UTM Computing Proceedings
Innovations in Computing Technology and Applications
Volume: 6 | Year: 2024 | ISBN No. : 2976-2278

Malaysian Sign Language Real Time Tutorial

Meor Adib Zakwan bin Meor Ahmad Fauzi and Yusliza Binti Yusoff
Faculty of Computing
Universiti Teknologi Malaysia
Johor Bahru, Malaysia
meoradibzakwan@graduate.utm.my, yulisza@utm.my

***Abstract*— A baby cannot choose to be physically perfect. Some babies are born deaf and unable to speak, which is a tragic occurrence. These individuals face challenges in their future lives as they struggle to communicate with the general public. To bridge the communication gap between those with hearing and speech impairments and others, an educational website called the Malaysian Sign Language Real Time Tutorial (MASRETT) will be developed. MASRETT will provide tutorials for learning Malaysian Sign Language (MSL), helping learners grasp the basics of signing. The chosen methodology for developing this system is agile software development, which allows for identifying and resolving errors through an iterative process. The thesis's outcomes include a thorough review of the system's requirement analysis and design. Ultimately, the Malaysian Sign Language Real Time Tutorial will increase the number of MSL students in Malaysia, while also eliminating communication barriers between individuals with and without speech and hearing impairments, leading to stronger social relations in Malaysia.**

Keywords-component; Machine Learning, Artificial Intelligence, Real Time Tutorial, image detection.

I. INTRODUCTION

Some newborns are permanently deaf and mute due to this condition. Knowing that these impairments cannot be changed is heartbreaking. In Malaysia, there are around 40,743 registered persons with hearing impairments [1]. Problems with communication with the general public give a difficult environment for people who have speech and hearing impairments. Communication between people with disabilities and those without is made more difficult by a lack of awareness and understanding of sign language. The difficulty of learning a new language, such as sign language, has resulted to a decrease in its use. Additionally, since sign language classes are frequently fee-based, those who cannot afford them, such as the deaf, the mute, and even those without disabilities are prevented from taking them.

An educational website called MASRETT is a proposed system that makes it easier for sign language learners to interact with the site. It analyses the accuracy of the learners' hand gestures in relation to Malaysian Sign Language (MSL) using the webcams on their computers. The website offers MSL tutorials that allow beginners to learn and practice, complete with diagrams and descriptions of various signs. To practice the sign poses shown in the tutorials, students can use their own webcams.

A. Problem Background

Effective communication is restricted by the communication gap between speech and hearing-impaired people and non-impaired people. Sadly, many non-impaired people and even some deaf or mute people choose not to attend Malaysian Sign Language (MSL) classes. As people rely on social connections for their overall health, this expanding barrier will have a negative impact on social interactions within Malaysia. To address this, all communities should be required to learn Malaysian Sign Language, and using MASRETT, an educational website, can make learning MSL simple. The goal of MASRETT is to fill the communication gap between those who are speech and hearing-impaired and those who lack by teaching appropriate MSL poses.

B. Project Objectives

The objectives of the project are:

- To identify existing approaches in learning MSL for learners.
- To develop a website that helps learners in detecting the MSL with its corresponding meanings by programming a sign language detector using transfer learning and Tensorflow API.
- To test and evaluate the MASRETT for accuracy and effectiveness.

II. LITERATURE REVIEW

C. Current System Analysis

Three current systems will be examined in this section which are SignAll Online, SLAIT, and SignSchool. These systems' features and functions will be compared with the features of the suggested system. SignAll Online is an AI-powered app that translates sign language into spoken English. It uses cameras and advanced algorithms to recognize and convert sign language gestures. Next, SLAIT is a mobile app that acts as a real-time sign language interpreter. It utilizes MediaPipe and custom neural networks to recognize American Sign Language (ASL). The app enables communication between signers and non-signers through video calls only. Lastly, SignSchool is an educational app that helps learners master ASL. It offers video-based learning resources with teachers explaining sign meanings. The app provides an

interactive learning experience with activities, lectures, and reviews.

D. Comparison between Existing systems

MASRETT makes use of the learner's camera for signing exercises, allowing students to engage with the program. SignAll Online has similar feature as well, but it also incorporates webcam use for video conversation. Webcams are not used in the classroom at SLAIT or SignSchool. Furthermore, none of the current systems particularly support Malaysian Sign Language (MSL), instead focusing on ASL. Table I. shows the comparison between MASRETT and reviewed systems.

TABLE I. COMPARISON BETWEEN SYSTEMS

Features	SignAll Online	SLAIT	SignSchool	Proposed System
Learner's webcam usage	Yes	Yes	No	Yes
Malaysian Sign Language	No	No	No	Yes
Sign Language exercises	Yes	No	Yes	Yes
Payment service	Paid	Free	Free	Free
Machine learning	Yes	Yes	No	Yes

III. SOFTWARE DEVELOPMENT METHODOLOGY AND TECHNOLOGY SELECTION FOR MASRETT

It is critical for the effective deployment of MASRETT to choose a suitable system development approach. The creation process may be shortened by adopting a disciplined strategy and addressing the goals of learners and the system owner, resulting in an efficient instructional website for learning Malaysian Sign Language. This chapter lays the groundwork for the succeeding phases of development, ensuring that MASRETT satisfies the goals of accessibility, usability, and operational efficiency.

E. Agile Software Methodology

By implementing the Agile software development methodology, the MASRETT project benefits from a systematic and flexible approach to effectively manage system development [2]. The iterative nature of Agile which focuses on continuous improvement align well with the goals of MASRETT which allows to adapt to changing needs and deliver an efficient educational website for learning MSL. Fig.1 shows the phases of Agile used for the project.

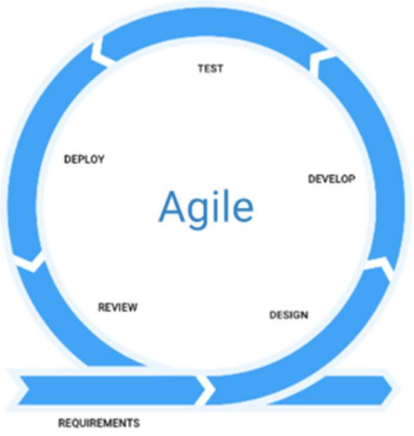


Figure 1. Phases of agile

F. System implementation

1) Design

The MASRETT project's design phase entails developing initial specifications and visualizing the system's design using the Draw.io and Canva tools. The project team may utilize these tools to record functional requirements using UML class diagrams, design system functionalities, and construct aesthetically appealing user interfaces. This strategy improves communication, makes requirement validation easier, and enables the creation of an intuitive and user-friendly instructional website for learning Malaysian Sign Language.

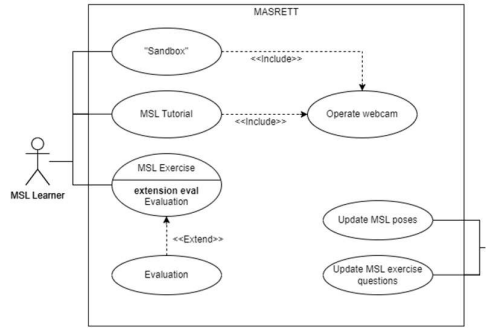
2) Development Requirements

The MASRETT project's development phase includes the installation of the React.js framework and Jupyter Notebook. React.js allows for the building of interactive user interfaces and backend web applications, whilst Jupyter Notebook allows for the integration of machine learning and deep learning capabilities. The MASRETT system may provide a complete and engaging learning platform for Malaysian Sign Language by using various technologies, improving accessibility and communication for those with hearing and speech impairments.

IV. PROPOSED SYSTEM

The requirements analysis and design features of the proposed system are discussed in further detail in this chapter. Requirement analysis is a collection of processes that assist clarify users' expectations and drive application development and revision [3]. Project design also includes essential features, project organization, success criteria, and significant deliverables. Data design is concerned with defining the essential data items and their connections, whereas interface design is concerned with the visual arrangement of the system's user-interactable parts.

G. Use Case Diagram



a)

Figure 2. System use case diagram

The use case diagram features two actors: the MSL learner and the administrator. In MASRETT, the MSL learner has access to three functionalities. They may practice signing in front of their webcam using the Sandbox option. The MSL Tutorial tool teaches and practices certain poses using the learner's camera. The MSL Exercise tool allows students to put their knowledge to the test by answering questions given by the programmed. The Evaluation tool, which is an extension of MSL Exercise, allows students to double-check their responses. The use case Operate Webcam is present in all three functions. The admin is in charge of keeping MSL poses and exercise questions up to date for the learners.

H. Sequence diagrams for MSL Learners in MASRETT

The sequence diagrams in Fig. 3 demonstrate the fundamental characteristics of MASRETT for MSL learners. The sandbox diagram represents the practicing environment, the MSL tutorial diagram shows how to obtain training materials, and the MSL exercise graphic focuses skill assessment. These examples demonstrate MASRETT's user-centered approach to increasing MSL learning and proficiency.

The MASRETT admins' sequence diagrams show in Fig. 4 illustrates their involvement in updating MSL handshapes and explanations for the MSL Tutorial function, as well as uploading MSL exercises for MSL learners in the MSL Exercise feature. These illustrations emphasize the critical role of administrators in ensuring accurate and up-to-date instructional materials and delivering relevant and challenging tasks to improve MSL learners' learning experiences. The administrators are responsible for maintaining and developing the instructional content and features of MASRETT for the benefit of MSL students.

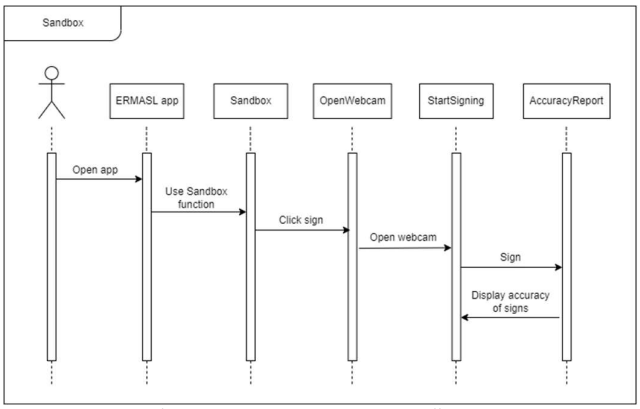


Figure 3. MSL Learners sequence diagram

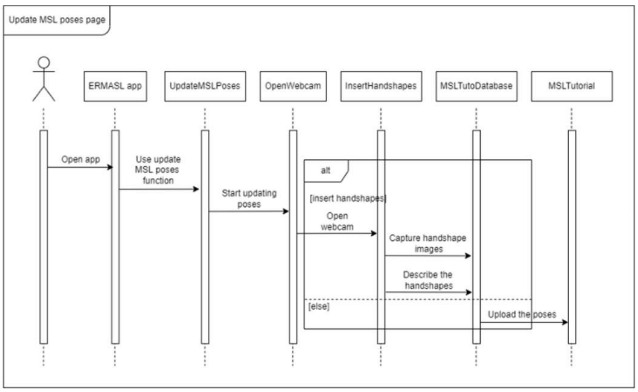


Figure 4. MASRETT Admin sequence diagram

I. System Architecture Design

MASRETT, a web application, has a system architectural design that is illustrated in the Fig. 5 below. By inputting data through the back-ends, the administrator can change the software's content. The data is sent to the Jupyter notebook server via the Application Programming Interface (API), which comprises the file system and Jupyter server. MSL students may access MASRETT material by requesting it using the front-end API, which talks with the web server. The students will then get and view the results.

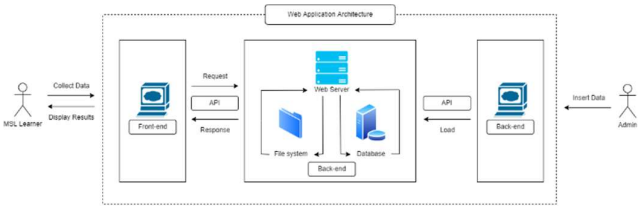


Figure 5. Web Application architecture design

J. MASRETT Interface Design

On this website, students may learn any MSL pose they like. After clicking any button, a picture will be presented that corresponds to the MSL pose that was previously clicked.

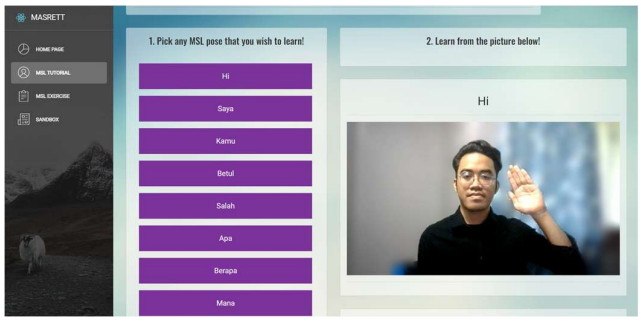


Figure 6.MSL Tutorial Interface

After learning the sign, the user will be requested to sign according to the stance using a webcam stream. Fig. 6 shows the MSL Tutorial interface with the camera feed. After finishing, students can attempt clicking to another stance to try to sign the new ones.

K. AI Model Training

An AI model for real-time object identification via the camera interface is required for system development. This AI model is embedded into the programmed to recognize learners' MSL poses. Furthermore, the program serves as a teaching website where MSL students may learn new poses and evaluate their practises.

This project collects 12 distinct MSL signs such as 'Hi', 'Saya', 'Kamu', 'Betul', 'Salah', 'Apa', 'Berapa', 'Mana', 'Boleh', 'Tidak boleh', 'Minta maaf', and 'Terima kasih'.

1) Image Collection

To achieve reliable recognition, a collection of 20 distinct angles of the same pose was obtained before teaching the AI to recognize various MSL poses. The LabelImg API was used in this procedure, as shown in Fig. 7. Jupyter Notebook used the LabelImg API to label the pictures obtained for training purposes. This thorough labelling method maintains precision and increases the AI model's efficacy in recognizing certain MSL indicators.

Figure 7. Training Image Collection

2) AI training

Once the images are labeled, the AI is ready for additional training. For each class, 18 out of the 20 collected MSL pose images will be used to train the AI model, while the remaining two will be used for testing. To improve detection accuracy, the images will undergo 10,000 training steps. Fig. 8 showcases the Jupyter Notebook command used to train the images over the specified number of steps.

```
In [9]: TRAINING_SCRIPT = os.path.join(paths['APIMODEL_PATH'], 'research', 'object_detection', 'model_main_tf2.py')

In [10]: command = "python {} --model_dir={} --pipeline_config_path={} --num_train_steps=10000".format(TRAINING_SCRIPT,
             paths['CHECKPOINT_PATH'],files['PIPELINE_CONFIG'])

In [11]: print(command)

         python Tensorflow\models\research\object_detection\model_main_tf2.py --model_dir=Tensorflow\workspace\models\my_ssd_mobnet_tune
         d1 --pipeline_config_path=Tensorflow\workspace\models\my_ssd_mobnet_tuned1\pipeline.config --num_train_steps=10000

In [ ]: !{command}
```

Figure 8. 10,000 training steps

3) Detection testing

To confirm the trained AI model's detection capabilities, picture testing is performed to check its ability to recognize and categories MSL stances appropriately. In the Jupyter Notebook, Fig. 9 , depicts the detection test for the "Boleh" stance. The AI model finds the specific symbol with a 95% confidence measure. This shows that the model is capable of reliably recognizing and detecting the MSL stance, resulting in a dependable and effective detection method.

```
In [28]: IMAGE_PATH = os.path.join(paths['IMAGE_PATH'], 'test', 'boleh.74937e3d-f5f1-11ed-9cd2-991f1eb50620.jpg')

In [29]: img = cv2.imread(IMAGE_PATH)
         image_np = np.array(img)

         input_tensor = tf.convert_to_tensor(np.expand_dims(image_np, 0), dtype=tf.float32)
         detections = detect_fn(input_tensor)

         num_detections = int(detections.pop('num_detections'))
         detections = {key: value[0, :num_detections].numpy()
                       for key, value in detections.items()}
         detections['num_detections'] = num_detections

         # detection_classes should be ints.
         detections['detection_classes'] = detections['detection_classes'].astype(np.int64)

         label_id_offset = 1
         image_np_with_detections = image_np.copy()

         viz_utils.visualize_boxes_and_labels_on_image_array(
                     image_np_with_detections,
                     detections['detection_boxes'],
                     detections['detection_classes']+label_id_offset,
                     detections['detection_scores'],
                     category_index,
                     use_normalized_coordinates=True,
                     max_boxes_to_draw=5,
                     min_score_thresh=.8,
                     agnostic_mode=False)

         plt.imshow(cv2.cvtColor(image_np_with_detections, cv2.COLOR_BGR2RGB))
         plt.show()
```

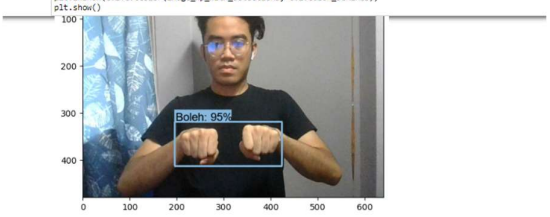


Figure 9. Detection testing

L. Streamlined User Interaction and Interface Efficiency

Learners can select any of the MSL stances available on the MSL Tutorial. After picking a posture as in Fig. 10, an image related to the MSL pose will be provided for the learners to study.

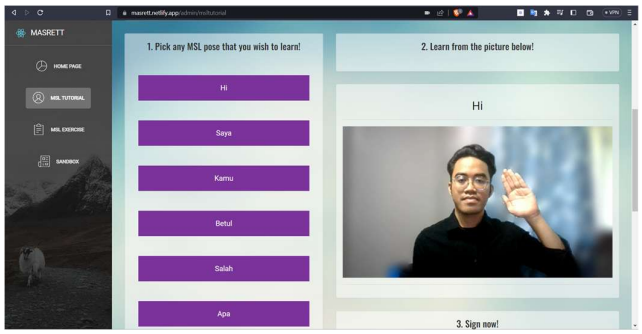


Figure 10 MASRETT pose learner

In the MSL Exercise, learners will be presented with multiple-choice questions to test their knowledge. As shown in Fig. 11 , they will be shown images and asked to select the correct sign. When the evaluate button is clicked, the answer will be assessed to determine if it is correct or incorrect.

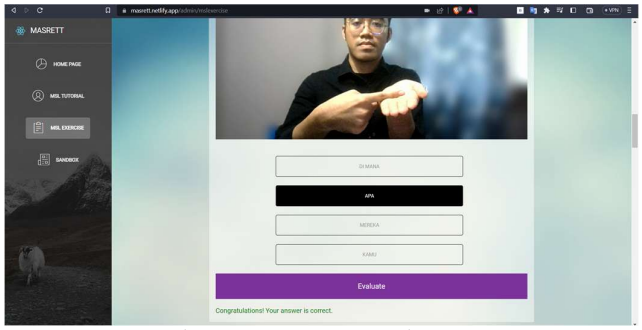


Figure 11. MASRETT exercise page

Next, the Sandbox function in MASRETT allows MSL students to freely practice signing MSL postures. Fig. 12 illustrates a webcam stream that allows learners to view their signing while attempting to reproduce the MSL stances offered by the program. This engaging and hands-on method promotes practical experience and improves participants' MSL signing ability.

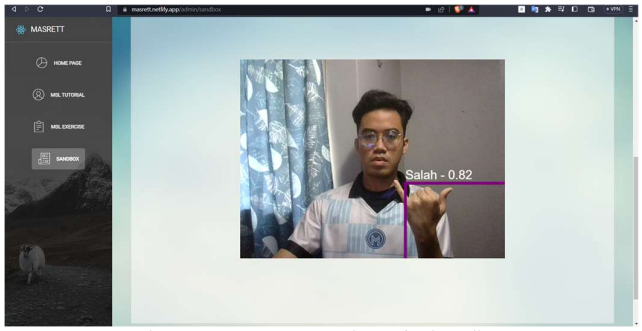


Figure 12. MASRETT webcam feed sandbox

M. Networking implementation

Several APIs are used to speed up the building of the AI model. The TensorFlow API was chosen because of its broad architecture, which is ideal for developing object identification models. The LabelImg API is a great tool for easily producing labelled datasets, which are required for training correct machine learning models. The TensorBoard API integration provides visualization tools that allow users to acquire important insights into their object recognition models' training process, model structure, and performance. These APIs add to the project's overall effectiveness by easing the development process.

N. Testing

This section explains the testing methodology used to assess the application's usability. The main goal is to examine the system's flow and identify any flaws or errors. Black Box testing and user testing are employed to uncover and resolve potential issues during the development process. Each approach is discussed in dedicated sections for a comprehensive understanding.

1) Black Box Testing

Table II displays the results of Black Box testing for the webcam interface and its ability to detect MSL poses. The testing involved performing each pose 10 times and assessing the accuracy of the AI's detections. While some poses were detected successfully, others were not, resulting in varying success rates. Despite extensive training and evaluation, the AI still exhibits errors in detecting certain poses and generating label maps.

TABLE II. MSL POSE DETECTION

No.	MSL Pose	Expected Label Map color output	Pass/Failed	Success Rate
1	Hi	Red	Pass	0.97
2	Saya	Cyan	Failed	0.0
3	Kamu	Lime	Failed	0.0
4	Betul	Brown	Failed	0.0
5	Salah	Purple	Pass	0.87
6	Apa	Orange	Failed	0.0
7	Berapa	Green	Failed	0.0
8	Mana	Pink	Failed	0.0
9	Boleh	Blue	Pass	0.93
10	Tidak boleh	Black	Failed	0.0
11	Minta maaf	Gray	Failed	0.0
12	Terima kasih	Yellow	Pass	0.94

System flow testing is conducted to verify the smooth operation of the application flow. Table III showcases the execution of Black Box testing for interface functions, particularly focusing on the MSL Tutorial page, MSL Exercise page, and Sandbox page. The primary goal is to ensure seamless interactions during the testing process.

TABLE III. INTERFACE FUNCTIONS

No.	Interface	Interactions	Output	Result
1	MSL Tutorial	12 MSL pose buttons	Displays a corresponding image to the pose	Success
		Webcam interface	Displays a webcam interface for detections	Success
2	MSL Exercise	Selecting an answer for a question	The selected answer is highlighted	Success
		Clicking evaluate button after choosing a wrong answer	Displays the selected answer is wrong and the proper answer	Success
		Clicking evaluate button after choosing a correct answer	Displays a congratulating text for choosing the right answer	Success
3	Sandbox	Webcam interface	Displays a webcam interface for detections	Success

2) User Acceptance Test case summary

Table IV presents a summary of the test cases and functions executed by the user, providing a comprehensive overview of their testing activities.

TABLE IV. TEST CASES SUMMARY

Test No.	Description	Results	
		Accepted	Not Accepted
1	1. Displaying MSL pose images 2. Displaying the name of the MSL pose 3. Signing in the webcam interface 4. Label map is drawn in the webcam interface 5. Label map is drawn correctly corresponding to the MSL pose		/ (All functions are behaving as expected except for some of the label maps are drawn incorrectly correspondin g to the MSL pose)
2	1. Selecting an answer in any questions 2. Clicking the evaluate button for a wrong answer 3. Clicking the evaluate button for a correct answer	/	
3	1. Signing in the webcam interface 2. Label map is drawn in the webcam interface 3. Label map is drawn correctly corresponding to the MSL pose		/ (Some of the label maps are drawn incorrectly correspondin g to the MSL pose)

V. CONCLUSION

MASRETT aims to increase the number of MSL learners in Malaysia to improve communication and foster inclusivity. The MASRETT website facilitates efficient learning of MSL and its interpretations through the use of machine learning and deep learning techniques. The goal is to reduce communication barriers and promote social connections across the country. The MSL Tutorial feature presents MSL pose images to help learners master accurate signing practices, while the MSL Exercise tool tests their knowledge through multiple-choice questions. The webcam tool allows students to practice signing postures and receive accuracy scores from the AI model. Despite minor flaws in the AI model's detections, the overall functionality of the program is effective. Future developments should focus on comprehensive research, data collection, and addressing any weaknesses in the AI model through thoughtful decision-making.

ACKNOWLEDGMENT

We would like to express our sincere gratitude to Faculty of Computing and Universiti Teknologi Malaysia (UTM) for their invaluable support and resources, which played a crucial role in the successful research.

REFERENCES

[1] Muthiah, W. (2022, March 3). *KJ: Over 40,000 hearing-impaired people registered in Malaysia*. The Star. Retrieved from https://www.thestar.com.my/news/nation/2022/03/03/kj-over-40000-hearing-impaired-people-registered-in-malaysia

[2] What is Agile Methodology in project management? Wrike. (n.d.). Retrieved from https://www.wrike.com/project-management-guide/faq/what-is-agile-methodology-in-project-management/#:~:text=The%20Agile%20methodology%20is%20a,planning%2C%20executing%2C%20and%20evaluating

[3] Requirements analysis - understand its process & techniques. ReQtest. (2020, September 9). from https://reqtest.com/requirements-blog/requirements-analysis