Bilkent University
Department of Computer Engineering

Senior Design Project

Project short-name: Prexcel

High Level Design Report

Burak Yiğit Uslu, Can Kırımca, Can Kırşallıoba, Alper Sarı, Barış Tiftik

Supervisor: Dr. Özcan Öztürk
Jury Members: Dr. Shervin Arashloo and Dr. Hamdi Dibeklioğlu

High Level Design Report
December 24, 2021

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

Contents

High Level Design Report

Project Short-Name: Prexcel

1 Introduction

This report includes the high level system design of the Prexcel app. Prexcel is a multifunctional presentation assistant, whose functionalities and usage is discussed and described in detail under section 1.1 “Purpose of the System”. Design goals for the system are discussed under section 1.2 “Design Goals”. Section 2 titled “Proposed Software Architecture” discusses the subsystem decomposition, the hardware/software mapping, persistent data management as well as the access control, security and global software control are discussed under their respective subsections. Section 3 titled “Subsystem Services” discusses the design of the presentation layer, logic layer and the data layer in their respective subheadings. Consideration and the effect of various factors in our engineering design is discussed under section 4 with the same title. Under section 5 “Teamwork Details” each of the team members' contributions in different contexts are discussed under their respective subheadings. Finally, the definitions and acronyms used in the report are discussed under section 1.3 titled “Definitions, Acronyms and Abbreviations” and definitions and explanations of various software systems or libraries we use in the project are explained in section 7 “Glossary”.

1.1 Purpose of the System

Prexcel is a presentation assistance application, whose main purpose is to help its users improve their presentation abilities and their presentations. The app accomplishes its purpose by giving feedback and grades in different categories to user’s presentations and by providing live feedback to the user when presenting. The areas graded are the delivery of the presentation, use of language, and facial orientation of the presenter. It is geared towards online presentations in particular. Prexcel has two main functionalities. The first main functionality is analyzing the user’s presentation from beginning to the end and providing detailed feedback that contains the app's grades for the particular presentation, recommendations and feedback regarding how to improve the presentation as well as various relevant statistics. This analysis is based on the user’s speech, as well as his/her facial

orientation. All of these are presented to the user in a detailed analysis report. The grading is done for 3 categories. The first category is the delivery, which is based on the number and length of pauses in the presentation and how fast or slow the user's speech is. Second category is the use of language, which is based on how often the user repeats the same words, and the amount of filler words in his/her speech. Last category is the facial orientation of the user, which is graded based on how much the user does not look at the screen, and whether the user's face could not be detected at certain times (for example when he/she closes his face with his/her hand). All of the filler words, and words that are frequently repeated and such are also marked on the transcript of the speech, which is also provided in the detailed analysis report. For creating the transcript of the speech, speech recognition is used to transcribe the presenter's words into text, from there the program recognises filler words and phrases as well as noting down extended pauses used in their speech. The end report provides alternative word recommendations to the user for words he/she uses excessively, as well as gives feedback to the user on talking faster or slower. It provides this feedback based on statistical data, such as "If you had made only half of the pauses you made, the presentation could have been 5 minutes shorter." or that "If you used %30 less filler words, the presentation could have been shortened by 10 minutes." and such. The user has the option to download the detailed analysis report.
Second main functionality of the app is the live feedback functionality. If the users choose to perform the presentation live, while using the app, the app can be used to give instantaneous feedback to the user regarding his/her voice level, and "word per minute" speed. It also recommends words to the user based on the flow of the sentence when the user gets stuck or has an extended pause.
The users have the option to either upload a prerecorded video of a presentation, or capture the video of their presentation via the app. The live feedback functionality is available only on the latter case.
Users can also track their progress across different presentations (or different trials for a particular presentation) using the progress report functionality. Fundamentally, users can select a subset of their previous presentations, and view using charts and tables how their grades have changed over these presentations. They can download these progress reports as well.
It is important that we can provide the user with a tangible set of data regarding their presentation. Therefore, we would like to emphasize that the report is mainly based on statistical and mathematical phrasing of the presenters' behaviors, such as for

how long have they looked in other places in total, or how much could they shorten their presentations if they got rid of the extended pauses in their speech. For the convenience of the user, this analysis data and their compiled reports will be held in their user accounts, but of course no personal or audiovisual data will be kept to comply with user safety.

1.2 Design Goals

In light of the requirements of the system, design goals considered for Prexcel are based on a variety of criteria including end-user criteria, performance criteria, and design criteria.

1.2.1 End-User Criteria

Our top end-user criteria for Prexcel is mainly usability and utility. As also discussed in the subsection titled “Consideration of Various Factors in Engineering”, our system targets a wide array of audience, some of whom may use the system for business purposes in a variety of fields, and others may use it for educational purposes or even as an assistance in their language learning efforts. Therefore, first of all, it is of crucial importance that Prexcel has a high usability and users from different backgrounds can use the application with ease. We aim to achieve this with an intuitive GUI design, and carefully crafter tutorials on both how to give effective presentations as well as how to use the app effectively. Second criteria in this part is utility. Prexcel must cater to the needs of its target audience. Therefore it includes many functionalities like exporting reports and transcripts to pdf, or an account system that allows users to access their past presentation data from any computer, both of which are fundamental features for both educational and for business use.

In conclusion, our top end-user criteria are usability and utility.

1.2.2 Performance Criteria

Our top performance criteria for Prexcel is mainly the reliability and the response time. As previously discussed, Prexcel has 2 main functionalities, one is the detailed end analysis and the other functionality is the live-feedback during presentations. For both of these functionalities, it is important that the app reliably predicts the words the user spoke (for speech to text conversion) and whether the user is looking at the screen/camera above a certain accuracy threshold, which is for the purposes of our

app, 70% of the time for an average English speaker. In addition to this, especially for the live-feedback functionality, response time is of critical importance. If the app cannot make recommendations to the user when necessary in a timely manner, live feedback functionality loses its purpose. Therefore, whenever the user stops for more than 2 seconds, the app should make a word recommendation to the user in no longer than 1.5 seconds. Furthermore, the statistics presented to the user on the screen such as the volume of his/her voice or the WPM of the user, should be at most the statistics from 2 seconds before the moment the statistic is presented. In order to achieve the optimal reliability and response times, some performance trade-offs might be necessary and additional memory space may be used in order to enhance the performance in the other areas mentioned. Still, the total memory requirements of the programme should not exceed 2 GBs at any time, so that the app can also be used by people who do not have high-end computers.

In conclusion, our top performance criteria are reliability and response times. This might come at a trade-off of memory space.

1.2.3 Design Criteria

Our most important design criteria for Prexcel is robustness. If the app crashes halfway during a presentation trial, the user would lose the presentation data and would have to start the presentation from the beginning in order to get the detailed end analysis for the presentation. It is important that the app does not waste the user's time and efforts. Furthermore, if the app crashes during an actual live presentation (where it might be used for its live-feedback purposes) the crash might throw off the user and make the user unnecessarily excited, which might affect his/her presentation in a negative way. Therefore it is of crucial importance that the app is robust and bugs that can crash the app must not occur.

In conclusion, our top design criteria is robustness.

1.2.4 Maintenance Criteria

Our top level maintenance criteria for Prexcel is extensibility and modifiability. We especially want to enhance the system's extensibility and modifiability by designing the system as modular as possible. We aim to design Prexcel in a way so that the

“top-level” subsystems (which are divided into subsystems within themselves), the database, the graphical user interface, and the backend processes that make-up the app have as minimal coupling as possible and are modular. This allows for different teams working on these subsystems to work independently from each other and an easier division of labor as well as a higher maintainability. The classes should communicate with each other through communicating functions (similar to the facade design pattern) and as long as these communicating classes/functions are not changed, the changes in one subsystem should not affect the others as much as possible. This design should render the app very extensible and modifiable.

In conclusion, our top maintenance criteria is extensibility and modifiability.

1.3 Definitions, Acronyms, and Abbreviations

AWS: Amazon Web Services. Database for the system is hosted on AWS. [1]

BERT: Bidirectional Encoder Representations from Transformers. It originally is a paper published by Google, it does a variety of tasks. These tasks include Natural Language Processing tasks, Question Answering tasks, predicting words from a context etc [2].

GUI: Graphical user interface.

Model: Model within the context of this project refers to machine learning models, which is fundamentally a program that is trained to recognize patterns. Within the context of this project, machine learning models will be used for converting users speech to text, to recommend words to the user based on the flow of the sentence, and to decide whether the user is looking at the camera (for the face contact grade).

OpenCV: Open source computer vision library [3].

Prexcel: Prexcel is the name of our system, the multifunctional presentation assistant desktop application.

Progress-Tracking: Within the context of Prexcel, progress-tracking refers to the comparison of grades between the selected presentations and consequently an overview of the users presenting ability's improvement.

Speech-To-Text: Conversion of auditory input to text format.

UI: User interface.

1.4 Overview

In conclusion, Prexcel is a multifunctional presentation assistant that helps users prepare for and deliver their presentations better by its two main functionalities, that is the detailed analysis and the related detailed analysis report, which contains a detailed analysis of the user's use of language and delivery as well as face orientation, and the live feedback mechanisms that makes word recommendations and presents the user with important data while presenting. There are also account-related features in the app, such as looking back at the previous presentation grades or transcripts, as well as progress tracking by selecting a particular subset of presentations.

For the design of this system, we had design goals based on a variety of different criteria. From the end user criteria perspective, the system is designed to maximize usability and utility. From the performance criteria perspective, our main design goal was to maximize the reliability and the response times, at the expense of memory requirements if necessary. From the design criteria perspective, our main design goals are robustness and modularity. Of these two design goals, the modularity is also closely related to the maintenance design goals such as extensibility and modifiability, which are also enhanced by opting for a modular design.

2 Proposed Software Architecture

2.1 Overview

Prexcel offers two options to the users. It offers a live presentation module; where the users give presentations while Prexcel is analyzing them. During the presentation, Prexcel gives live feedback to the users when it detects gaps in the presentation. At the end of the presentation, it generates a report containing the transcript of the

presentation and various statistical data regarding the users' presentation. The second module is the pre-recorded presentation analysis. In this module, the user uploads either an audio file or a video file. Then these files are processed by the analyzer modules. This analysis is demonstrated to the users via a report, the contents of the report are the same as the report generated at the end of the live presentation module. Another functionality of Prexcel is that the users are able to access the reports and statistics of their previous presentations, and also the users are able to select a few of their prior presentations to generate a 'progress plot'; where they are able to track their progress over time.

2.2 Subsystem Decomposition

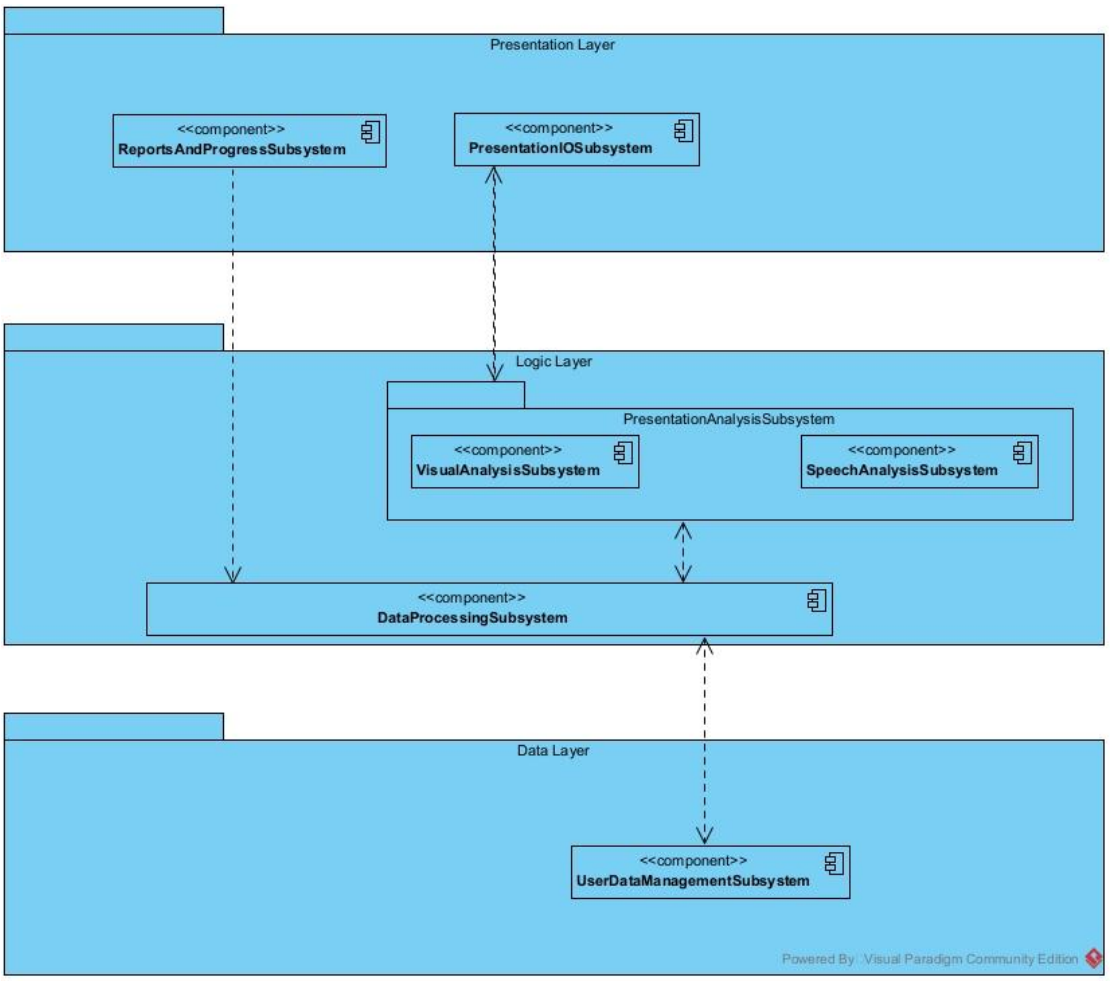

Figure 1. Subsystem Decomposition of Prexcel

For the subsystem decomposition, the system is divided into three layers; Presentation Layer, Logic Layer and Data Layer. Within these layers reside the appropriate subsystems. The reason for this mechanism is to decrease the coupling between the subsystems and to increase the coherence between them. Therefore, by using 3-Layered Architecture, the logic of the program is separated, which allows the system to be organized in a better manner.

- **Presentation Layer** includes the subsystems that interact with the user, which means that it consists of front-end components. This layer communicates with the logic layer to send the raw input data and receive processed data.
- **Logic Layer** includes the subsystems that implement the application logic. These subsystems communicate with the presentation layer to process user input, and also with the data layer to save and retrieve persistent data.
- **Data Layer** communicates with the external database server. This layer performs the data storage and retrieval operations on the MySQL database.

2.3 Hardware/Software Mapping

Figure 2. Deployment Diagram for Prexcel

Hardware/Software mapping of Prexcel is simple since the application will run only on the user's computer. For full functionality, Prexcel requires hardware resources of the computer such as camera and microphone. The user and presentation data will be stored on a MySQL database [4] hosted on Amazon Web Services (AWS) server. The application will communicate with this server in order to send and receive necessary user data.

2.4 Persistent Data Management

Prexcel will store the presentation reports and transcripts associated with each registered user. Therefore, user and presentation data needs to be stored in a database. For this purpose, a MySQL database on Amazon Web Services (AWS) Cloud Server will be used. The reason cloud database is chosen is to not bother the users with additional steps when trying to access their account information.

In the database, each user's user ID, username, password and email address is stored in a table called User. Furthermore, each user's presentation records are stored in a table called Presentation. Each presentation will be associated with a user ID so that each user will be able to access their past presentation reports and transcripts. As more users register to Prexcel, the user table will be populated, and when the users upload or record their presentations, the presentation table will be populated.

Since Prexcel will only store the reports and transcripts of the presentations, any uploaded or recorded visual and auditory data will be disposed of after necessary analysis operations.

2.5 Access Control and Security

Prexcel has an accounting system with a single type of user. The users need to provide their unique username and password to log into the system and use the application.

For the live recording feature, the system needs to have access to the camera and microphone of the user. After downloading the application, the system asks for permission to get access to these hardware components. If the user prefers not to give this permission, the application can still work without the live recording feature. Prexcel does not require any additional installation other than simply downloading the application package. For security, we will implement several safety measures since sensitive data is being stored in the database. Passwords of the users will be encrypted to increase the security of the user's account. In addition to that, token based authentication will be implemented to provide an even safer system. Lastly, the users' videos and voices will not be stored in the database after the processing of the videos for the sake of privacy.

2.6 Global Software Control

Prexcel is an event driven application, in line with the general OOP design paradigm. Functionalities of the system are initiated by events from the user, using the GUI. One important process that is worth looking at in detail in this section is the live feedback functionality. It is first started by an event (the user starting the presentation), and terminated by an event (the user terminating the live presentation session) however, between these 2 events, the live presentation feedback session continues indefinitely (given the computer does not run out of memory). Therefore even though in general this application is event driven, this particular functionality is procedural, it keeps updating the screen with new statistics and words until terminated.

2.7 Boundary Conditions

- **Initialization**

After Prexcel is downloaded and opened, a register screen will be displayed. In this screen, the users can create new accounts, which will add a user instance to the database. If the user already has an account, they are able to navigate to the login screen where they can access their accounts by providing their usernames and passwords. After registering or logging in, Prexcel becomes ready to use. Since Prexcel utilizes a cloud database, it is completely portable. Which means it is compatible to use from multiple computers without data loss on the account.

- **Termination**

By clicking on the ‘Exit’ button on the main menu, the users can close the application. After that, the application will log out of the user’s account and terminate the connection with the database. Any user input other than reports and transcripts will be disposed of after termination.

- **Failure**

In order not to disturb the workflow of the users, Prexcel autosaves the presentation data (such as the transcript and the statistics) within predefined time intervals. This is especially useful in an occurrence of a crash, in that case, the last autosave before the crash will be available for the user to view.

3 Subsystem Services

3.1 Presentation Layer

Presentation layer consists of the GUI and is implemented in JavaScript [4], using React [5] and Electron [6].

3.1.1 ReportsAndProgressSubsystem

This subsystem is responsible for displaying the reports and the progress plots of the selected presentations from the user's account. This subsystem depends on DataProcessingSubsystem, it gets all the processed data and statistics from that subsystem.

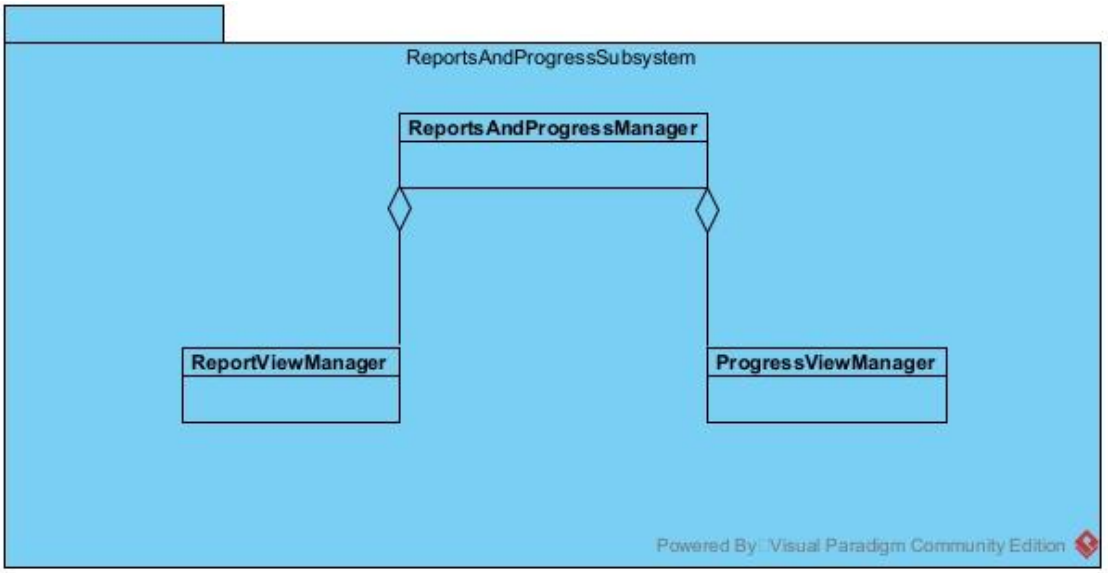

Figure 3. Reports and Progress Subsystem Service

3.1.2 PresentationIOSubsystem

This subsystem is responsible for displaying the presentation recording, uploading modules' user interface and the live recording screen's user interface. This subsystem depends on the VisualAnalysisSubsystem and the SpeechAnalysisSubsystem, it gets the live feedback data from said subsystems but it also provides them with the data to be processed (raw presentation stream, video file or audio file).

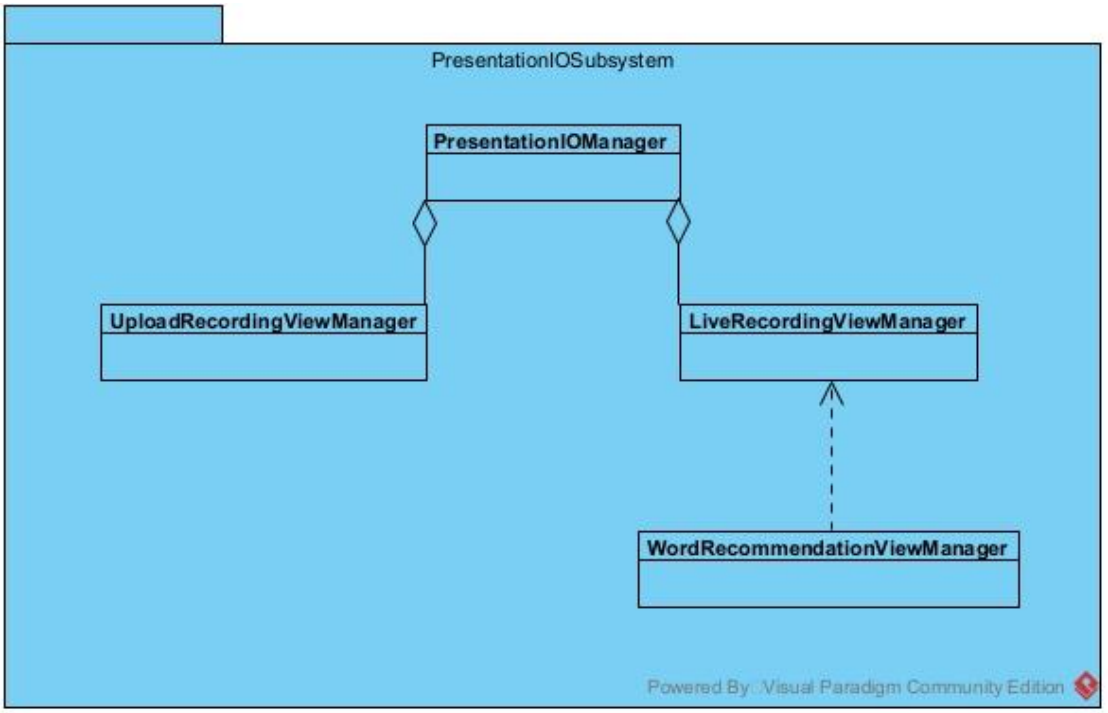

Figure 4. Presentation IO Subsystem Service

3.2 Logic Layer

3.2.1 PresentationAnalysisSubsystem

This subsystem includes two subsystems, VisualAnalysisSubsystem and SpeechAnalysisSubsystem. Since there is a common ground between these two subsystems, the union of the functionality is handled on the PresentationAnalysisManager. These functionalities include PresentationAnalyzer and the PresentationAssistant modules. Apart from that, the VisualAnalysisSubsystem is responsible for the analysis of the visual data. This includes the Face Recognition model (which will utilize the OpenCV [3] module), which aims to detect the presenter's face, or in other words, it will give the series of time segments of when it cannot detect the presenter's face to the DataProcessingSubsystem. And lastly, the SpeechAnalysisSubsystem (which utilizes DeepSpeech [8] module) is responsible for the analysis of the speech data. This includes the Speech to Text model and the Word Recommendation Model. Speech to Text Model transforms the audio files or the live streams audio to a transcript, it also takes into account where there is a gap in the presentation's flow and records its timestamp. The transcript and the timestamps are required by the DataProcessingSubsystem. The timestamps are also shared with the Word

Recommendation Model (which utilizes BERT [2] module) within the same subsystem. The Word Recommendation Model detects the gaps in the presentation and when the gap is larger than a certain threshold; it starts generating words to fill in the gaps by looking at the context of the sentence.

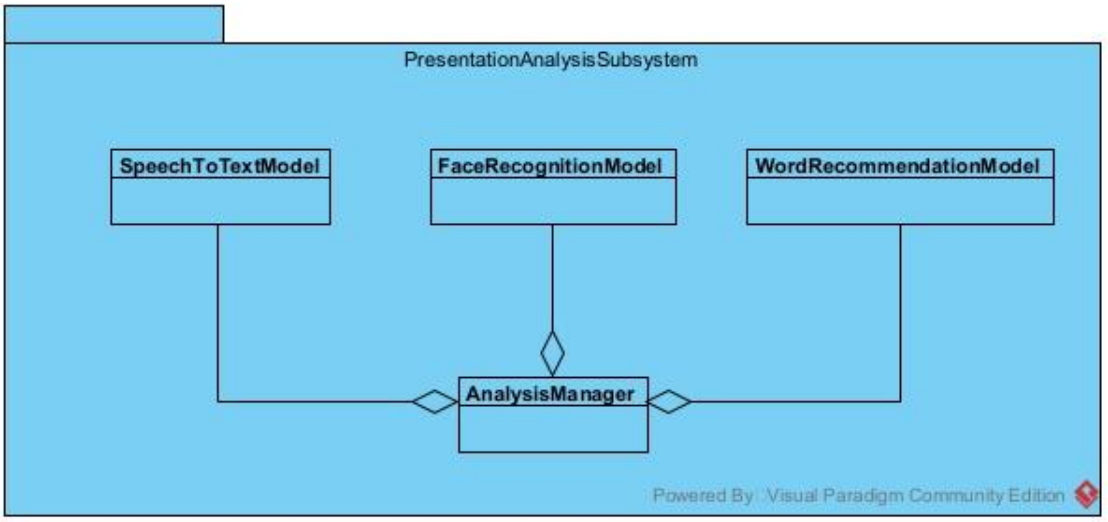

Figure 5. Presentation Analysis Subsystem Service

3.2.2 DataProcessingSubsystem

This subsystem is responsible for the processing of the data that is passed from other subsystems (VisualAnalysisSubsystem, SpeechAnalysisSubsystem). It prepares various statistics and transcripts to be stored at the database through the UserDataManagementSubsystem. It also delivers the processed data to the ReportsAndProgressSubsystem by requesting it from the UserDataManagementSubsystem.

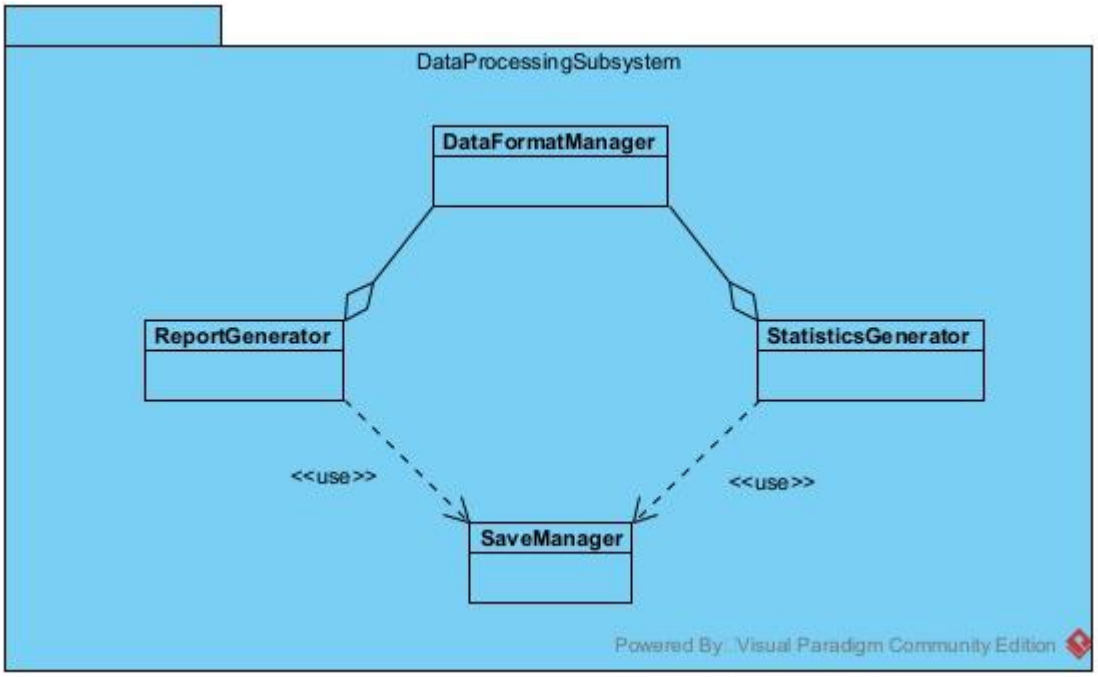

Figure 6. Data Processing Subsystem Service

3.3 Data Layer

3.3.1 UserDataManagementSubsystem

This subsystem is responsible for the communication with the database. Its responsibilities include the retrieve, add, update and delete operations for users and presentations. Since 3-layered architecture is applied, all communication between Presentation Layer and Data Layer is going through the Logic layer. This suggests that, as it can be seen from the Subsystem Decomposition diagram, only the Logic Layer communicates with the Data Layer. It is also responsible for delivering data to DataProcessingSubsystem to be eventually delivered to the user interface.

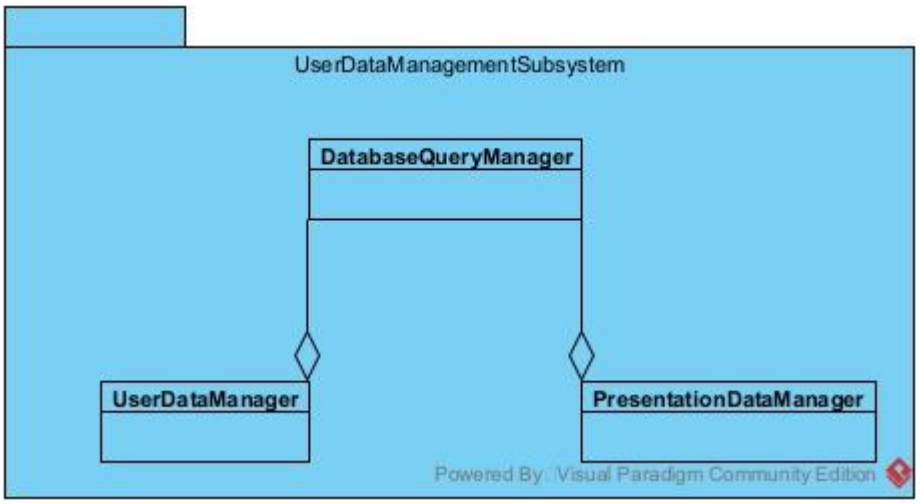

Figure 7. User Data Management Subsystem Service

4 Consideration of Various Factors In Engineering Design

Our project, its analysis and its design are affected by various factors like public health factors, cultural factors, global factors and social factors. Similarly, our project is to affect those areas as well.

With regards to public health, Prexcel is expected to have a significant positive impact, especially during the pandemic conditions. Prexcel is a tool that is optimized and geared towards online presentations, and in particular, it improves the users' online presentation abilities and enhances their online presentations with the live feedback features. The increased effectiveness of online presentations may decrease the need for face to face presentations, and help improve public health during the pandemic in this way.

With regards to cultural factors, again, Prexcel has significant positive effects. Today's working culture is swiftly shifting towards remote/online work. This is especially true for software development companies, which usually have a unique culture where working remotely is more accepted and is significantly more prevalent. Prexcel has the potential to significantly improve remote working culture by improving how well people can prepare for and deliver remote online presentations, which makes remote work more viable and desirable from the companies' perspective.

With regards to global factors, education is one of the most important global efforts. Prexcel is a great complimentary educational tool that could help students improve their presentation skills. Furthermore, the word recommendation feature of the live feedback feature of Prexcel could be very beneficial for students whose native language is not English, during their presentations, if they are receiving an education in English.

These three areas, which are public health, culture (in particular remote/online working culture) and global factors and efforts (in particular education efforts) also significantly shaped our conceivement, analysis and design of our project. First of all, consideration of these 3 areas have shown us the target audience of our system, which are mainly business and educational users who make online presentations. This has led us to shape our analysis and design in a way that caters to this particular audiences, by adding functionalities such as progress tracking, which is very helpful for educational purposes, and exporting reports in pdf, which is again, a key selling feature for both business and educational purposes, to increase its utility for these audiences. Moreover, robustness and reliability are must to have

qualities of a system for it to be able to confidently be used in a business environment for business purposes. Furthermore, since these audiences may come from a wide variety of backgrounds, it is crucial that the system has high usability and people from these different backgrounds can use the system intuitively and with ease.

With regards to social factors, establishing effective communication and expressing one's ideas is extremely crucial. With this in mind, Prexcel enhances the user's ability to express themselves in a better way by giving them directions and statistics on how they can improve. Therefore, while giving online presentations the presenters are also going to develop themselves in their social relationships.

With regards to economic factors, effective communication and presentation skills are a very important skill in life that many people struggle with, and in general, many people spend a lot of money on various effective speech & presentation courses and resources. Prexcel is a tool that will help people improve those skills for free, saving many people a considerable amount of money. In addition to this, the above mentioned effects on remote work also have a significant positive economic impact.

With regards to environmental factors, as also previously mentioned, Prexcel makes remote work more feasible. Making remote work more feasible would, in time, stop migration to big cities & allow people working in smaller towns & villages, and help stop the excess crowding in cities. This would mean less time spent commuting, consequently less pollution and a net positive impact on the environment.

The latter three factors, that are the social factors, economic factors and environmental factors, all enhance the importance of a system/application like Prexcel for its target audience, which just further cements significance of Prexcel's designs goals such as utility and usability, so that these target audiences can use the application for their needs effectively and easily.

The project does not have a significance with regards to public safety and public welfare.

Table 1: Factors that can affect analysis and design.

	Effect level	Effect
Public health	7/10	Prexcel is geared towards online presentations. It helps the users improve their online presentation skills, as well as

		enhancing their online presentations by live feedback features. The increased effectiveness of online presentations may decrease the need for face to face presentations, and help improve public health during the pandemic.
Public safety	0/10	None.
Public welfare	0/10	None.
Global factors	8/10	Education is one of the most important global efforts. Prexcel could greatly help students improve their presentation skills as a complementary tool. Therefore, Prexcel is designed in a way that caters to educational users' needs.
Cultural factors	5/10	Many companies, especially software development companies, are shifting their working culture to include working online, remotely. Prexcel helps people develop better online presentation skills, making remote working more feasible and desirable, making a cultural impact in this aspect as well.
Social factors	6/10	Prexcel allows the users to explain their ideas in presentations better, thus elevating their social skills.
Economical factors	9/10	Prexcel would help people develop their presentation skills without attending paid courses, helping many people save money. Furthermore, promoting remote work would also have a positive impact in this regard.
Environmental factors	5/10	Prexcel would have a positive impact on the environment, as it makes remote work more feasible, and makes people spend less time commuting. (Hence less pollution.)

5 Teamwork Details

5.1 Contributing and functioning effectively on the team

It is important that we, as a team, understand the responsibilities that come with the territory of working on a group project. It is important that every member of the project is aware of at all times what the other members' responsibilities are and the work being done. This is so that no confusion exists during the development of the project regarding what piece of work must be done next, allowing for effective use of every member's time.

In light of this, each of us have contributed in the following way:

Can Kırşallıoba: Started the implementation of the word recommendation module, and tested it with some of the inputs that the system might encounter in later stages. The optimization part of the model is also underway, however it is still a working progress. The system is also currently compatible with the input that will come from the DeepSpeech module, with the exception of the masking functions for the gaps in the presentation. Also, prepared the database query functions for the Presentation table.

Can Kırımca: Mainly responsible for speech-to-text and word recommendation features with Can Kırşallıoba. Started the implementation of the speech-to-text model using DeepSpeech. Created a MySQL database on Amazon AWS in order to store persistent data. Implemented the initial database connection component that performs data storage and retrieval.

Burak Yiğit Uslu: He took chief responsibility in the development of the user interfaces and development of the logic implemented in Javascript. Furthermore, with Can Kırımca and Can Kırşallıoba, he took role in the integration of the UI with the database for user accounts (logging in/signing in functionality.) He created and configured a custom React + Electron boilerplate for the project and approximately half of the UI screens. Also, he created the routing algorithm/function with Barış.

Alper Sarı: Worked on the back end implementation of the recording systems as well as the live feedback systems for both visual and word selection feedback systems.

Barış Tiftik: Implemented half of the UI screens. Created the routing algorithms with Burak.

5.2 Helping creating a collaborative and inclusive environment

We paid great mind to making sure that lines of communication between members of the team were open and clear. It is important that we are aware of each other's shortcomings to be able to work together without any friction between team members.

In particular, each have contributed in the following way towards this goal:

Can Kırşallıoba: Collaborated with Burak to better understand the design of the user interface, which led to developing the word recommendation module in a way that is more compatible with the user interface design. Also collaborated with Can Kırımca on the output of the DeepSpeech model. Collaborated and shared knowledge with the entire team to figure out the details of the integration of the backend processes with the frontend. Lastly, worked with Can Kırımca to design subsystem decomposition and services.

Can Kırımca: Collaborated with Can Kışallıoba for the research and initial implementation of word recommendation and speech-to-text models. Also worked with Can Kırşallıoba to design subsystem decomposition and services.

Burak Yiğit Uslu: Collaborated with Can Kırşallıoba and Can Kırımca for the design and subsystem decomposition of the project for the parts that include UI elements and logic implemented in Javascript. Collaborated with Barış for the implementation of the user interface. Collaborated and shared knowledge with the entire team to figure out the integration of the backend processes with the frontend.

Alper Sarı: Collaborated closely on the integration of back end python scripts with the UI system as well as the ML scripts that are used for presentation analysis and

live feedback. Closely worked with Burak Uslu and Can Kırımca as well as Can Kırşallıoba on the aforementioned matters.

Barış Tiftik: Shared the knowledge and collaborated with Burak Yiğit Uslu for the UI design and implementation. Worked together with him to create the necessary UI screens.

5.3 Taking lead role and sharing leadership on the team

As the project has been broken down into work packages of differing sizes and complexities, we have taken care to divide the tasks to the appropriate members while regarding both their technical skills as well as leadership skills. While it is straightforward to assign work packages by order of experience in the subject of the package, we decided to share leadership of packages mostly equally so that every member of the team could hold some responsibility over parts of the project.

In particular, each of us have contributed in the following way:

Can Kırşallıoba: Shared the leadership position for the Word Recommendation and Speech to Text modules with Can Kırımca. And started the implementation of the Word Recommendation module. Collaborated with Burak Yiğit Uslu on the connection aspect of the user interface (for issues such as how the live feedback will be given when the user is presenting his/her presentation). Also collaborated with Can Kırımca on how the DeepSpeech module gives its output (the format and the type of the data it gives is very important as it will be directly used for the input to the Word Recommendation model).

Can Kırımca: Shared the leadership position for the Word Recommendation and Speech-to-Text models with Can Kırşallıoba. Also took the leadership position for the database integration of the project with Amazon AWS server.

Burak Yiğit Uslu: He took the lead role in the development of user interface and logic implemented in Javascript. He delegated responsibilities to the team members for the parts that relate to the development of the UI or integration of the backend parts to the UI, and followed the leadership of the other team members when working in/contributing to the areas of the project that are under the leadership of other team

members, mainly backend parts (especially when modifying them to integrate with the UI) and thus shared leadership for this process.

Alper: Took a lead role regarding the back end processes of gathering user input and its integration to both the UI of the application as well as the machine learning modules. It was deferred to Burak's leadership of the UI front where they worked to find out how it could be integrated and call the python scripts necessary for user audio and video input. On the other hand work with Can Kırımca was done with regards to how we would pass data between user input during live presentation and the ML models.

Barış Tiftik: Took initiative and autonomy for the development of parts of the UI screens assigned to me. Followed the leadership of the team member responsible for the development of the UI for the general integration of the UI and in order to create a compatible UI.

6 Glossary

DeepSpeech: An open source machine learning algorithm developed by Mozilla that is used for converting speech to text.

Electron: A Javascript library that is used for creating web-based apps on the desktop platform.

MySQL: An open-source relational database management system.

OpenCV: An open source computer vision library.

React: A Javascript framework developed by Facebook. It is used for creating web apps for browsers or mobile platforms (using React native). Within the context of the project, it will be used alongside Electron to create a desktop app.

7 References

[1] AWS. https://aws.amazon.com/, “Amazon Web Services”, [Accessed: Dec. 15, 2021].

[2] Bert, https://huggingface.co/docs/transformers/model_doc/bert, “BERT”, [Accessed: Dec. 21, 2021].

[3] OpenCV, https://opencv.org, “OpenCV”, [Accessed: Oct. 8, 2021].

[4] MySQL, https://www.mysql.com/, “MySQL, ”[Accessed: Nov. 14, 2021].

[5] JavaScript, https://www.javascript.com/, [Accessed: Nov. 14, 2021].

[6] React, https://reactjs.org/, “React- A Javascript library for building user interfaces”, [Accessed: Nov. 14, 2021].

[7] Electron, https://www.electronjs.org, “Electron | Build cross-platform desktop apps with JavaScript, HTML, and CSS” [Accessed: Oct. 10, 2021].

[8] DeepSpeech, https://deepspeech.readthedocs.io/en/r0.9/#, [Accessed: Oct. 8, 2021].