting operation information from sequence diagrams, Generating tutorial system automatically and Weaving tutorial system into target software automatically. It had an interactive GUI where features like showing Software Functions and Operations, showing examples of Software Usage, demonstrating Input Texts etc were highlighted for easy navigation of the software for the users.

This was an intuitive method at tutorial generation, but it has a scope limiting only to software. Furthermore, the sequence diagrams have to be annotated by the developers which limits the degree of automation of tutorial generation.

3.1.2 Automatic Generation of Contents Models for Digital Learning Materials [2]

In this paper, a method to automatically generate a contents model by analysing learning materials with the aim of supporting the construction of knowledge structures is proposed.

Learning materials are text-mined and analysed to generate a co-occurrence graph. The co-occurrence graph is then modelled to generate an organized content graph. To reduce the complexity of the graph to the relevant key points, an optimal spanning tree that only selects the strongest relations between nodes is searched. The relations between nodes are expressed through weighted edges.

The limitations include that the generation of the graph is dependent on the input text and therefore may create off topic nodes from examples. The detailed level of the output contents-model requires supervision from the domain expert.

3.2 Presentation Generation from Text files

This section details the survey of PPT generation from a text file which also include attempts at text summarisation.

3.2.1 Effective Classroom Presentation Generation Using Text Summarization [3]

In this paper, classroom presentation generation was aimed at using text summarization. Single text file taken as input and pre-processed. Sentences were scored using Cue-Phrase Method, Word frequencies, Title Method and Location Methods and the highest rated sentences formed the summary. Summary was presented as a PPT with 3 sentences per slide.

More Importance was given to summarising the entire text rather than the PPT creation and the drawbacks being that it does not correctly divide on sub-topics as such.

3.2.2 A Semantic Machine Learning Approach to Automatic PPT Generation [4]

In this paper, Machine learning was employed to automate PPT generation. Input of multiple documents of type pdf, text and MS word was taken and a PPT was generated. First, Input was

concatenated into a single string and pre-processed. Then, Feature Extraction (Title Feature, Sentence Length, proper Noun, Numerical Data, Term Weight) was done and values were stored.

The tree containing the sentences was traversed in preorder and the sentences were classified using Random Forest. The result generated was a concise presentation of slides containing indices like Abstract, Motivation, Problem Statement, Objectives, introduction, Summary and Key words only.

The resulting presentation contains pre-decided indices which fixes the structure of the PPT. Contiguous sub topic wise PPT was not formed, which is the main aim of a tutorial.

3.2.3 Enhancing Automatic PPT Generation Technique through NLP for Textual Data [5]

In this paper, NLP was the main idea behind PPT generation. Input of any format (PDF/DOC) file was given and output was a PPT. Input was concatenated into single string and preprocessed. Feature Extraction (Title Feature, Sentence Length, proper Noun, Numerical Data, Term Weight) was done and values stored. Data was then passed to the fuzzifier, which grouped the sentence into categories of importance according to the numerical score calculated. Fuzzy IF THEN Rules were then performed, on fluffy sets portrayed by proper enrolment capacities. It then utilized distinctive elements to make PPT on the basis of the analysis performed. MRR was used as a measuring strategy.

All the text isn't very conveniently segregated into headings. Key-expressions and sentence mapping is not performed per slide.

3.2.4 SlidesGen: Automatic Generation of Presentation Slides for a Technical Paper Using Summarization [6]

In this paper, Input was documents in LATEX form. These documents were converted to XML format which was parsed and information was extracted. Sections were categorized and then key phrases were extracted for some sections of the paper following which details were saved in a file (configuration file). Extraction of key phrases was used in summarization of the sections. To generate slides QueSTS (query specific extractive Summarizer) was used. Input was text which is in the form of an Integrated

Graph (IG) i.e. nodes representing sentences, linked to each other via weighted edges if the sentences were similar. Based on a particular query, nodes were generated as output, which bear greatest relevance to the query, calculated using the node weight which was further used to generate a coherent and readable summary. The XML file and the configuration file were given as input to the Slides Generator component and it generated slides for each section and subsection.

Natural language processing techniques are not used in this paper to compress the extracted sentences and to identify appropriate indentation structure for them. PPT produced as output is less appealing, containing only 4 slides only. The domain is restricted as it is research-paper specific.

3.3 Question Answer Generation

This section details the survey of the current state-of-the-art methods for question answer generation.

3.3.1 Enhanced Automatic Question Creator – EAQC: Concept, Development and Evaluation of an Automatic Test Item Creation Tool to Foster Modern e-Education [7]

In this paper, most important concepts out of textual learning content were extracted and single choice, multiple-choice, completion exercises and open-ended questions on the basis of these concepts were created. The system had three main modules:

1. **The Pre-processing module** – files of several formats were text cleaned, language detected and was transformed involving internal XML
2. **The Concept Extraction module** - ran term weighting extraction, statistical, semantic and structural analysis of the most suitable phrases was achieved
3. **The Assessment Creation module** - the suitable sentence for each phrase, also identified distractors and antonyms, created question items and reference answers were determined

This approach is a Question Answer generator, and does not embody tutorial generation as such. This could be used to guarantee interaction from the user, as an extra feature of our project.

3.4 Multimedia Handling

This section details the survey of multimedia generation from text.

3.4.1 Video Generation from Text [19]

In this paper, Video was generated using a hybrid framework employing a VAE and a GAN. The static features, called “gist” are used to sketch text-conditioned background color and object layout structure (Universal features). Dynamic features are considered by transforming input text into an image filter (Details). However, the video generated is generated for short sentences. The video generated is short with not much descriptive differences between the frames.

3.4.2 TiVGAN: Text to Image to Video Generation With Step-by-Step Evolutionary Generator [20]

This paper uses Generative Adversarial Network. Rather than first finding a mapping function between the text and all video frames, the network is trained with respect to one image and is gradually extended to longer frames. It has two stages: Text-to-Image Generation and Evolutionary Generation. In each step m , $2m$ images can be obtained by iterative operations of recurrent unit, R and generator, G learned in the first step of test-to-image generation. At each step m , m th step-discriminator DS_m is newly added to discriminate the sequence of $2m$ frames along with the image discriminator $D1$. Several experiments were conducted on three diverse datasets: KTH Action, MUG, and Kinetics. The video generated is generated for short sentences. The video generated is short with not much descriptive differences between the frames.

3.5 Language Agnosticism

This section details the survey of how language agnosticism was achieved.

3.5.1 A Study on Abstractive Summarization Techniques in Indian Languages [18]

This paper provides an overview of abstractive text summarization for Indian languages. Abstractive summarization is predominantly classified into two types namely the structure based and semantic

based approaches. The structure-based approach involves summary generation in a predefined structure without losing any meaning. The semantic based approach involves a document input, a semantic representation and a natural language generation phase to arrive at the output. One of the approaches for abstractive summarization involves getting the extractive summary which is further used to get the important concepts from the source text. The pre-processing step consists of data chunking, part of speech tagging, word sense disambiguation, etc. The 4 approaches described in this paper are:

Summarizer extracts key phrases and post processing through refinement and rephrasing results in the abstractive summary.

Consists of a pre-processing module, categorization module and a final post processing module to extract the key phrases and further mapping, resulting in the summary.

Rich semantic graph (RSG) is constructed and topics are ranked based on relevance.

The linguistic triples were obtained in the document and an RSG is used to obtain the abstractive summary.

CHAPTER-4

PRODUCT SURVEY

There are some already existing products that are similar to some components of our project.

4.1 Text to Video Products

This section details the survey of the current products for creating videos from text.

4.1.1 Raw Shorts [8]

Created Animated Videos from Text using AI within seconds. It is more of an editor rather than a completely automated video generator. Presented a storyboard to populate. Used AI to recommend short images/videos for each storyboard. The recommendations were based on the keyword identified which wasn't very accurate. Provided audio for the text with the availability of several languages. However, it is more of an editor and no Indian languages are available.

4.1.2 Viomatic [9]

Created Video from content. It is more of an editor. It summarised the article and populated the slides and also gave an option to add images. Provided audio for the text with the availability of several languages.

However, no Indian regional languages are available. Also, it was not instant (upwards of 5 minutes depending on the content). Maximum length of video possible was 10 minutes (paid). Free version allows a maximum of 3 minutes with a maximum of 12 slides.

4.1.3 Lumen5 [10]

This tool was similar to Raw Shorts [Raw Shorts: Online Video Maker | Video Creation Software](#).

Here, the background image was generated based on relevance.

However, it is more of an editor and no Indian languages are available.

4.2. Text to PPT Products

This section details the survey of the current products for creating presentations from text.

4.2.1 Google Slides [11]

This tool populated text based on hierarchy into the different slides of a PPT.

However, it offers just direct conversion of text to PPT with no summarisation or Semantic analysis done. Also, addition of images, audio, etc are not provided automatically.

4.2.2 Convertio [12]

It took a .txt file input and provided a .ppt file output.

However, no summarisation or Semantic analysis is done. Automatic Multimedia inclusion is not supported.

4.3 Assessment Generation Products

This section details the survey of the current products for generating questions and assessments.

4.3.1 Typeform [13]

This tool created online tests. It provides a UI to copy paste questions into a template on sharing, the student could answer the tests. A timer was provided. Post finishing the test, the results of the test were instantly generated to the user.

4.3.2 Smart Question Generation using Natural language processing [14]

Input can be provided in the form of direct text, text file or a test link. The best/important sentences which could be converted into questions were selected. These sentences were analysed and the most sensible question from them were provided.

4.3.3 Questgen AI [15]

It was an open source NLP library focused on developing easy to use Question generation algorithms. It used AI leveraging state-of-the-art transformer models like T5, BERT and OpenAI GPT-2 etc. It was able to create MCQs, Boolean Questions (Yes/No) and General FAQs.

4.4 Language Agnosticism Products

This section details the survey of the current products for Language agnosticism.

4.4.1 INLTK

It is a Natural language toolkit for around 15 Indian languages. Compatible tasks include tokenization, getting embedded vectors, predict next n words, language identification, removing foreign language sentences and so on. Can be used for transfer learning and paraphrasing.

4.4.2 Indic NLP Library [16]

It is used for advanced text processing tasks for Indian languages. Functionalities provided by this include Text Normalization, Script Information, Tokenization, Word Segmentation, Script Conversion, Romanization Indicization, Transliteration and Translation.

4.4.3 Hindi Text Summarisation

Summarizes the text document using numerical methods.

4.5 Multimedia Handling

4.5.1 PyMuPDF

Parses images from the input PDF file successfully.

4.5.2 Text2img API

Generates an image from given text.

CHAPTER-5

PROJECT REQUIREMENT SPECIFICATIONS

5.1 Introduction

Our project is an automatic tutorial generation from an input text file. This section aims to specify the high-level software and hardware requirements of the project. It also specifies the target audience. This is a complete requirements study with information about the project scope, software and hardware requirements, intended features and functionalities, constraints, assumptions and dependencies.

5.1.1 Project Scope

The scope of the project will cover the following:

1. Build a full-fledged tutorial from an input text file.
2. Provide users an intuitive and convenient GUI to maximise ease of use
3. Developing the end product to include the following features:
 - a. creating a tutorial from a text file
 - b. generate questions from the given topic,
 - c. provide user personalization with progress tracker
 - d. classify based on subtopics
 - e. summarized presentation generation on the text file
 - f. make the application language-agnostic
 - g. Inclusion of multimedia
4. Testing the application
5. Deliver the product to the target audience which includes teachers, students or trainees in a company.

5.2 Product Perspective

1. Work done w.r.t automatic generation of tutorial from a custom text input has been very minimal.
2. The existing products do not involve an assessment module, which is our aim to ensure user-interaction.
3. No personalization or feedback module has been implemented in any of the existing tutorial applications like 'GeeksForGeeks' and 'TutorialsPoint'.
4. A few components in our proposed solution such as Text summarisation, PPT generation, Q/A system have been independently explored before.
5. Our product aims to solve the above challenges and include already developed sub components.

5.2.1 Product Features

1. Our application proposes to create a full-fledged tutorial given a text file (Textbook chapters, pdf documents, etc).
2. The text is analysed to identify the headings, subheadings and paragraphs as a hierarchy in a tree-based approach.
3. The category of the input document is utilized to scrape the web for images pertaining to the specified topic to enhance the dynamically generated user interface.
4. The tree-based hierarchy is mapped and a navigational tutorial is rendered to the user.
5. A PPT component is also provided with the summarised text as required after text analysis and consists of the major points representing the topic of concern.
6. Audio explanation using voiceover is used to enhance comprehension of the user.
7. In addition to this, assessments are provided after every two subtopics to ensure user interaction.

8. The scores of the assessments and viewing activity are taken into account to provide a personalized progress tracker enabling the user to view his errors and analyse them as required.
9. A complete user-friendly web interface is created for easy use of all the features and hassle-free learning and customization based on the user's preferences.

5.2.2 User Classes and Characteristics

1. Teachers and students in a school/university
2. Employees and trainees in a company
3. Salesmen to demonstrate the working of their product to the customers
4. Any kind of audience with an aim to learn something.

5.2.3 Operating Environment

Operating environment for automatic generation of tutorials is as listed below.

1. Operating system: Windows/Ubuntu/Mac.
2. Client/server system
3. Database: sqlite3 database
4. Backend server: Flask
5. Frontend server: ReactJS
6. Platform: python/javascript/npm

5.2.4 General Constraints, Assumptions and Dependencies

1. Constraints:
 - a. Acceptable input:
 - i. Only PDF format documents accepted.
 - ii. Proper formatting of the document is a prerequisite.
 - iii. Text should be clearly formatted into a well structured document.

-
- iv. Should be formatted into proper hierarchy of headings according to font size
 - v. Size of the text should be consistent.
 - vi. Can accept images in pdfs placed anywhere without captions
 - b. Non acceptable input:
 - i. Cannot handle Mathematical Formulae
 - ii. Cannot handle non-English documents.
 - iii. Cannot handle tables currently.
2. Assumption:
- a. A text file (.pdf or .txt) is provided by the user.
3. Dependencies:
- a. A robust system to handle the load of the application.
4. Limitations:
- a. Handling multimedia content in the input text file
 - b. Handling different symbols in math and source code files
 - c. Storage and scalability

5.3 Functional Requirements

- 1. Client-server architecture
- 2. SQLAlchemy (ORM) used for Database.
- 3. User authentication.
- 4. The type of input file is checked.

5.4 External Interface Requirements

5.4.1 User Interfaces

- 1. Front-end software: ReactJS, HTML, CSS, Bootstrap

2. Back-end software: Flask, SQLAlchemy database

5.4.2 Hardware Requirements

1. Windows/Linux/MAC operating system

5.4.3 Software Requirements

Name	Description
Python	<ol style="list-style-type: none">1. Version = 3.x2. Python is used because of familiarity and good support for libraries.3. Python libraries include: Python-pptx , NLTK , Spacy, Numpy,etc
Flask	<ol style="list-style-type: none">1. Version = 1.1.22. Flask is a python based backend
ReactJs	<ol style="list-style-type: none">1. Version = 16.14.02. Front-end framework supporting HTML, CSS, Javascript, Bootstrap3. Installation requires npm
SQLAlchemy	<ol style="list-style-type: none">1. Version = 2.4.42. ORM based database
Operating System	Works on Windows, Ubuntu, MAC
Browser	Supported on all major browsers

Table 5.1: Software Requirements

5.5 Non-Functional Requirements

5.5.1 Performance Requirements

1. The application gives the users an option to create a tutorial once they upload the text file.
2. The application renders all the components of the tutorial without loss of information.
3. The database is regularly updated and maintained as and when necessary.

5.5.2 Security Requirements

1. Only the REST API is allowed to connect to the databases.
2. The application doesn't allow users to access secure information.

5.6 Other Requirements

5.6.1 Software Quality Attributes

1. **Availability:** The application allows multiple users to access the same tutorial at once and create tutorials at the same time, without crashing.
2. **Correctness:** The application never allows anyone to access the tutorials that a person did not create or subscribe to.
3. **Maintainability:** Supports continuous integration to deploy features and bug fixes without downtime.
4. **Usability:** Easy to use interface so that application can be used without a tutorial on how to use it.

CHAPTER-6

HIGH LEVEL DESIGN

6.1 Introduction

This section aims to specify the high-level design of the project. It specifies the application flow, database architecture, application architecture and technology architecture of the system.

6.2 Design Considerations

6.2.1 Architecture Choices

Client-Server Architecture is suitable for our project. In client-server architecture, the load is distributed between the client and the server. The main reason for this architecture is, it involves all the data in a centralized system. The server component is requested for some functionalities and services by one or more clients.

6.3 Design Description

The modules used are:

1. Tutorial generation module
2. Assessment module
3. User progress module

6.3.1 Use Case Diagram

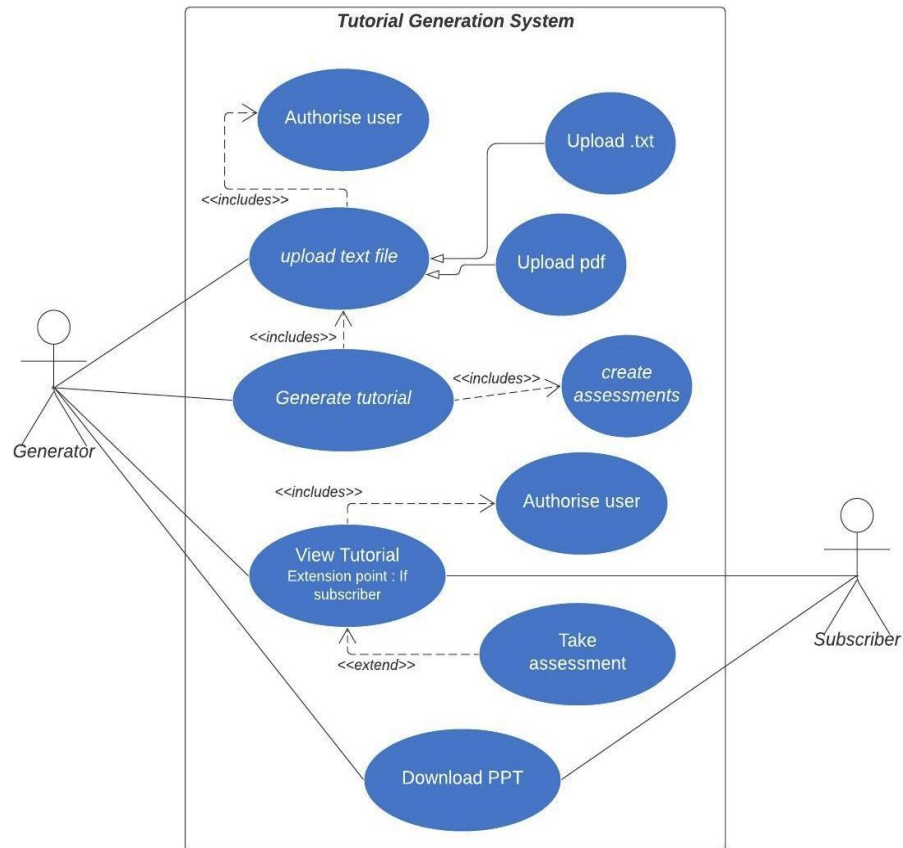


Figure 6.1: Use Case Diagram

6.3.2 Master Class Diagram

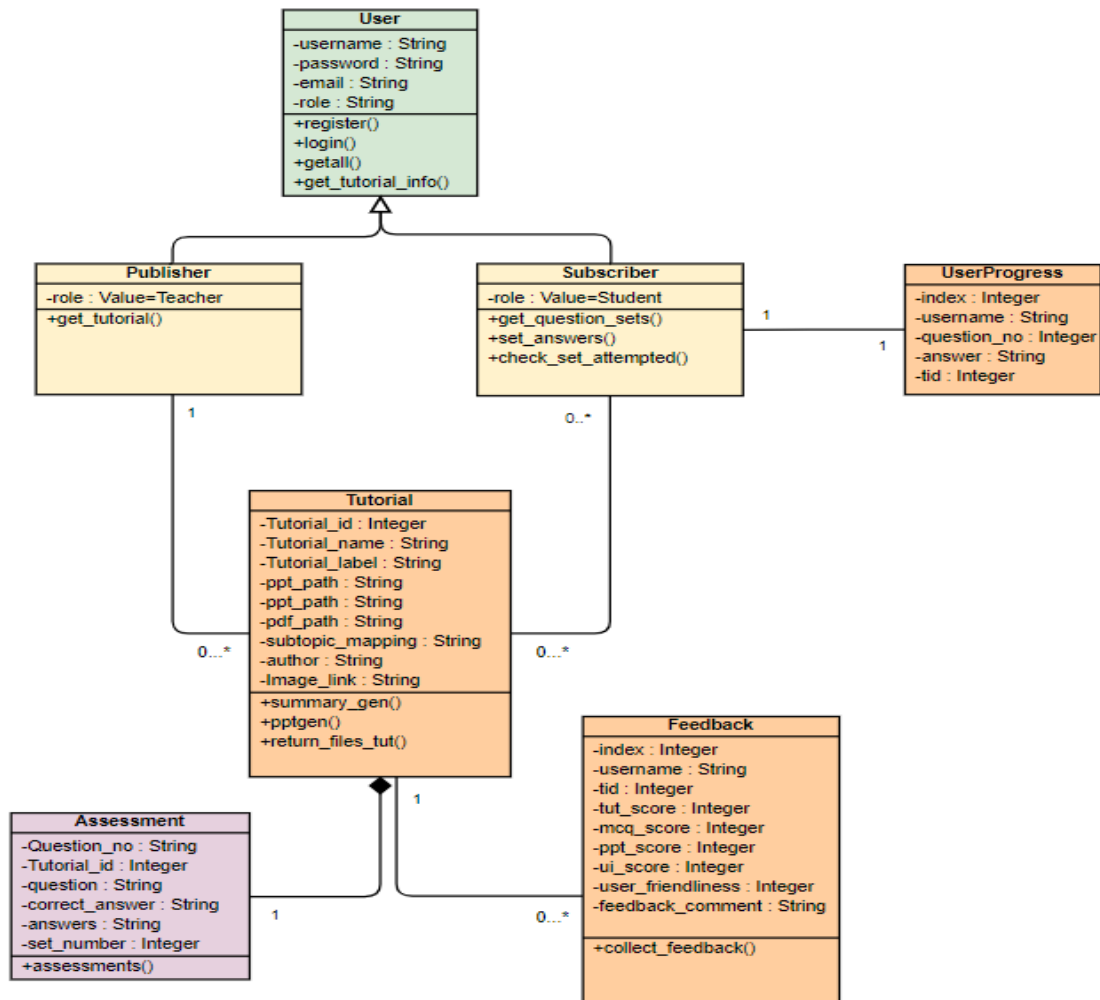


Figure 6.2: Master Class Diagram

6.3.3 Reusability Considerations

1. The tutorial generation module post creation by the creator is reused to provide access to the subscriber.
2. The assessment module post creation is reused to provide to the subscriber.

6.4 High Level System Design

High level representation of all the components of the project is specified in a concise way.

6.4.1 Component Diagram

Our application follows a component-based architecture as it can be easily reused and replaced.

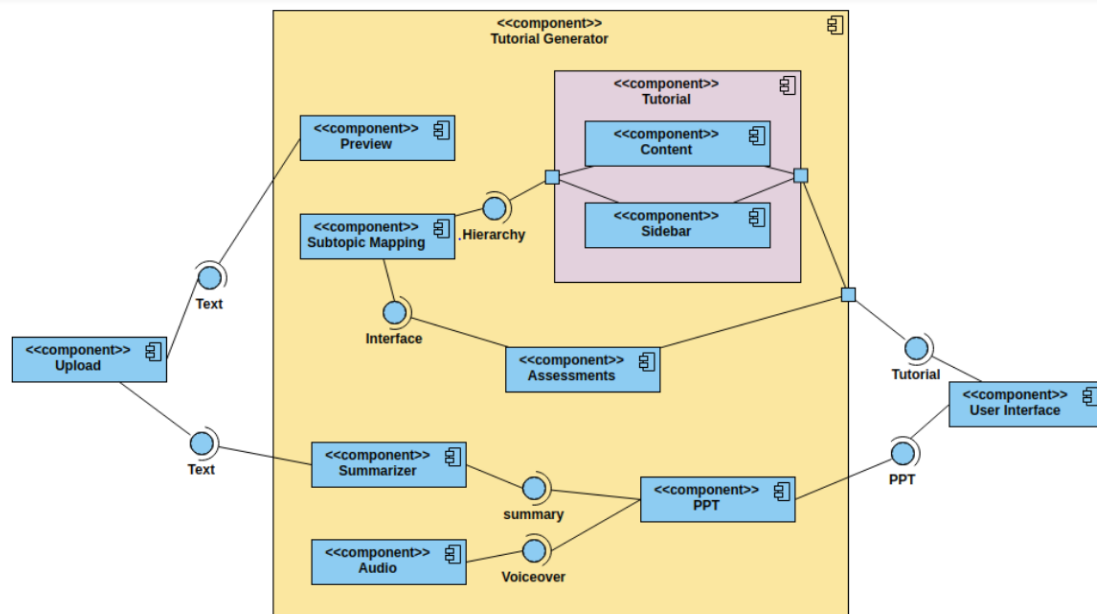


Figure 6.3: Component Diagram

6.5 ER Diagram

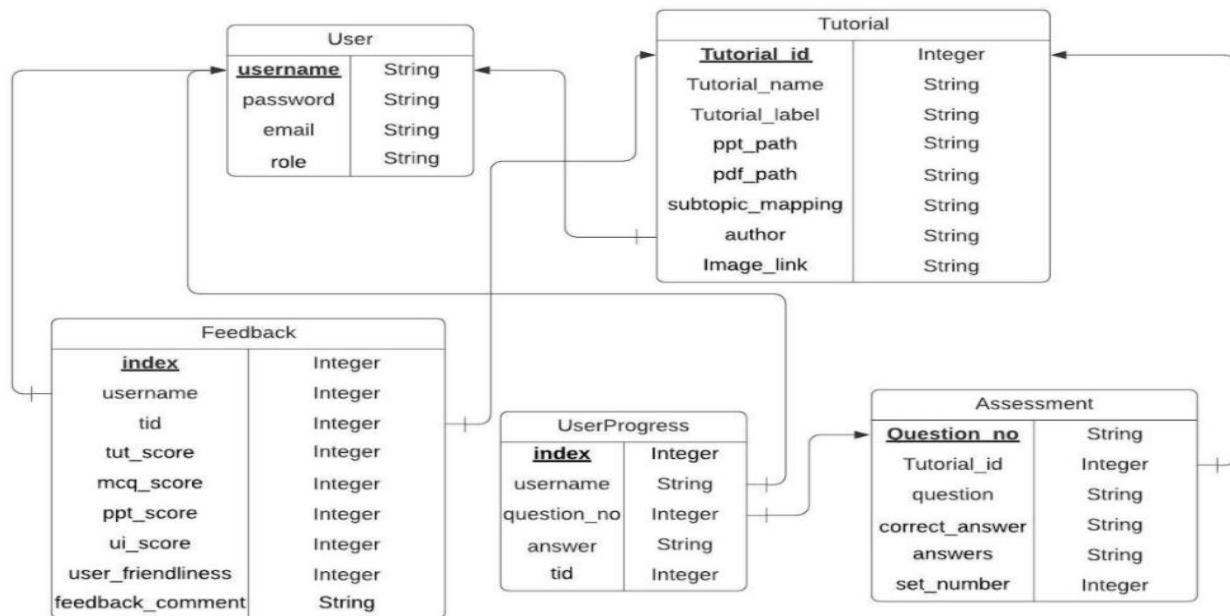


Figure 6.4: ER Diagram

#	Entity	Definition	Type
ENTITIES			
1.	User	Contains user details	Strong
2.	Tutorial	Tutorial path and details	Strong
3.	Assessment	Stores details about questions, answers and distractors	Strong
4.	User Progress	Has information about student scores	Strong
5.	Feedback	Contains user feedback provided.	Strong

Table 6.1: Entity Table

6.6 Activity Class Diagram

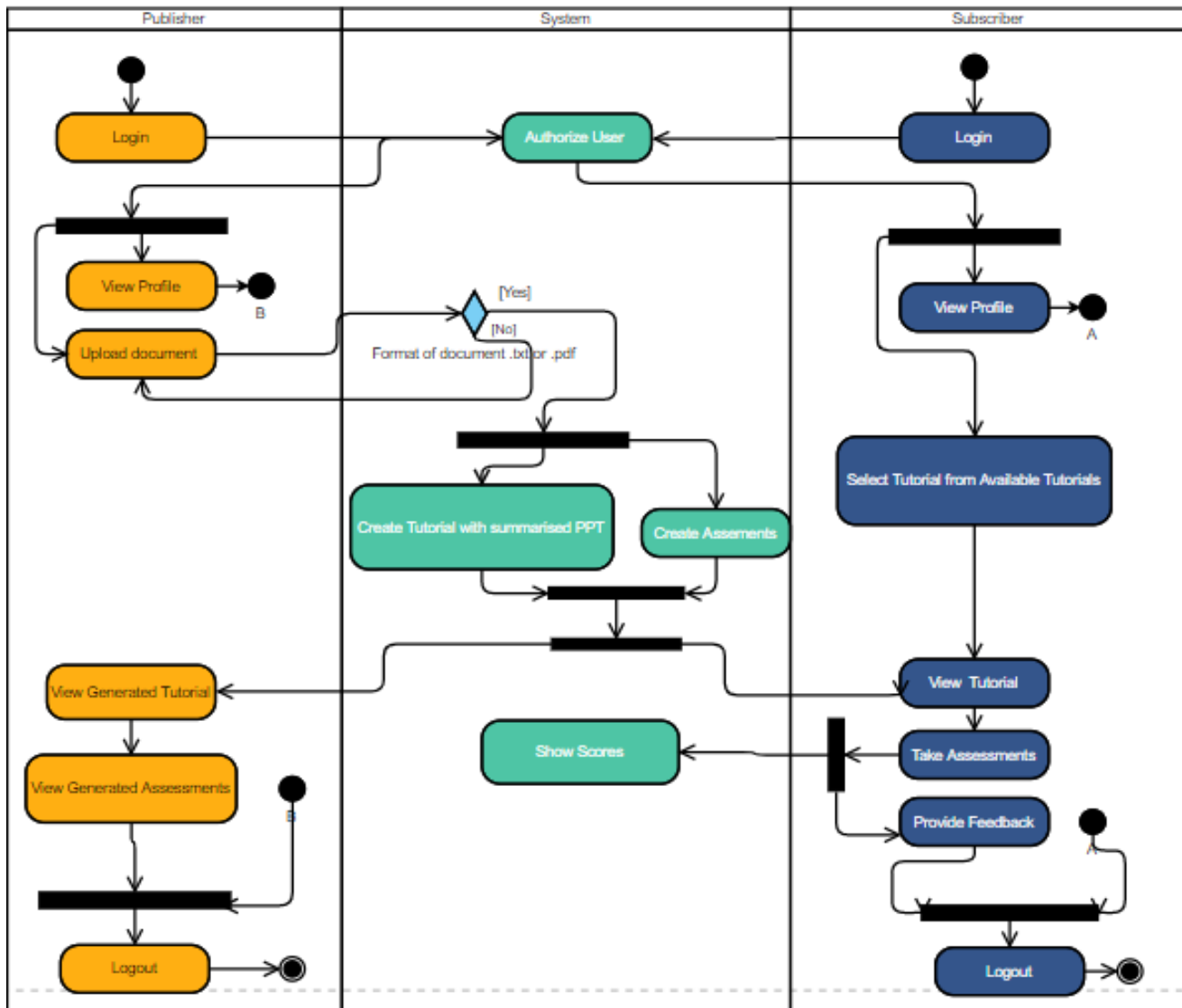


Figure 6.5: Activity Class Diagram

6.8 User Interface Diagram

6.8.1 Sign-up/Login

Options to login or sign-up. User is supposed to choose a specific role(creator/subscriber).

6.8.2 Creator

Provides the input text file. The input is processed to generate a navigable tutorial. Also contains a PPT with audio voice over. Subscriber: Subscribes to tutorials. Views and accesses the generated tutorials.

6.8.3 Assessment

Assessments are automatically generated through the creator post tutorial generation. The subscriber takes the assessment and instant results are provided.

6.9 Help

For help and resources, an installation guide and a user manual is provided at chapter 9 and 10 respectively of this document.

CHAPTER-7

IMPLEMENTATION

Our project involves creating a Navigable tutorial based on the input text file (PDF/txt) along with automatic generation of several components such as Summary Presentation, inherent Q/A system, Multimedia handling, Language Agnosticism and progress tracker.

7.1 Input

The generator is authorized to input a text file of PDF/.txt format. He/she will provide the name and category of the tutorial to be generated. On submitting the text file, the entire navigable tutorial with voiceover along with Q/A system and summarised presentation is created. A labelling picture for the tutorial is also automatically generated using web scraping. In addition to this, a quick preview of the text and important sentences containing summary is provided for the Generator.

7.2 Topic-Subtopic Modelling

The input text is analysed and the headings, sub-topics and paragraphs are extracted. The most used font can be considered the paragraph. The font size generated for text in the input document varied a lot due to size variation. This was then pre-processed to bucket the surrounding font sizes in pre-assigned categories, hence tagging the text with appropriate annotation to indicate post processed font sizes. PyMuPDF is used to identify the paragraphs, different levels of headers (larger than paragraph) and different levels of subscripts (smaller than paragraph).

This information is used to create a hierarchy of the headings, sub-topics and content in a recursive tree-based approach. This topic-subtopic hierarchy is later used to populate the navigable tutorial and sidebars in the correct order.

7.3 Navigable Tutorial

A navigable tutorial interface is provided using React components. All the important navigation tabs are placed on the top making them prominent for easy user. The user can sequentially access the tutorial in a hassle-free manner. Options to navigate to a specific sub-topic is provided too with the help of SideBar Menu. The SideBar Menu component is dynamically populated in a recursive manner using the tree-based Topic-Subtopic Hierarchy. The assessments can be easily accessed and taken by the 'subscriber' with automatic navigation to the next question, once he answers the current question. Advertisement sections are provided for online advertising which the users can navigate to. Subscribers can view the percentage of correct and wrong answers for all the subscribed tutorials in a visually engaging way.

7.4 Audio Voiceover

For each page in the navigable tutorial, the user can either read the content or listen to the content being read out with an audio voiceover which is provided. The context of the input text is mapped using the Topic-Subtopic hierarchy to provide audio explanation of the content. Implementation of the voiceover was done through the python gTTS module.

7.5 Assessment Generation

Automatic generation of assessments/MCQs is provided. This question generation module was implemented through an open source module named Questgen.ai. This module uses natural language processing techniques to parse and analyse what kind of questions to be generated on the key sentences. It uses WordNet to efficiently generate hypernyms which act as effective distractors/options for the MCQs generated. The generator can view the generated questions. The subscriber is provided the option to take assessments through the generated questions. These assessments are integrated with the tutorial to ensure user interactivity. After every two subtopics, the corresponding questions are generated from the text and mapped which is rendered as part of the tutorial. The scores/results of the assessments are instantly provided.

7.6 Presentation Generation

A presentation is also provided for the user that can be downloaded for a quick revision.

The first step towards creating a presentation is summarization. To do this, the input is firstly pre-processed to remove stop words. The text is then lemmatized and is passed to the NLTK summarizer. The NLTK summarizer is an extractive summarizer which efficiently fetches the important components of the input text.

The summarized text is provided as an input to the PPT module. Automatic generation of concise PPTs is implemented through the Python PPTX module. The slides are made effective by presenting the summarised text as concise bullet points not exceeding three per page. The presentation generated preserves the Topic-Subtopic hierarchy providing one slide from the main titles and maintaining the size of the subtitles for better user readability. Audio voiceover is also provided for every slide for better comprehensibility.

7.7 User Profile

Both the generator and the subscriber have user profiles which serve different purposes. The generator's user profile is dynamically populated with the tutorials created by him/her. The subscriber's profile is automatically populated with his/her progress. It shows the tutorials that the subscriber has finished along with the percentages of right answers and wrong answers in the assessment component of that particular tutorial.

7.8 Database

SQLAlchemy database was used.

The database consists of 5 tables - User, Tutorial, Assessment, UserProgress, Feedback.

1. User table is used to store the profile information and the respective roles (Creator/Subscriber) of the user.
2. Tutorial table consists of details about the tutorials stored in the database and is used to retrieve created tutorials.

3. Assessment table consists of the questions generated for the tutorial, the corresponding answers and the distractors.
4. UserProgress table is used to store the scores/results of the user and to keep track of the assessments which are taken up by the user.
5. Feedback table is used to get a detailed feedback about the application and to help enhance the user experience of the application.

7.9 Web Interface

A complete user-friendly web interface with a catalogue of automatically generated tutorials is provided for easy use of all the features and hassle-free learning. Fig.~\ref{fig1} shows the activity diagram depicting the flow of the activities in generating and accessing the tutorials and its various components.

- A. Frontend : Javascript based ReactJs is used for frontend. Different Components are written for different concerns using Single Responsibility Principle. Styling is done with the help of CSS.
- B. Backend : Python based Flask is used for backend. Different endpoints are written for different concerns using Single Responsibility Principle which is accessed by the specific frontend components.

7.10 Multimedia handling

The application can successfully handle multimedia in the input text file. The input PDF is parsed and the images in each page are identified and displayed on the tutorial. To achieve this, all the images are first extracted from the pdf. Then analyzing the topic-subtopic hierarchy, the appropriate image is rendered on the tutorial, under the correct heading to enhance user comprehensibility, thus making the tutorial more interactive rather than just informative.

7.11 Language agnosticism

Language agnosticism has been attempted by extending the application to handle Indian languages.

Currently, the application supports Kannada and Hindi.

NLP techniques and libraries have been utilized to clean, parse, summarize, display a navigable tutorial and generate assessments.

The application can further be extended to handle other languages using a similar methodology.

The input is firstly pre-processed to remove stop words. The text is then lemmatized and is passed to the NLTK summarizer. The NLTK summarizer is an extractive summarizer which efficiently fetches the important components of the input text.

CHAPTER-8

INSTALLATION GUIDE

The following section details about how to install the entire system in the local machine.

1. Pre-requisites:

- a. python3
- b. node.js
- c. java

2. Clone the git repository with the following command:

```
git clone https://github.com/KritikaKapoor13/Tutorial-generation-from-text-file.git  
cd Tutorial-generation-from-text-file
```

3. Create a virtual environment and activate it with the following commands:

```
python3 -m venv env  
source env/bin/activate
```

4. Setup ReactJS for frontend. First delete the existing node-modules and install them with node.js by following these commands:

```
cd frontend/new-ui/  
rm -rf node_modules  
rm package-lock.json  
npm install  
npm install react-scripts --save  
npm start
```

Navigate to localhost:3000 to view the frontend. However, the backend is not yet setup and hence you cannot navigate right now.

5. Setup flask for backend and download the necessary files with the following commands:

```
cd ../../backend/Create-tutorials-from-text-file/scripts/  
pip install -r requirements.txt
```

-
6. Some additional packages need to be downloaded with the following commands :

```
python3 -m spacy download en_core_web_sm
```

```
wget
```

```
https://github.com/explosion/sense2vec/releases/download/v1.0.0/s2v\_reddit\_2015\_md.tar.gz
```

```
tar -xvf s2v_reddit_2015_md.tar.gz
```

7. Run the backend in another terminal with the following command :

```
python3 trial.py
```

Installation is complete. You can now access your application at localhost:3000

CHAPTER-9

USER MANUAL

9.1 Prerequisites

1. A robust system with Windows/Ubuntu/MAC OS.
2. A browser to access the website.
3. A stable internet connection.

9.2 User Access Levels

There are two types of roles in this application:

1. **Generator** - The generator is majorly responsible for the creation of tutorials. Generator provides the input text file which is analysed, resulting in an automatic navigable tutorial with a concise presentation and an audio voice-over. The generator also has the option to view the assessment that is created for the generated tutorial
2. **Subscriber**- The subscriber can access the generated tutorials. On subscribing, a navigable tutorial with next and previous buttons are provided. An organized topic and subtopic hierarchy is presented where the user can directly navigate to a specific topic/subtopic.
The subscriber is also presented with timely assessments to enhance user interactivity. The results of the assessments are instantly generated and the progress of the user is tracked.

9.3 Generator Features

1. Sign-up specifying the role as 'Teacher' and login with the specified credentials.
2. Provide an input study material in the form of a text file (.txt/pdf)
3. View the automatically generated tutorial with an option to download a concise PPT with an audio voice over.
4. View the assessment/MCQs generated by analysing the input study material.

9.4 Subscriber Features

1. View the catalogue of all generated tutorials.
2. Subscribe to existing tutorials.
3. Access the navigable tutorial through next and previous buttons.
4. Access the topic-subtopic hierarchy and navigate instantly to a specific topic/subtopic.
5. Take-up the timely assessments that are provided.
6. View the results instantly and get access to information about past assessments and user progress.

9.5 How to access the application?

1. Set it up on the local machine - To set it up on the local machine, please follow the detailed installation guide specified in Chapter 10.
2. Access the website on - <https://frontend-tutorial-gen.herokuapp.com/>.

9.6 User Interface Screenshots

9.6.1 Generator Screens

Home Screen

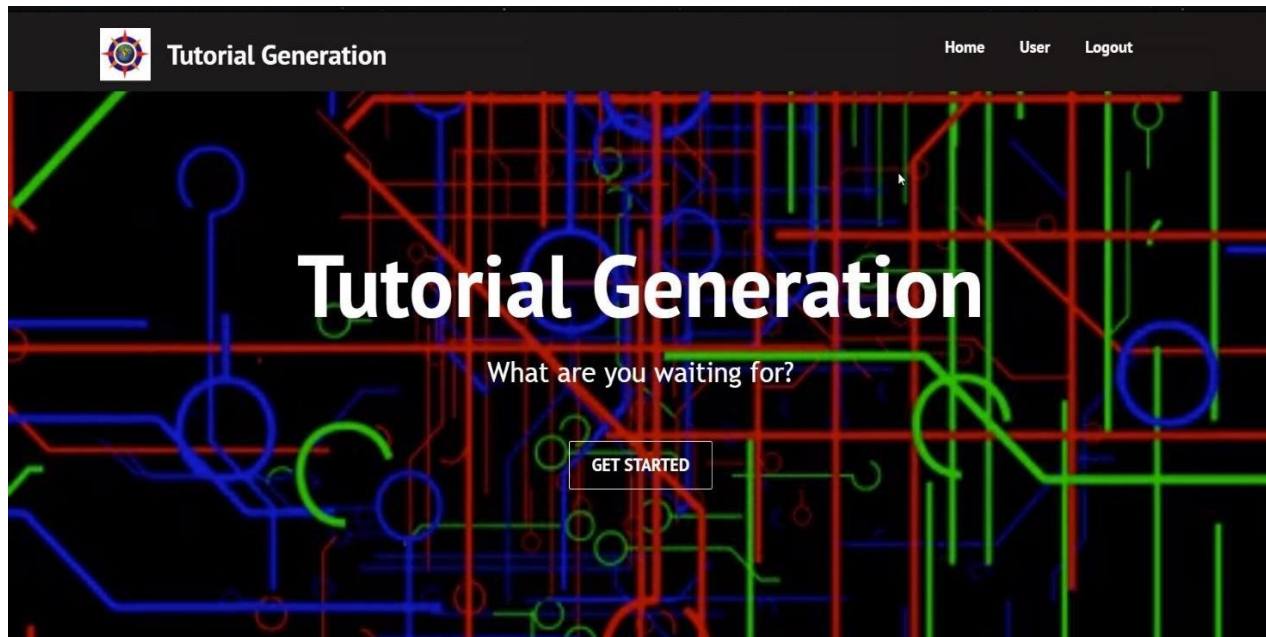


Figure 9.1: Home Screen

Upload Screen

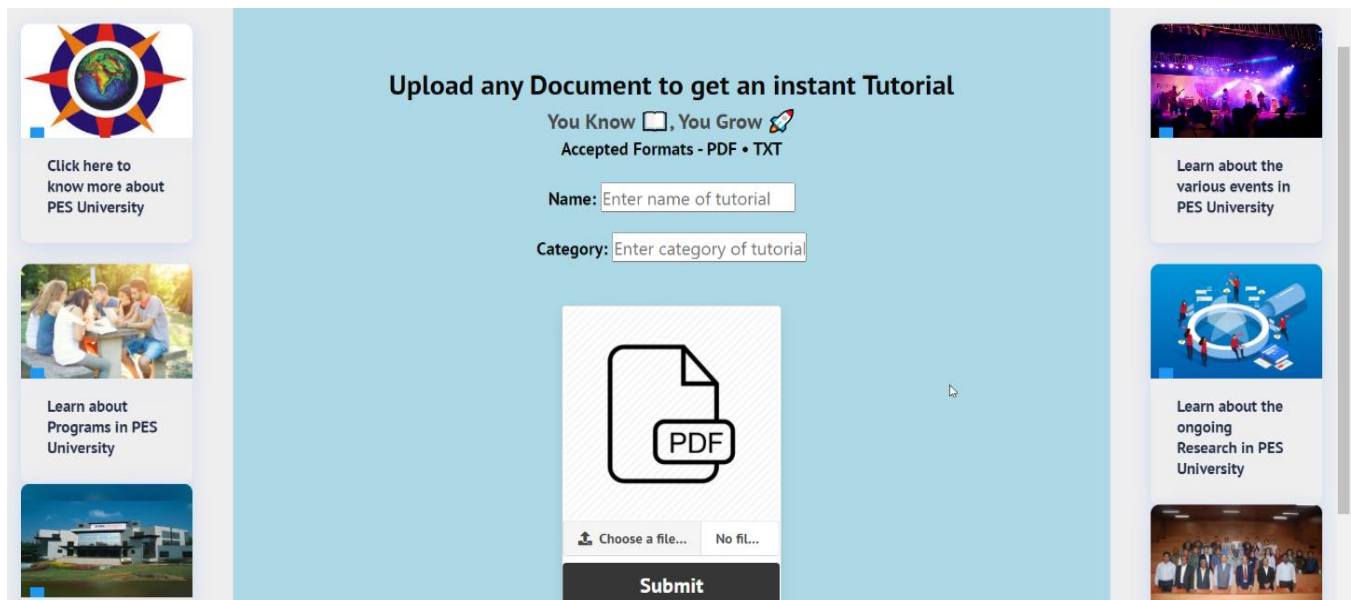


Figure 9.2: Upload Interface

View Assessment Option

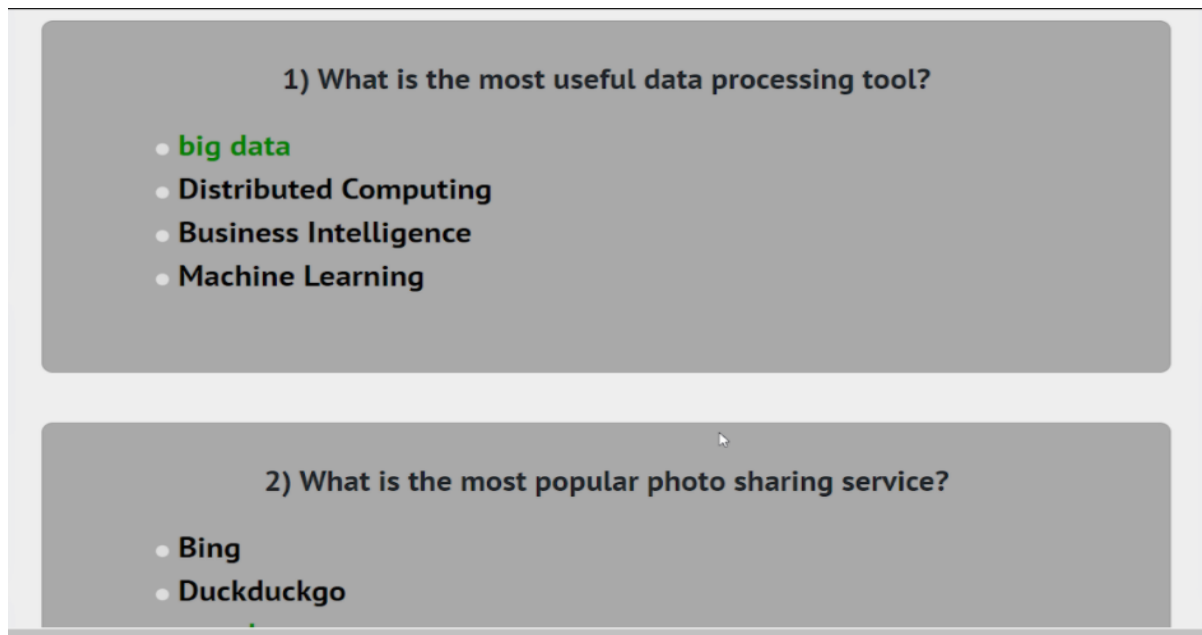


Figure 9.3: View Assessment Interface

9.6.2 Subscriber Screens

View List of all Existing Tutorials

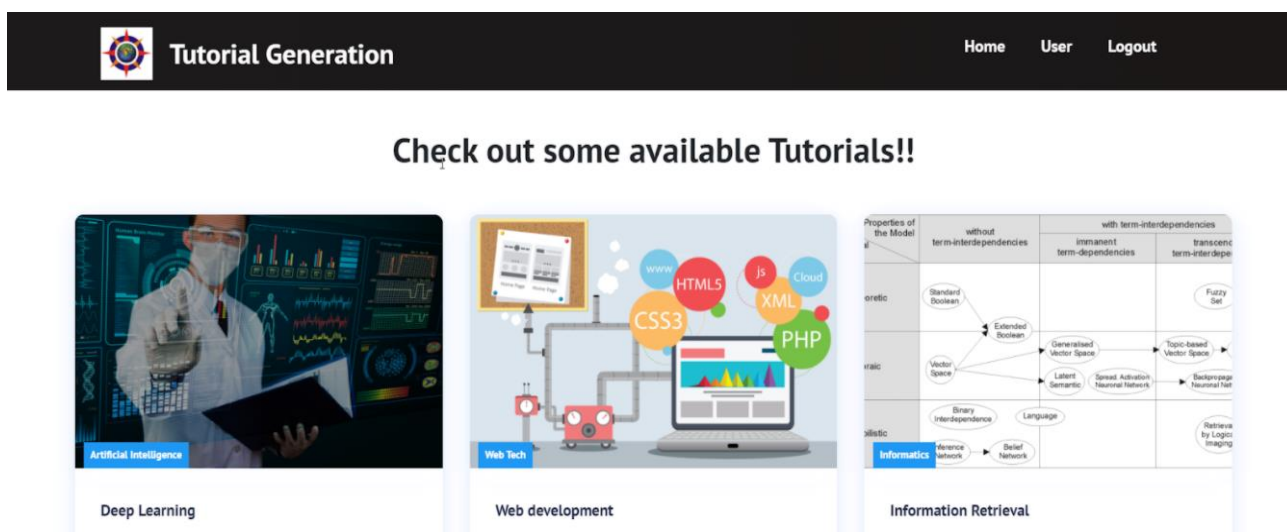
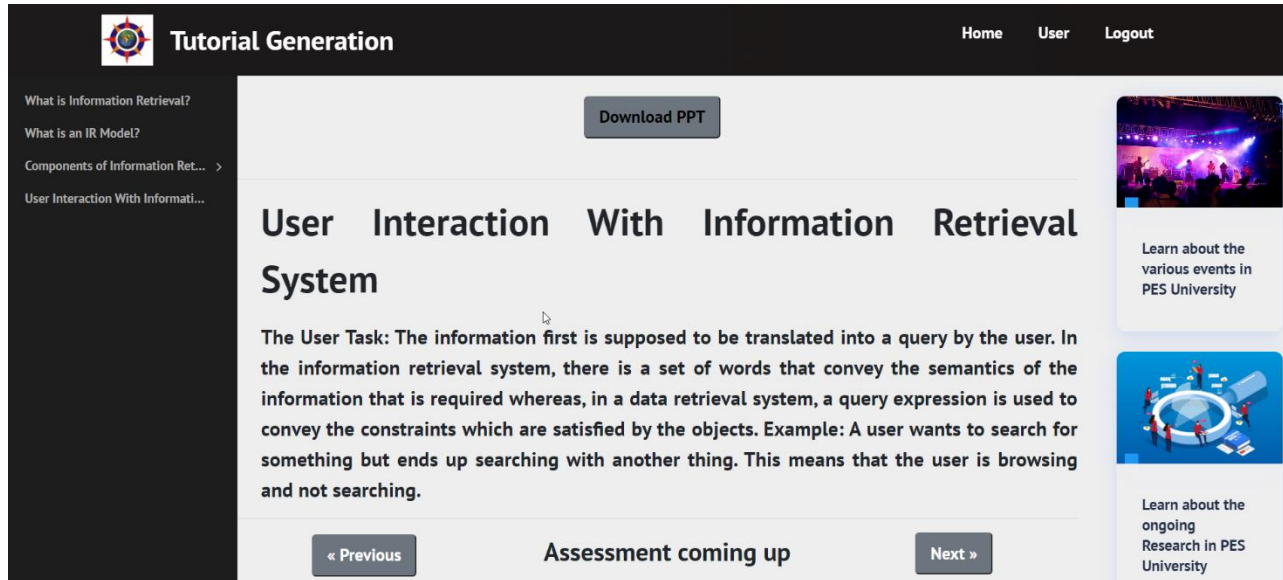


Figure 9.4: List of Existing Tutorials Interface

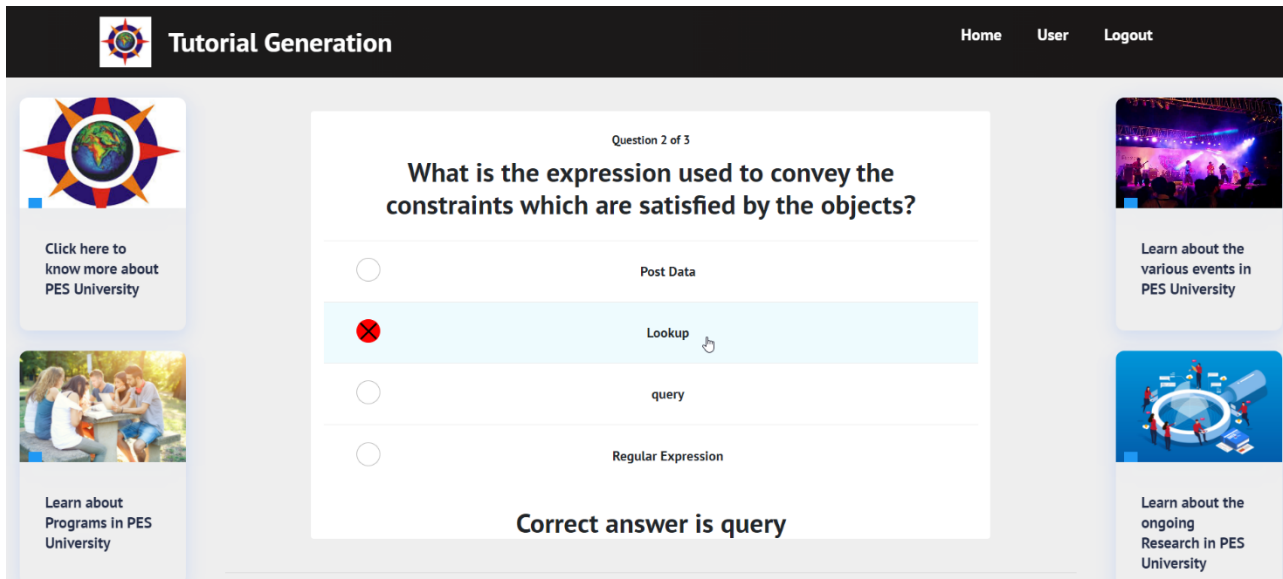
Access a navigable tutorial of choice with corresponding topic-subtopic hierarchy



The screenshot displays the 'Tutorial Generation' web application. The top navigation bar includes 'Home', 'User', and 'Logout' links. A sidebar on the left lists topics: 'What is Information Retrieval?', 'What is an IR Model?', 'Components of Information Ret...', and 'User Interaction With Informati...'. The main content area features a 'Download PPT' button and the title 'User Interaction With Information Retrieval System'. Below the title, a paragraph explains the user task: 'The information first is supposed to be translated into a query by the user. In the information retrieval system, there is a set of words that convey the semantics of the information that is required whereas, in a data retrieval system, a query expression is used to convey the constraints which are satisfied by the objects. Example: A user wants to search for something but ends up searching with another thing. This means that the user is browsing and not searching.' Navigation buttons for '« Previous', 'Assessment coming up', and 'Next »' are visible at the bottom of the main content area. The right sidebar contains promotional images and text about PES University events and research.

Figure 9.5: Tutorial Interface

Take-up Assessments



The screenshot shows the 'Take-up Assessment' interface. The top navigation bar is identical to the tutorial interface. The sidebar on the left includes a 'Click here to know more about PES University' link and a 'Learn about Programs in PES University' link. The main content area displays 'Question 2 of 3' with the question: 'What is the expression used to convey the constraints which are satisfied by the objects?'. Four options are listed: 'Post Data', 'Lookup', 'query', and 'Regular Expression'. The 'Lookup' option is selected, indicated by a red 'X' in a circle. Below the options, the text 'Correct answer is query' is displayed. The right sidebar contains promotional images and text about PES University events and research.

Figure 9.6: Take-up Assessment Interface

CHAPTER-10

USER STATISTICS

To collect user statistics, the application was deployed on PaaS (Heroku), so that it could be freely accessible to anyone with the link <https://frontend-tutorial-gen.herokuapp.com/>. We added feedback metrics at the end of the tutorial to be rated by the users after completion of the tutorial. The website link was sent to an audience of approximately around 20 people, who were asked to take up a tutorial of their choice from a set of six automatically generated tutorials.

The average time taken to create a tutorial from the text file was 103 seconds or 1 minute 33 seconds. This was averaged over 6 PDFs of varying sizes with maximum size being 19 pages and minimum size being 2 pages.

Feedback scores for UI, MCQ, User friendliness, PPT quality and Tutorial quality were recorded and averaged over to rate the entire application on a whole. The average mcq scores of every tutorial were also recorded.

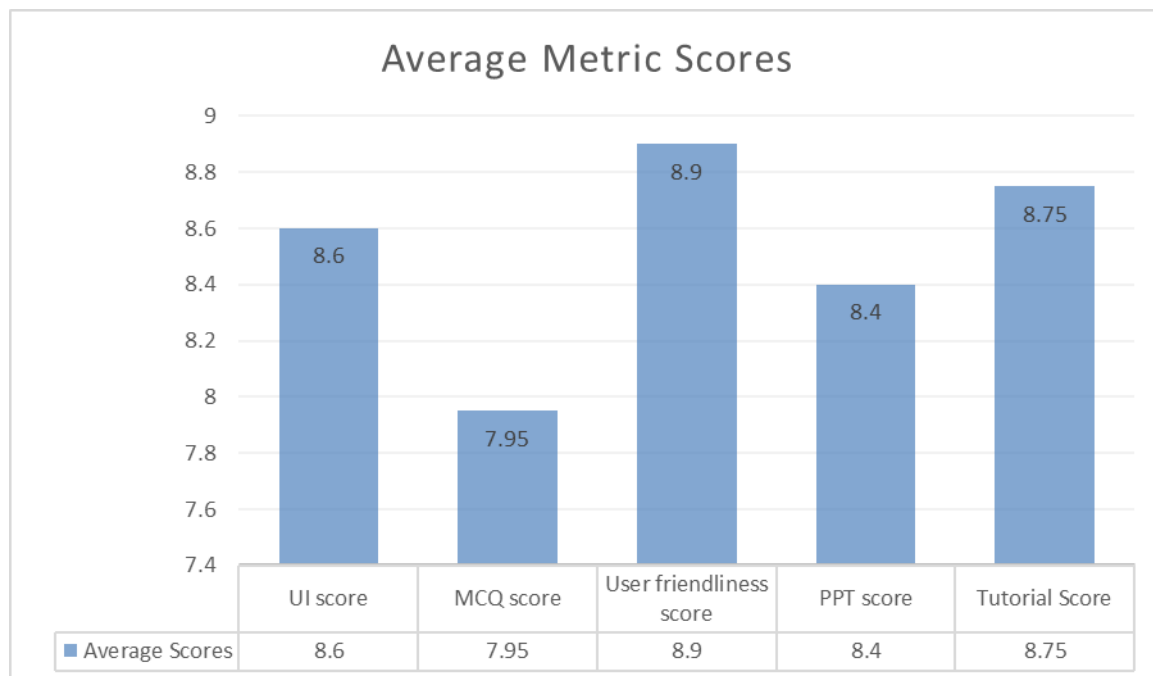


Figure 10.1: Average Feedback Metrics

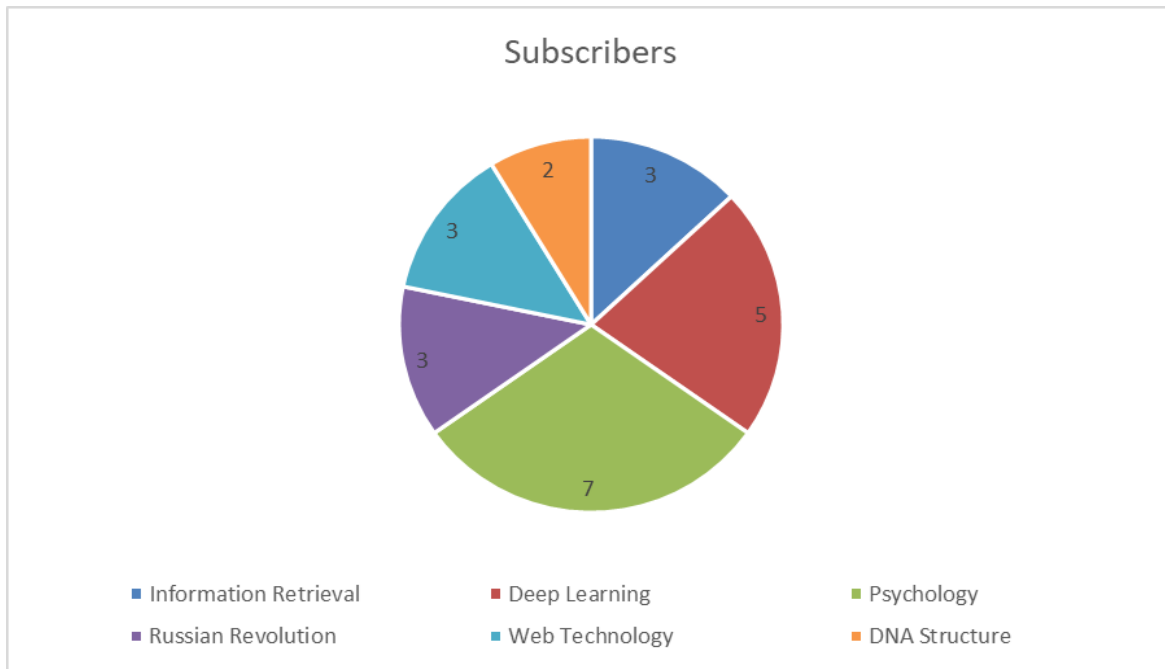


Figure 10.2: Subscriber Distribution

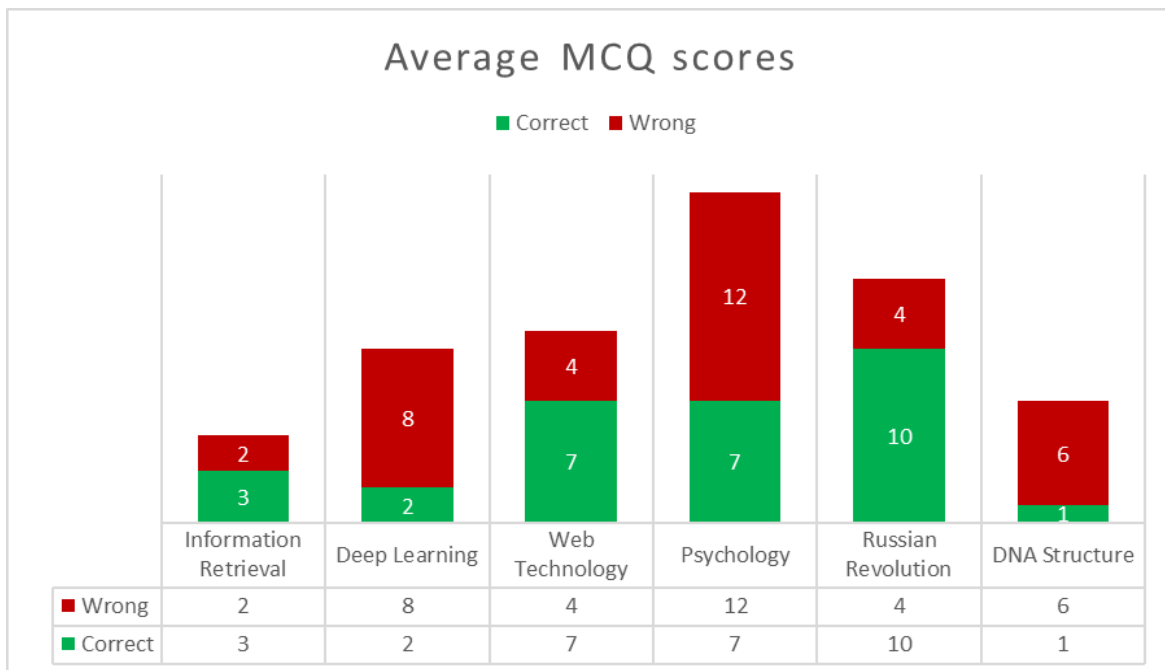


Figure 10.3: Average MCQ Scores of Pre-Generated Tutorials

We received valuable feedback from the users. It contained constructive criticism and some appreciation. A few selected verbatim feedbacks received are:

Sl.no	Feedback
1.	Quality of questions can be improved.
2.	It's difficult to see the correct answer for the last question. Please improve the quality of the question and the options.
3.	UI seems good. Instead of just showing User (next to Home button), one expects see his logged in Username over there. The content and the name of the PPT should be in a generalized format. None would read the PPT without any visual content in it. Given that this is a bachelor's project, it would be great if you show what kind of password protection techniques are you using during creation of a user entry. There is scope for improvement in assessment questions generation. Intuitive interface, but can be smoother. Could have been a bit more organized with the course content. The assessment questions were very vague and sometimes had same options worded differently. But the options were indeed relevant and within the scope of the question
4.	A more detailed navigation bar, that is, with more tabs, like progress, tutorial, assessments etc would be better
5.	The generated ppt covers the topic well. The quality of assessments can be better. In particular, the options for the questions need to be improved.

Table 10.1: Feedback comments

CHAPTER-11

CONCLUSION AND FUTURE WORK

Successfully built an end-to-end application where the content-creator can provide an input text file and a full-fledged tutorial with assessments and progress tracking options, and an end user, a subscriber can take advantage of the created tutorial to learn the topic.

However multiple failures in different parts of the application were encountered.

11.1 Sentence Simplification

Multiple attempts were made at sentence simplification to display concise and crisp sentences as bullet points in the summarized PPT component. However, they involved the use of a complicated Deep Learning model to achieve satisfying results, which hasn't been looked into yet. The methods tried were -

1. Basic lemmatization, Stop Word removal and POS tagging
2. Spacy-Claucy
3. SVO method
4. Paraphrasing

All of these methods returned substandard results, hence the sentences in the PPT are directly picked up from the input document, performing raw extractive summary.

11.2 Assessment Generation

Multiple attempts were made at the Automatic assessment generation component and the failures are documented below.

1. Our first attempt used spacy and had manually hardcoded cases to generate questions based on the parsed input text. This was not completely efficient and the approach posed new challenges for evaluation.

2. Our second attempt involved generation of MCQs along with distractors for achieving automatic evaluation. The performance was average, but the distractors generated weren't good enough.
3. Our third attempt was QuestGen.AI, which performed considerably well, with relatively good performance. This technique was employed in the project.

11.3 Topic Detection

Attempts at detecting the topics in the input document are described as follows.

1. The sentences of the document were represented as word embeddings, converted to vectors to better emphasize the meaning of the sentence. Multiple types were tried upon like -
 - a. Bert
 - b. InferSent
 - c. Word2vec
 - d. Universal Sentence encoder
2. Clustering methods were tried to get subtopic segregation for the entire document namely K-Means and Agglomerative. This method performed appreciably well with Word2Vec but better with Universal Sentence Encoder at the expense of time, but the sentences being clustered weren't following the paragraph hierarchy.
3. Next attempt was based on sequential sentence clustering to generate subtopics, similar to consecutive track generation algorithm.
4. For generating a topic for the document LDA topic detection algorithm was tried with TF-IDF scoring methodology.

We tried to resolve the failures.

11.4 Subtopic Modelling

For a document with a proper outline, a font-size analysis method was employed to get the structure of the document. For an improper structured document, the following methods were attempted upon.

1. Font based approach:

The document was parsed based on font sizes and were classified into their corresponding HTML tags according to the document hierarchy.

2. Structure based approach:

- a. Grobid
- b. PDFtotree
- c. Docx

3. Removal of header and footer from the input document to create the hierarchy in a cleaner manner.

4. Extraction of tables and their content from PDF

- a. Tabula
- b. Camelot
- c. Excalibur

11.5 Future Work:

1. Provide user recommendations
2. Handling of code segments, tables and mathematical formulae

REFERENCES/BIBLIOGRAPHY

- [1]. Iwata, H., Shirogane, J. and Fukazawa, Y., 2006, March. Automatic generation of tutorial systems from development specification. In *International Conference on Fundamental Approaches to Software Engineering* (pp. 79-92). Springer, Berlin, Heidelberg.
- [2]. Flanagan, B., Akcapinar, G.Ö.K.H.A.N., Majumdar, R. and Ogata, H., 2018, November. Automatic generation of contents models for digital learning materials. In *26th International Conference on Computers in Education Main Conference Proceedings* (pp. 804-806). Asia-Pacific Society for Computers in Education (APSCE).
- [3]. Sariki, T.P., Kumar, B. and Ragala, R., 2014. Effective Classroom Presentation Generation Using Text Summarization. *IJCTA, ISSN*, pp.2229-6093.
- [4]. Phage, K.R., Rawade, S.S., Thorat, K.S. and Agawane, R.V., A SEMANTIC MACHINE LEARNING APPROACH TO AUTOMATIC PPT GENERATION.
- [5]. Belote, P., Bidwai, S., Jadhav, S., Kapadnis, P. and Sharma, N., Enhancing Automatic PPT Generation Technique through NLP for Textual Data.
- [6]. Sravanthi, M., Chowdary, C.R. and Kumar, P.S., 2009, March. Slidesgen: Automatic generation of presentation slides for a technical paper using summarization.
- [7]. Gutl, C., Lankmayr, K., Weinhofer, J. and Hofler, M., 2011. Enhanced Automatic Question Creator--EAQC: Concept, Development and Evaluation of an Automatic Test Item Creation Tool to Foster Modern e-Education. *Electronic Journal of e-Learning*, 9(1), pp.23-38.
- [8]. [Raw Shorts: Online Video Maker | Video Creation Software](#)
- [9]. <https://www.viomatic.com/en/home/>
- [10]. <https://lumen5.com/>
- [11]. <https://www.google.com/slides/about/>
- [12]. <https://convertio.co/>
- [13]. <https://www.typeform.com/>
- [14]. <https://software.intel.com/content/www/us/en/develop/articles/using-natural-language-processing-for-smart-question-generation.html>

-
- [15]. <https://questgen.ai/>
- [16]. Details of Indic NLP libraries.Retrieved from <https://www.analyticsvidhya.com/blog/2020/01/3-important-nlp-libraries-indian-languages-python/>
- [17]. Datta, A., Ramabhadran, B., Emond, J., Kannan, A. and Roark, B., 2020, May. Language-agnostic multilingual modeling. In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 8239-8243). IEEE.
- [18]. Sunitha, C., Jaya, A. and Ganesh, A., 2016. A study on abstractive summarization techniques in Indian languages. Procedia Computer Science, 87, pp.25-31.
- [19]. Li, Y., Min, M., Shen, D., Carlson, D. and Carin, L., 2018, April. Video generation from text. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 32, No. 1).
- [20]. Kim, D., Joo, D. and Kim, J., 2020. TiVGAN: Text to Image to Video Generation With Step-by-Step Evolutionary Generator. IEEE Access, 8, pp.153113-153122.