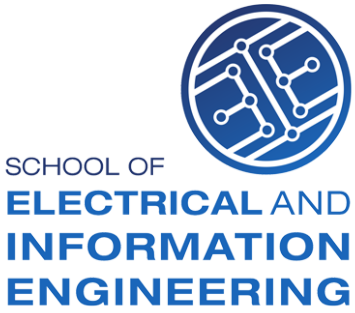


Faculty of Engineering and the Built Environment
University of the Witwatersrand, Johannesburg, South Africa

Deepfake Hunter: An AI-enabled Software Platform for Detecting Deepfakes on the Internet

ELEN4011A - Information Engineering Design II

Robin Jonker
Student Number: 1827572
Supervisor: Prof. T. Celik

9 September 2022

Abstract: The rise of deepfake technology and the threats they present is a challenge for society. Possibly malicious media files getting spread across social media, which is practically impossible for humans to detect, exposes society to mass misinformation. The purpose of this project is to design an AI-enabled software platform for detecting deepfakes on the Internet. This platform is designed to be a user-friendly free-to-use tool that allows anyone to detect the authenticity of media files. Driving principles in the design relates to its social, economic, and environmental impacts along with ensuring its scalability and maintainability. A cloud computing solution is selected using the powerful Amazon Web Services (AWS) ecosystem of technologies. AWS's services ensure the platform and the data it receives is extremely secure. A full-stack platform with a Python backend and React frontend along with defined testing outcomes is designed. Deep learning methods such as convolutional neural networks with the specific EfficientNet architecture and the S3FD detector for image and video detection along with the WaveFake architecture for audio detection are selected as the optimal models to process the media files using the AWS SageMaker service. This platform has a net zero carbon policy with distinct processing reduction features. The platform with its open-source technologies and income generation strategies focuses on long-term sustainability with crowdsourcing inputs to remove biases within the model creating a sense of hybrid intelligence.

CONTENTS

List of Figures

1. INTRODUCTION

Deepfakes are the manipulation of data in order for them to appear authentic. From basic humour to more malicious impersonations. Creating fake images, videos or audio files that fool people into believing them, has the potential to spread mass misinformation. As a solution to this problem, the Deepfake Hunter website is to be designed. This platform will be a tool for the public to use in order to verify the authenticity of files which will greatly benefit society.

The background research, Section 2, for this project will focus heavily on the Artificial Intelligence (AI) technology that will be used for this platform, this can considered the 'brain' of the software platform and in depth research is done in selecting the best methods available. The creation of the whole system, its user interface, how it will be developed and the exact technologies and architecture that will be used along with why it was chosen, will be described in the design solutions found in Section 5. The exact plan to achieve this design and the requirements of the project will be described in Sections 3 and 4 respectively. The specific use case scenarios and unit tests that drive the design choices will be outlined in Section 6, with the design solutions critically analyzed in Section 7. Appendix A will contain the non-technical report with Appendix B containing the engineering notebook used throughout the project.

2. BACKGROUND RESEARCH

In order to select the best method to detect deepfakes among images, videos or audio files, a high level understanding of AI is needed. AI is effectively machines performing human intelligence that can improve its performance on tasks based on the data it collects. There are multiple different fields, which often overlap, within AI that specialises in specific tasks. Out of the traditional methods, State Vector Machines (SVM) is a method that can solve the problem of detection. SVM maps training data into a hyper-plane that can be linearly separable into specific categories [2]. Out of the traditional methods, SVM using a SIFT and SURF algorithm is the best method for deepfake detection [3]. The notable problem with using SVM for deepfake detection occurs when handling large datasets and the need for manual feature extractions [4]. For smaller datasets SVM trains the model extremely fast in comparison, however training may not work at all on larger scale datasets. Neural networks in particular could solve these problems. De Luca created a great comparison between their differences and use cases [5]. Neural networks have automatic feature extractions and can scale to larger datasets which processes more information. Methods such as multi-threading that modern machinery has made accessible, can largely reduce the training time that neural networks require to a more manageable level [6]. Sharkawy conducted research into the different types of neural networks and their use cases [7]. From this research it is found that the convolutional neural network (CNN) method is the best method for both image and audio recognition. A breakdown of different types of CNN configurations and architectures will be analysed in subsections below. Subsections 2.1.1 to 2.1.3 explain specific neural network methods that can be used in order to detect deepfakes within images and videos, while subsections 2.1.4 and 2.1.5 look at audio detection.

2.1 Neural network encoders

2.1.1 Neural Network Encoder 1 - ResNet: ResNet architecture enables the training of very deep neural networks by performing skipped connection points. This allows the amount of layers in the neural network to exponential increase. As shown in research evaluating the performance of the ResNet model, there is a direct correlation with increased depth and model accuracy [8].

2.1.2 Neural Network Encoder 2 - EfficientNet: EfficientNet applies the same principle as ResNet in its goal of scaling for increased accuracy, except where ResNet and similar methods randomly scale the width, depth, or resolution of the model to increase its layer count, EfficientNet uses scaling compound

coefficients for uniformly ordered scaling which increases the accuracy of the model [1].

2.1.3 Neural Network Encoder 3 - MesoNet: MesoNet is a model based on the Xception architecture that has been optimized for video detection. This method is effective and is a highly accurate model [9].

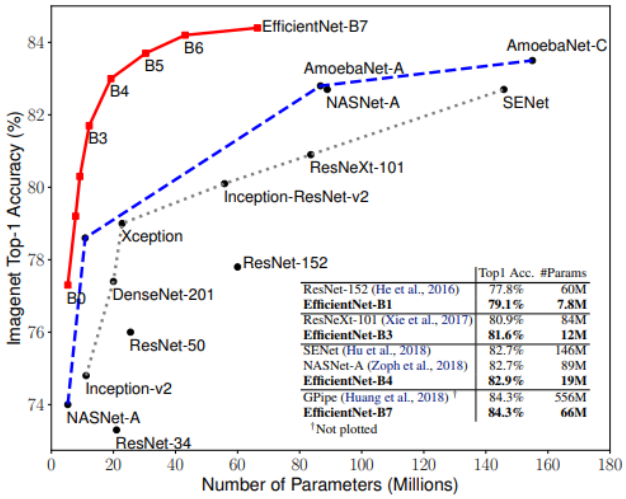


Figure 1: Neural Network Architecture Comparisons [1]

The three different models are all present and tested at a variety of different depths and parameter counts. Figure 1 above shows that the EfficientNet architecture is the most accurate model. For video detection, frame-by-frame detection will occur at 32 frames per video.

2.1.4 Neural Network Encoder 4 - WaveNet: WaveNet is an audio model based on the PixelCNN architecture. It has been optimized for long-range temporal dependencies which is a requirement for raw audio analysis [10].

2.1.5 Neural Network Encoder 5 - WaveFake: WaveFake is a model that is based on the WaveNet architecture that has been optimized for audio deepfake detection. The results showed that the highest average error rate detected was 0.003 which is very good [11].

Due to CNN architecture being very accurate and the fact that WaveFake is an optimized version of WaveNet, and has added training data sets, makes using this model as a baseline to begin with as an optimal choice.

2.2 **Face detectors**

Face detectors need to be used in conjunction with the neural network encoders in order to accurately detect the images or videos. There are two notable detector that are used which will be explain subsections 2.2.1 and 2.2.2 below.

2.2.1 Face Detector 1 - Multitask Cascaded Convolutional Neural Network (MTCNN): The raw image data of faces are presented to the model where initially face detection and face alignment is performed using MTCNN. Once that is complete, facial feature extraction and face recognition is performed using deep convolution neural networks. The advantage that this method has is the fact that the face detection and face alignment occur simultaneously instead of sequentially [12].

2.2.2 Face Detector 2 - Single Shot Scale-invariant Face Detector (S3FD): This detector is based on frameworks such as RPN and SSD which focuses on generic object detection tasks of RPN and the multi-scale mechanism of SSD. The stride size of the anchor points gradually double from 4 to 128 pixels in order to make the detector more robust regardless of face size. The scaling method for anchors and the effective receptive field from the location of anchors, guarantees each scaled anchor to be more precise. The results figures presented in the research relating to multiple different focus areas showcases this detector to be far superior to MTCNN and all other competition, especially in difficult to detect images [13].

There is a notable problem with the MTCNN method. Research into real-life adversarial attacks on the MTCNN method showcases an easily reproducible way to attack it [14]. Therefore the S3FD method will be used.

There was a $1 million competition on deepfake detection hosted on Kaggle by Facebook, Amazon, Microsoft, etc [15]. The winner of this challenged used a similar selection of tools and was a key differentiator in the selection decisions behind the models and methods chosen [16]. He used the MTCNN detector, however he did state that the S3FD method is more precise, robust, and quicker. The performance times for different detectors proves this [17]. The only reason he selected MTCNN instead was due to licensing issues.

2.3 Summary of AI models selected

The EfficientNets neural network encoder model pretrained with ImageNet and Noisy Student is best [18]. In conjunction with the S3FD detector, this is the selected method for image and video recognition with the WaveFake model for audio detection.

3. PLANNING

3.1 Plan to develop a design solution

In order to come up with a well researched and functional design solution, a structured plan is created to ensure the solutions are achieved within the 7 week design period. In summary, the first 5 weeks are used to research 5 key components of the software platform and the final 2 weeks are used to amalgamate the information and research into a concise and clear report, both technical and non-technical reports.

Week 1, Focus: Processing - The fundamental architecture of the system needs to be researched as this will showcase how all components will interact and where and how the Machine Learning model will process the information.

Week 2, Focus: Frontend - The user interface needs to be decided upon and what technologies will be used to create it. It is vital that it is very user friendly. As the intended target is general users and not experts, a basic and simple design will be more effective.

Week 3, Focus: Backend - How is the frontend going to process the user requests. The platform functionality and how the Machine Learning model will be integrated needs to be decided upon.

Week 4, Focus: Database - Where is the information that is received going to be stored. What information will be stored. How does the different data relate to each other.

Week 5, Focus: Security - How is the data that is stored going to be protected. How to prevent attacks on the system and how to ensure that hackers do not effect the results of the system.

Week 6 and 7, Focus: Report writing - Ensure the researched and decided upon solutions are described

in a clear and concise manner. Ensure all the needed figures are included and both the technical and non-technical reports are completed.

3.2 Plan to develop software platform from design

As the requirements for the project are clear and a set plan and idea have been made, the waterfall software development approach will be used in creating the software platform. Once the initial design has been completed, the system will switch to an Agile approach as the market guides the development team into working on specific features or fixing certain bugs.

Due to the research being extensive during the design stage, the actual project implementation time will be reduced. A further 4 weeks will be required to produce an MVP that can be tested, a week for each component with security development throughout. Once the MVP has been created, market testing will occur and new goals and plans for improvement of the platform will be further developed. These include implementing the income generation plan, creating organic marketing campaigns, and requesting expert feedback.

4. PROJECT REQUIREMENTS

The purpose of this project is to design a software platform that the general public can use in order to verify the authenticity of an image or video on the Internet. The platform should be very user-friendly to ensure everyone understands how to use it. The goal behind the platform is to remove misinformation that deepfakes cause, therefore it is of utmost importance that the system itself does not cause misinformation due to erroneous representation. The platform should be secure to prevent hackers from altering its function, causing misinformation to be spread from the platform. Any user information that is collected should be stored in a safe manner to ensure that there are no sensitive data leaks. The platform needs to be easily maintainable and sustainable, factoring in events such as data shifts. The platform also needs to create meaningful ways to reduce its carbon footprint. There is a list of essential design considerations that will be examined in Sections 5.1 to 5.14.

5. DESIGN SOLUTION

The design solution contains all the information regarding the system and how it will be developed. A summary of the design solutions of choice will be outlined whilst some of the essential design considerations and additional features will be broken down in further detail in subsections below.

Cloud computing is the solution of choice for this design. This is preferred over grid computing due to its higher flexibility, pay-as-you-use services, higher scalability and maintainability whilst ensuring far lower downtime and better performance overall [19]. Within cloud computing, there is a couple of high-quality options such as AWS, Azure, and Google. AWS was selected as the optimal cloud service for the design of this platform and will be used as the ecosystem throughout. Google cannot be used as there is no Google Cloud Region based in South Africa, which will violate the PoPI act for user data storage. AWS is the clear market leader in this field. Selecting the most popular technologies opens development up to a greater developer pool. AWS has the largest array of services which enables the platform to develop further without the need for a large migration. AWS has over 80 availability zones whereas Azure has less than 30. This is the key deciding factor between these two services as AWS will have a greater reach with lower latency and fewer downtime [20] [21].

There are 5 large components of the system, namely the frontend, backend, database, processing, and security. Figure 2 showcases the system architecture of the design solution. The frontend will be developed using React, this component is broken down in Section 5.6. The backend will be developed using Python, this component is broken down in Section 5.9 referring to the systems architecture where the connection between all components will also be analysed. This section will also breakdown into

more detail the use of FastAPI to create RESTful endpoints for the backend along with AWS Lambda functions, which will be used throughout, and AWS API Gateway to host the endpoints and create connection points between the frontend and backend. These components will be containerised using Docker which will be analysed in Section 5.11. The database will be developed using AWS DynamoDB with the Machine Learning model processing occurring on AWS SageMaker, both of these components are broken down in Section 5.2. This section will also detail the use of AWS S3 Buckets in order to store the raw data uploaded by users for processing and additionally to host the website. Security aspects will be integrated throughout along with using AWS Virtual Private Cloud (VPC) and AWS Identity and Access Management (IAM), this component is broken down in Section 5.8.

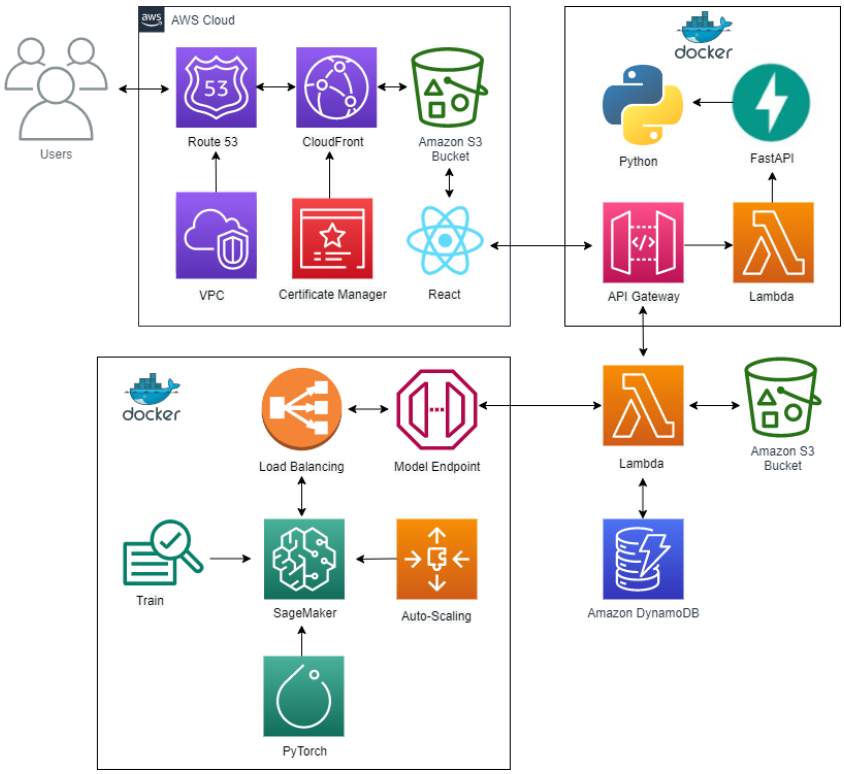


Figure 2: System Architecture

5.1 Hardware requirements

5.1.1 AI processing: The hardware requirements that are needed in order to train the model are 4 GPUs with at least 12 GB of memory [15] [22]. The model will be pre-trained before being uploaded to AWS SageMaker, which will be broken down in Section 5.2. The full training set that will initially be used is approximately 470 GB in size. These are the hardware requirements that were used in creating a model that won 1st place in a million-dollar competition funded by Facebook [16]. As SageMaker is a pay-as-you-use service with multiple levels, the hardware that will be utilised for processing the AI will fluctuate and can adapt processing power quickly. For the most basic version available in the Cape Town region, the hardware used is an ml.t3.medium instance that contains 2 vCPU with 4 GB of memory that is billed at $0.065 per hour. The most advanced version available is an ml.r5.24xlarge instance with 96 vCPU and 768 GB of memory billed at $9.677 per hour [23]. Therefore the hardware requirements for AI processing are easily changeable and scalable. For this design, costs are vital therefore the most basic version will be used. SageMaker has a serverless infrastructure which gets billed per millisecond too. The cost implications will be described further in Section 5.12.

5.1.2 Platform hosting: The platform will be hosted on AWS S3. AWS has a reasonable free tier for S3 hosting that will be used initially for the platform. However, with increased growth and traffic,

the ecosystem is scalable. The free tier allows for 5GB of Amazon S3 storage in the S3 Standard storage class which includes 20,000 GET Requests; 2,000 PUT, COPY, POST, or LIST Requests; and 100 GB of Data Transfer Out each month [24]. This will be enough to host the website and all the processing needs for the development. The use of AWS as a cloud computing solution ensures low downtime due to numerous servers, low costs, and built-in security protocols.

5.2 Data acquisition, storage, and processing

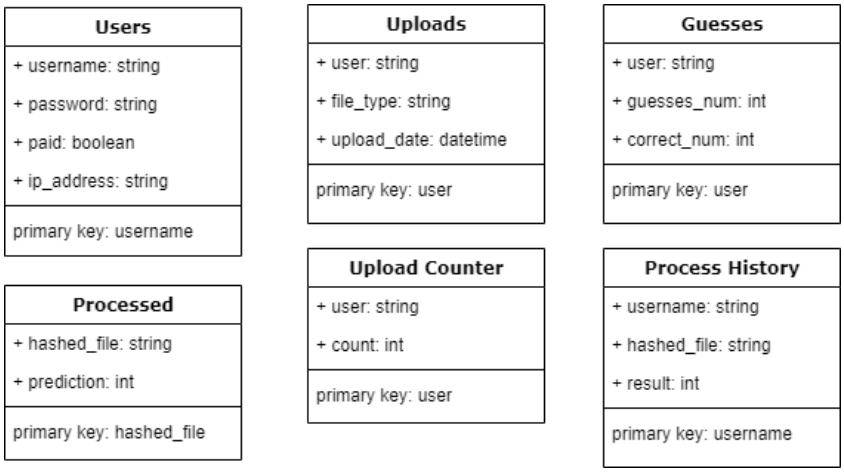


Figure 3: Database Design Schema

Figure 3 above showcases the different NoSQL tables that will be present within the database. 'Users' store the account details. 'Processed' stores the hashed version of uploaded files and their respective results. 'Uploads' tracks when and what type of files users upload. 'Upload Counter' keeps track of how many files each user has uploaded. 'Guesses' keeps track of users' scores in the engagement game. 'Process History' keeps track of all uploads and their results from respective users. All of these tables and where they are used will be elaborated on within this section.

5.2.1 Acquisition - User Data: User data refers to the information provided by the users, such as the email and password when they create a profile. The acquisition of this data will occur via a post form on the frontend which will be described in Section 5.6. Other data that will temporarily be collected such as the IP address of all users that upload on the site will also be acquired from TCP/IP packets [25]. The purpose of this is for security reasons that will be described in Section 5.8. This information will directly be sent to the database for storage.

5.2.2 Acquisition - User Uploads: Users will upload images, videos or audio files to the site. This will be done via a simple form on the frontend. This data will then be transferred using an AWS Lambda function made accessible by an API Gateway. The use of this technology will be described further in Section 5.9. The user uploads will be restricted to files of a maximum of 2 MB. This is to limit the processing time and costs.

5.2.3 Storage - User Data: User data will be stored using a simple and fast NoSQL key-value pair technology offered by AWS called DynamoDB. For database storage there are SQL and NoSQL databases [26]. SQL databases refer to relational tables whereas NoSQL refers to key-value pairs. SQL databases offer more complex functionality and use, however, the reason for the choice of using a NoSQL database is due to its superior simplicity and performance. This database service is also very cost-effective as the serverless method is billed at a pay-as-you-use rate which is very reasonable in comparison to the need for persistent connections and storage allocations found when using a

SQL database. Some disadvantage to using this service is that NoSQL databases do not offer the future flexibility that relational databases offer. Other major advantages such as scalability and maintainability will be further described in Section 5.11 and 5.10. Another important aspect regarding storage is the fact that AWS has a Cape Town location. This ensures the platform abides by the PoPI act.

5.2.4 Storage - User Uploads: As mentioned above, a Lambda function will connect the uploaded data to a RESTful endpoint created by AWS API Gateway and then transfer the data into AWS S3 buckets [27]. There are countless different options with regards to raw data storage, however, using S3 buckets was the obvious choice considering the whole AWS infrastructure that is used within the platform's development. Using technologies within the same ecosystem allows for much easier integration across services. S3 buckets are very easy to use and have automatic encryption of data with an unlimited server capacity which makes the platform exceptionally scalable.

All user-uploaded files will only temporarily be stored in S3 buckets. Once processing has occurred, the uploaded file will be hashed, and the hashed file and result will be stored in a DynamoDB database. This reduces billing for large file storage and reduces the carbon footprint of the platform, which will be further described in Section 5.13.

5.2.5 Processing - User Data: In order to ensure security from the platform's side, all user data will be encrypted before any transfer of data occurs. In addition, the storage of the data and the transferring process has automatic security built-in to the service, which is a major advantage to the ecosystem selected.

5.2.6 Processing - User Uploads: The user data that is uploaded is the data that needs to be processed by the AI model of choice as explained in Section 2.3 in order to detect if it is a deepfake. The software of choice is AWS SageMaker [28]. AI models can be hosted and run directly from functions, which offer poor management and performance or from cloud computing hosting. Azure and Google both offer similar functionality at comparable specs and pricing with their AI model hosting, but due to the integration with the other AWS services such as Lambda functions, using AWS's AI hosting service made the most sense. As mentioned above, the data will temporarily be stored in S3 buckets to ensure the platform is extremely scalable. Once a file is successfully uploaded to an S3 bucket, a Lambda function will be triggered that connects the data uploaded to a Model endpoint (RESTful API) created by SageMaker. This allows for easy connection and input to the model that is accessible and returns the model's prediction. The prediction and hashed version of the file are then sent to DynamoDB.

AWS SageMaker allows for easy deployment of a model that consists of multiple EC2 instances (virtual servers) which can all respond and contains the model artefact. It is automatically scaled and balanced by AWS ensuring high performance and low downtime regardless of use. The software's load balancer automatically does health checks on the EC2 instances. SageMaker gives model response in real-time at sub-second latency. SageMaker allows for multiple models to be present within one container and one endpoint. This allows the endpoint to accept either image, video or audio files to be accepted as inputs and the relative models defined in Section 2. to process the specific file type.

5.3 Use of open-source data and software

5.3.1 Technologies: The AWS ecosystem of software solutions is used throughout the platform development. There are numerous reasons why using a cloud computing infrastructure is beneficial and it is highlighted within each section. The ability to create a platform that is scalable to global levels at rapid performance speeds without the need for extensive software maintenance while reducing

your carbon footprint from unused and unallocated excess capacity, along with pay-as-you-use costing methods is unparalleled. AWS offers an extensive array of services for every need that integrates perfectly together and is the perfect hosting solution.

All of the following technologies are open-source. Python is the software that is used within development along with the machine learning framework of choice being PyTorch. FastAPI is an API framework that is used to create RESTful endpoints. React is a frontend Javascript library. Docker is a containerization ecosystem. Each technology is free to use with a great community for support.

5.3.2 ML Training Data: The training data that will initially be used comes from the Deepfake Detection Challenge hosted on Kaggle in partnership with AWS, Facebook, Microsoft, etc [15]. This data has a publicly available MIT license for use. This will be used to pre-train the model. The platform will also collect data for model retraining to mitigate for data shifts to ensure the model stays within an acceptable accuracy range.

5.4 AI-model governance

5.4.1 Testing: Once the platform was hosted and live, the model will continuously be tested against known results to ensure the model's predictions were within a reasonable range of accuracy. This will be an indicator that is used to determine when the model should be re-trained or not to avoid unnecessary repetitive model training [29].

5.4.2 Company policy: The output prediction results will reframe from stating if the uploaded file is either fake or real, but rather indicate a level of confidence along with a disclaimer. This removes liability from the model and ensures users understand that the model alone cannot be trusted. This is to remove the spread of misinformation due to faults within the model. The platform will also have a feature that showcases where on the uploaded file the deepfake was detected to ensure transparency and make the AI explainable to the general public [29].

5.4.3 Security: To combat adversarial attacks, the company policy above will be in place along with security measures. This will be explained in Section 5.8. This will ensure the users of the platform know their data is secure and the platform is well protected from attacks.

5.4.4 Sustainable training data: The model will continuously be re-trained with new data to counteract data shifts. This will be done periodically with the hope the new training data will ensure the model is fair and has no biases. Crowdsourcing, described in Section 5.7, will allow the public to influence and correct the model creating a hybrid intelligence which will remove biases and ensure the model remains sustainable.

5.5 Sensitive user information must have adequate protection

The company will attempt to collect as little information as possible to reduce the risk of attackers retrieving sensitive information. The platform will be open to the public under a free tier without the need to sign up. This ensures that the majority of users of the platform will not share their information and will not be at risk. For the users under the paid tier, the platform will have a strong focus on security which will be further expanded on in Section 5.8 where the system's security will be analysed. All information provided by the users will be securely encrypted before any transfer of data occurs on networks.

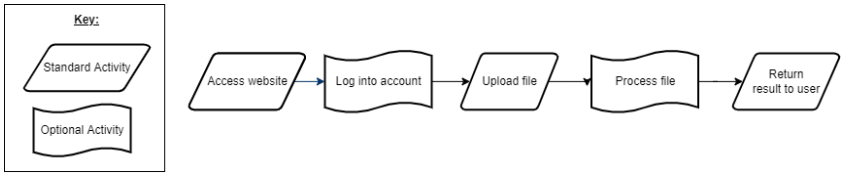


Figure 4: Activity Flow Diagram

5.6 **User interface and user experience**

Figure 4 above shows the basic activity flow for the platform. Logging in is optional, you can upload and process data without needing to be logged in. The processing of the file is also optional, the file may have been processed already and its result is already stored in the database. A more comprehensive version of this is shown in Figure 6 in Section 6. below.

As mentioned above, the website will be hosted on AWS with S3 [30]. There are numerous website hosting services. Any web hosting platform such as Hostinger can suffice while using services such as Heroku are famous for quality cloud-based web hosting. The reason AWS is the best option is due to the array of compatible services you can use alongside hosting the storage. This makes management, monitoring, and security much easier. Along with S3, one of these services is Route 53, which is used for the platform. Route 53 is a scalable Domain Name System (DNS) that automates requests over the internet. It is highly available with servers that are distributed across many availability zones, which removes downtime and helps in consistently routing end users to the platform. It is a reliable and secure way for connecting users on the internet with AWS. Cloudfront will be used in conjunction with S3 and Route 53 as the content delivery network service. This acts as the middleman between the users (Route 53 connection) and the data in S3 (where the website is stored). AWS Certificate Manager is used to ensure the security of Cloudfront for handling this connection. Cloudfront is in charge of distributing the website to that the users will interact. The combination of all the services makes this AWS architecture in conjunction with the whole system architecture the best choice.

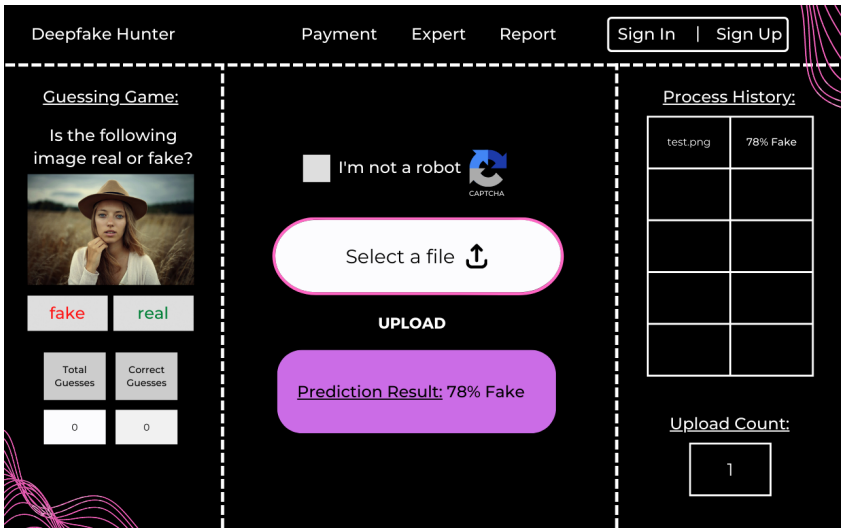


Figure 5: User Interface Design

The frontend of the platform will be developed using React [31]. React is the most popular frontend framework used on the internet today. Frontend development is up to preference, however, React was selected due to large market share with a strong community. Another reason React is selected ahead of standard HTML is that it makes the UI design easier, performs faster, and is more responsive. High-quality UI is vital when creating the frontend of the website. The design focus is on simplicity. This platform should be very straightforward to use. This is shown in the very basic activity flow

diagram shown above in Figure 4.

5.7 Crowdsourcing to verify and rectify incorrect predictions

5.7.1 Advertising data collection: In order to gain valuable advertising data that can be used to showcase the true societal impacts the platform could have, the platform will create a simple detection game for users. Users will select if an image is fake or real. This image will be predetermined in either case and this can be used to showcase to users how difficult it is to detect deepfakes without software. This game will also engage users which will help in advertising.

5.7.2 Expert data collection: To avoid attackers flooding the platform with edge case files, this service will only be available to paid users. The users will upload files and state if they are real or fake. This crowdsourcing approach will help in training the model as it gains a level of hybrid intelligence. To avoid paid users damaging the model with vastly incorrect data, the results should still fall within a certain level of accuracy to qualify as new training data. If a known result is incorrect when it is uploaded, the paid users can report this image for further investigation. Files are not processed again but rather their results are stored with the hashed version of the file, thus the results of such cases will need to be investigated before changing the result confidence level. The result can automatically be temporarily changed to an uncertain confidence level until the investigation is complete, users will only be able to report on a specific amount of images.

5.8 Security of the system against cyberattacks

5.8.1 AWS Systems: AWS Virtual Private Cloud (VPC) will be used for security measures to ensure the system does not get compromised [32]. AWS Identity and Access Management (IAM) will manage the control between different connections and services to limit access using specific roles that are allocated. These technologies will provide general security to ensure attackers cannot access the system.

5.8.2 Captcha: All users will need to successfully pass through a Captcha bot screening protocol before they can upload files to be tested [33]. This will ensure that large amounts of bots do not flood the system and cause the system to crash due to high traffic. As a safeguard for the Captcha technology, users cannot upload files within a minute of each other. In addition to this, for free users, each IP address will only be allowed to upload 10 files a day. These methods will ensure that adversarial attacks will not compromise the system.

5.8.3 Database security: AWS DynamoDB has on-demand backups. This ensures that if data is compromised, backups can be restored to the platform. With AWS's vast servers, if one network or host goes down, the data will still be accessible from other servers.

5.8.4 Data encryption: Client-side encryption of all data will occur. AWS also has vast measures as mentioned before. IAM-specific access controls for all services. AWS encrypts the data again at rest with another layer of security keys. The use of VPC endpoints to ensure safe access to any data.

5.9 System architecture

5.9.1 Development: As mentioned above, the backend will be developed using Python [34]. The main reason Python is used over backend frameworks such as Node.js is due to the powerful libraries that are present with Python. This greatly decreases the development time. Python also has great community support and is the most popular backend technology. Using FastAPI, Python functions

will be turned into RESTful endpoints [35]. FastAPI is preferred over micro-services such as Flask, as FastAPI offers better performance, easily supports concurrency, and has a great dependency injection system. Flask is more flexible, however, considering the performance enhancements, FastAPI is the best choice for creating endpoints for Python functions.

These endpoints will be hosted and made accessible by AWS API Gateway [30]. When called, AWS Lambda functions will be triggered to access these functions' endpoints from the frontend. Lambda functions are the best choice due to the ecosystem of choice for the platform. Google and Azure have similar services such as Google App Engine and Azure App Service. Lambda is selected due to its vast flexibility with excellent performance. The frontend will call these specific endpoints using fetch statements that then trigger the Lambda function to process the Python functions with the given parameters. The Lambda function will consist of both the RESTful function endpoints and a base layer. This base layer consists of all the software dependencies for the platform that is built using Docker. Docker ensures that specific versions of different software match as intended to avoid version collisions. This separation between the infrastructure and the actual application ensures development times decrease.

Based on the specified parameters passed to the Python functions, another Lambda function will be triggered. The uploaded data will be transferred into S3 buckets before being hashed. The hashed version of the file will then query the DynamoDB database to verify if the file has already been processed before. If it has, then the result will be returned to the user. If not, then the file from the S3 bucket will be transferred to the model processing section. The model has been trained and developed using PyTorch and hosted on AWS SageMaker [36]. The only disadvantage AWS is that it does not allow scheduled training jobs. However, since the training data needs to be verified and checked first, this is not a concern for this platform. The only reason SageMaker is selected ahead of Google and Azure's platform is due to the ecosystem of choice being AWS.

The model processing section is accessible by another RESTful endpoint created with the deployment of the model. The data is then processed by the AI model which is automatically scaled and balanced to ensure the best performance. The model endpoint returns the prediction result to the lambda functions input request. The resultant value is then stored in a new entry in the DynamoDB database with the hashed version of the file. The result is returned to the user. Once the result is returned to the user then the file present in the S3 bucket is deleted. Figure 6 in Section 6. showcases this conditional data flow diagram. This diagram along with Figure 2 fully explains the system architecture of this platform.

5.9.2 Infrastructure: A serverless approach is used within the whole infrastructure of the platform. From hosting functions to the database to calling the AI model endpoint, each aspect of the ecosystem is serverless [37]. Section 5.10 and 5.11 will refer to the scalability and maintainability aspect of using a serverless approach. The ease of use and setup time during development is a key factor to consider too which makes using a serverless approach favourable instead of standard server architectures. The pricing is the deciding factor, the use of the software only when traffic is present and users interact with the software is vital. If no users use the software then it comes at no cost, if users use the software then the likelihood of income generation to mitigate the costs that then occur reduces financial risk. Traditional servers consist of continuous payment for the allocation of persistent connections regardless of their inactivity. Another disadvantage to using a server architecture is the limitation that must be set. In its creation, the developer decides how much resources to allocate to it instead of it being fully scalable. There is another aspect to consider regarding latency where servers are better, however, when the platform is in active use then latency from the cold-start of function in the serverless approach is negligible. The possible latency that occurs on first use is acceptable considering all the other advantages.

5.10 **Scalability and fault tolerance**

5.10.1 Scalability: One of the major advantages and deciding factors in the design of the systems architecture relates to the scalability of the platform. The ability for the platform to adapt from having singular users to having thousands of users from one day to the next without the need for added infrastructure or development is crucial. The use of AWS's serverless ecosystem allows the platform to be accessed rapidly from anywhere in the world at extremely high speeds. The system and how it scales is automated and effectively controlled removing the extra costs relating to aspects such as setup and management.

5.10.2 Fault tolerance: A key deciding factor in using a cloud computing solution is the fault tolerance aspect. As mentioned in Section 5.2 and 5.6, AWS manages multiple instances and servers and automatically balances the system to ensure the system is always online and available for use. When a server goes down or a database fails, backups are in place to rapidly fix any issues.

5.11 **Maintainability of the system**

5.11.1 Sustainable Infrastructure: A big reason behind the architecture of the system is its maintainability. Once the system is created, it becomes basically autonomous unless feature improvements want to be made. There is no large server management or database connection control that needs to occur as it is all automatically managed by AWS whereas using a server architecture requires a lot of management and control. The serverless architecture makes the platform very sustainable as AWS controls many aspects that would otherwise need to be managed. The systems and the software are all controlled by Docker images to remove any version errors and the technologies used are either in a perfectly integrated ecosystem or are open-source with large community backing in modern relevant fields.

5.11.2 AI Model: The AI model that is to be used is an open-source, award-winning model that is recent and up to date. The principles of continuous improvement using new training data along with the platform policies to engage users, crowdsource into hybrid intelligence along with user feedback reporting all promote a highly maintainable and sustainable AI model. The hosting platform of choice controls many aspects that would be strenuous and allows for ease of use. Using similar serverless principles as other areas of the design, the dynamically scalable approach with a pay-as-you-use model with RESTful endpoints to interact with the model makes the model easily maintainable.

5.12 **Estimated cost to develop such a system**

5.12.1 Technology costs: Correctly estimating the costs of the system is extremely difficult due to the nature of the infrastructure. Basically all of the technologies used fall within the AWS Free Tier at the very basic use case and then will depend on the number of requests per month. An estimation will be shown of what the AWS SageMaker pricing would cost under a particular situation. If the model endpoint is allocated with 2 GB of memory that gets executed for 1 million seconds (approximately 10 million requests at 100ms each) with a data processing of 10 GB, the total for that will be $40.16. 10 million API Gateway Rest requests cost approximately $41.70. Cloudfront is free up to 10 million requests. For 10 GB of DynamoDB storage, it will cost approximately $38.71. AWS Lambda will cost $26.73 for 100 million lambda requests. S3 with 100 GB of storage with 10 million GET/SELECT and 1 million UPDATE requests will cost approximately $13.01. AWS VPC will cost $36.50 for 1 site-to-site connection that is available 24/7. Almost all amounts are for over 1 million requests but assume with 100 000 users a month, a very conservative estimation would total in $196.81 per month.

5.12.2 Developers costs: Determining the costs of development is another difficult task to do. A large set of assumptions will be made in order to give a rough estimation. Considering the large extent to which this design goes, the development period will not take that long. An estimated 4 weeks of development will be required. At an approximate salary of $30 an hour for the 4 weeks working 8 hours a day, the developer would cost the company $4800. The hourly wage can be considerably lower and also considerably higher depending on where and with what experience the developer is hired.

Expanding the break-even target to 12 months. If 100 000 users are actively using the platform every month, the total year 1 costs will be $7161.72. At a rate of $3 a month for the paid version, the platform will need 200 paid users a month in order to break even. This is 0.2% of users and is an achievable rate. Other aspects such as advertising costs that may be needed to gain 100 000 active monthly users were not considered and are not in the scope of this design.

5.13 The carbon footprint of the platform

5.13.1 Development: The uploaded files are temporarily stored in S3 buckets to prevent a bottleneck at the SageMaker entry point. These files will be hashed using methods such as a 64-bit SHA-3 hash function which is notoriously good in avoiding hash collisions [38]. The hashed files and the prediction result from SageMaker will then be stored in DynamoDB. The files will then be removed from S3 buckets. This process reduces the carbon footprint of the system as the files are not stored long-term. Another vital aspect relates to processing power. Whenever a file is uploaded and hashed, the DynamoDB table will be checked to ensure the hashed file is not already present in the database. If it is present then the system does not continue to SageMaker for processing but rather delivers the result straight away from the database storage.

5.13.2 Company strategy: A company policy to ensure the reduction of its carbon footprint would be to set a fixed percentage of profits to be donated in order to make the company carbon neutral. The saving and storing of 50 GB of data within the cloud will result in approximately 0.1 tons of CO2. At an estimation of $29 per ton, the company will donate $2.9 to offset its carbon emissions so the platform is a net zero carbon platform.

5.14 Income generation strategy

5.14.1 Free version: Similar to the principles followed by Wikipedia, a platform with a goal to help fight misinformation cannot be subject to advertising. Therefore the platform will have a call to action for any possible donations that companies or individuals are willing to give. The engagement game where users are tested will be available to everyone, this information can be collected and used for advertising purposes or simply to enhance the number of active monthly users.

5.14.2 Paid services: All users will be limited to 10 file uploads a day. In order to upload an unlimited amount of files, contribute to the crowdsourcing efforts and community, and gain additional features such as deepfake explained detection, it will cost a user $3 a month. This is considered a relatively low figure as the intention of the platform is not to generate large incomes but rather to provide a service for the general public.

6. TESTING

Figure 6 below showcases a sequential and conditional flow chart relating to the functionality and activity of the system. For testing purposes, each step within the flow chart should be considered as a scenario test case. The expected functionality needs to occur in order for the scenario test case to be successful.

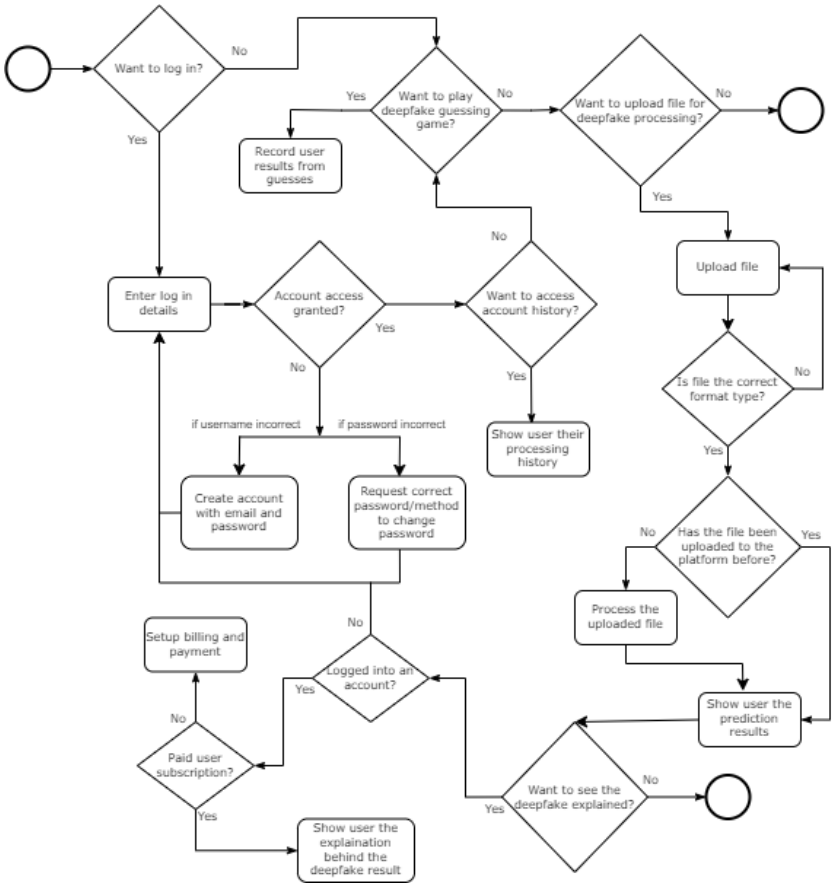


Figure 6: Conditional Data Flow Diagram

Subsections 6.1 to 6.3 below breaks down the different unit tests that will be present within each component of the project. This will be presented as an array of questions and occasionally some of their expected outcomes, each of which is considered as its own individual test case that needs to be passed. These questions relate to functionality within each component and these test cases drive the design solutions found in Section 5. above and their desired functionality during development.

6.1 **Frontend**

Are buttons clickable and connected to functionality? Can the user upload their file to the input slot provided? Can users enter logging-in details? Can users pay for the service? Can users see if they are logged in? Can users see the prediction results returned to them? Can users see the processing history tables if they a paid users? Does the clicks within the guessing game work? Is the website responsive and dynamic? Is the platform accessible on multiple browsers? Can users see the deepfake result explained?

6.2 **Backend and Processing**

If the user clicks to log in, do the logging procedure and DynamoDB GET/CREATE requests occur? If the upload button is clicked, is the file uploaded to the S3 bucket? If the file is uploaded, does the lambda function trigger to check if its hashed value is present in the database? If it is present in the database, does it return the result values to the frontend? If it is not present in the database, does the file get transferred from the S3 bucket to the AI model endpoint? Does the model process the inputs received? Does the model endpoint return valid prediction results?

6.3 Database and Security

Is the data entered encrypted client-side? Is the data received encrypted server-side? Is the IAM access roles valid? Does the VPC work on the website? Is the data received and stored successfully in the database? Does the transfer of data from one source to the next occur? Once all procedures are complete, does the uploaded data automatically get deleted from the S3 bucket? Does the Captcha mechanism work to prevent bots? Is the database logging the number of uploads per user correctly? Is the conditional rejected if uploads exceed the upload limit? Does the platform store crowdsourced data securely for retraining and processing?

7. CRITICAL ANALYSIS

The focus of the design of this platform is related to a couple of basic things. The platform has to be very cost-effective for sustainability purposes which leads to the serverless architecture decision with their pay-as-you-use service being optimal. The platform has to be user-friendly which leads to simple UI design choices. The platform has to keep users engaged and drive traffic which leads to the decision for the deepfake guessing game. The platform has to be available at all times whilst abiding by the PoPI act which leads to the AWS ecosystem with its Cape Town location as the cloud computing solution of choice. The carbon footprint of the platform has to be low, which leads to the decision to store hashed versions of the uploads with their results to reduce unnecessary processing occurring, which lowers its environmental impacts. The platform needs to be easily maintainable and scalable which makes the serverless system AWS offers the perfect choice. The platforms prediction results need to be as accurate as possible whilst removing its responsibility for any inaccuracies, therefore the models of choice uploaded on SageMaker with its easy-to-manage and train functionality whilst returning prediction results that will be shown as a confidence level instead of a definite result, allows for this to occur.

The main reason this platform is to exist is for the social impacts it will have. Misinformation has dire effects on the world relating to many fields. The use of this platform to help reduce misinformation by giving users clarity regarding possible deepfakes is the main driving factor. The use of disclaimers and reporting to reduce negative impacts along with crowdsourcing allows users to interact and turn the model into a sense of hybrid intelligence managed and overseen by experts to give the public confidence in its results, allowing the full fruition of its social benefits. From an ethical standpoint, this hybrid-intelligence strategy ensures the model does not train with biases, it keeps the model monitored and in check. The addition of an income generation strategy will allow will ensure the long-term sustainability of the platform whilst providing jobs to the economy for the platform's upkeep and progression. The company policy with its target to be a net zero carbon emission platform will be achievable with this income generation strategy. However, the platform and its main purpose are free to use to the public without the need to provide any user information making it secure and safe for everyone's use.

8. CONCLUSION

Deepfakes present a challenge that must be addressed. The public needs tools to be able to verify the authenticity of the media they are presented with. The design of this platform focused on many crucial aspects such as the selection of the best artificial intelligence methods for computing the detection along with the whole system architecture within a secure ecosystem. The design broke down the planning and project requirements before presenting an in-depth design solution with clear and easy-to-follow descriptions and diagrams. Design by testing was a principle that was followed as an in-depth look at all the desired functionality took place showcasing the desired flow and key unit tests that should be passed in successfully creating the platform. The design was then critically analysed highlighting key aspects of its functionality, sustainability, environmental, social, and economic impacts.

REFERENCES

[1] M. Tan, "Efficientnet: Rethinking model scaling for convolutional neural networks," *ICML*, 05 2019. [Online]. Available: https://arxiv.org/abs/1905.11946

[2] T. Evgeniou and M. Pontil, "Support vector machines: Theory and applications," *Machine Learning and Its Applications, Advanced Lectures*, 01 2001. [Online]. Available: https://www.researchgate.net/publication/221621494_Support_Vector_Machines_Theory_and_Applications

[3] A. T. Nasser, "Signature recognition by using sift and surf with svm basic on rbf for voting online," *2017 International Conference on Engineering and Technology (ICET)*, 08 2017. [Online]. Available: https://ieeexplore.ieee.org/document/8308208

[4] G. Neha, "Image recognition using svms vs. cnns," *ResearchGate*, 02 2019. [Online]. Available: https://www.researchgate.net/post/Image-recognition-using-SVMs-vs-CNNs/5c63e3e6aa1f093fcb698dbf/citation/download

[5] G. D. Luca, "Svm vs neural network," *Baeldung on Computer Science*, 06 2020. [Online]. Available: https://www.baeldung.com/cs/svm-vs-neural-network

[6] K. Ando, "A multithreaded cgra for convolutional neural network processing," *Circuits and Systems*, vol. 08, 06 2017. [Online]. Available: https://www.scirp.org/Journal/PaperInformation.aspx?paperID=77319

[7] A. Sharkawy, "Principle of neural network and its main types: Review," *Journal of Advances in Applied Computational Mathematics*, 08 2019. [Online]. Available: https://www.researchgate.net/publication/343837591_Principle_of_Neural_Network_and_Its_Main_Types_Review

[8] R. Khan, "Evaluating the performance of resnet model based on image recognition," *2018 International Conference on Computing and Artificial Intelligence*, 11 2018. [Online]. Available: https://www.researchgate.net/publication/328955911_Evaluating_the_Performance_of_ResNet_Model_Based_on_Image_Recognition

[9] D. Afchar, "Mesonet: a compact facial video forgery detection network," *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, 12 2018. [Online]. Available: https://arxiv.org/abs/1809.00888

[10] A. van den Oord, "Wavenet: A generative model for raw audio," *Google DeepMind*, 9 2016. [Online]. Available: https://arxiv.org/abs/1609.03499

[11] J. Frank, "Wavefake: A data set to facilitate audio deepfake detection," *Horst Görtz Institute for IT-Security*, 11 2021. [Online]. Available: https://arxiv.org/abs/2111.02813

[12] H. Ku, "Face recognition based on mtcnn and convolutional neural network," *Frontiers in Signal Processing*, vol. 04, 01 2020. [Online]. Available: http://isaac-scientific.com/images/PaperPDF/FSP_100038_2019102417055645757.pdf

[13] S. Zhang, "S^3fd: Single shot scale-invariant face detector," *Institute of Automation*, 08 2017. [Online]. Available: https://arxiv.org/abs/1708.05237

[14] E. Kaziakhmedov, "Real-world attack on mtcnn face detection system," *Intelligent Systems Lab*, 10 2019. [Online]. Available: https://www.researchgate.net/publication/336550776_Real-world_attack_on_MTCNN_face_detection_system

[15] Kaggle, "Deepfake detection challenge," 12 2019. [Online]. Available: https://www.kaggle.com/competitions/deepfake-detection-challenge/overview

[16] S. Seferbekov, "Deepfake detection (dfdc) solution by @selimsef," 09 2021. [Online]. Available: https://github.com/selimsef/dfdc_deepfake_challenge

[17] Team-Neighborhood, "Awesome face detection - processing time," 11 2019. [Online]. Available: https://github.com/Team-Neighborhood/awesome-face-detection

[18] Q. Xie, "Self-training with noisy student improves imagenet classification," *Google Research*, 11 2019. [Online]. Available: https://arxiv.org/abs/1911.04252

[19] S. Hashemi, "Cloud computing vs. grid computing," *ARPN Journal of Systems and Software*, vol. 02, 05 2012. [Online]. Available: https://www.researchgate.net/publication/327982199_cloud_computing_vs_grid_computing

[20] I. Nwobodo, "A comparison of cloud computing platform," *International Conference on Circuits*

and Systems, vol. 15, 01 2015. [Online]. Available: https://www.researchgate.net/publication/282323121_A_Comparison_of_Cloud_Computing_Platform

[21] T. Mufti, "A review on amazon web service (aws), microsoft azure google cloud platform (gcp) services," *International Conference on ICT*, 01 2021. [Online]. Available: https://www.researchgate.net/publication/282323121_A_Comparison_of_Cloud_Computing_Platform

[22] K. Marko, "Infrastructure for machine learning, ai requirements, examples," *SearchDataCenter*, 10 2020. [Online]. Available: https://www.techtarget.com/searchdatacenter/feature/Infrastructure-for-machine-learning-AI-requirements-examples

[23] AWS, "Amazon sagemaker pricing," *MachineLearning*, 01 2022. [Online]. Available: https://aws.amazon.com/sagemaker/pricing/

[24] AWS, "Amazon s3 pricing," *Storage*, 01 2022. [Online]. Available: https://aws.amazon.com/s3/pricing/

[25] T. Ondeng, "Tcp/ip technology," *University of Nairobi*, 01 2017. [Online]. Available: https://www.researchgate.net/publication/318960971_TCPIP_TECHNOLOGY

[26] T. Asaad, "The sql vs nosql differences and similarities," *International Journal of Scientific Engineering Research*, vol. 10, 08 2019. [Online]. Available: https://www.ijser.org/researchpaper/The-SQL-vs-NoSQL-Differences-and-Similarities.pdf

[27] P. Boisrond, "A position paper on amazon web services (aws) simple storage service (s3) buckets," *Cybersecurity / Malware / Reverse Engineering*, 08 2021. [Online]. Available: https://www.researchgate.net/publication/353922666_A_Position_Paper_on_Amazon_Web_Services_AWS_Simple_Storage_Service_S3_Buckets

[28] I. Shcherbatyi, "Amazon sagemaker automatic model tuning: Scalable black-box optimization," *ResearchGate*, 12 2020. [Online]. Available: https://www.researchgate.net/publication/347300653_Amazon_SageMaker_Automatic_Model_Tuning_Scalable_Black-box_Optimization

[29] G. Sharma, "Artificial intelligence and effective governance," *Sustainable Futures*, 01 2020. [Online]. Available: https://www.researchgate.net/publication/338369839_Artificial_Intelligence_and_Effective_Governance_A_Review_Critique_and_Research_Agenda

[30] N. Kewate, "A review on aws - cloud computing technology," *Cloud Computing*, 01 2022. [Online]. Available: https://www.researchgate.net/publication/358243805_A_Review_on_AWS_-_Cloud_Computing_Technology

[31] C. Ritwik, "React.js and front end development," *International Research Journal of Engineering and Technology (IRJET)*, vol. 07, 04 2020. [Online]. Available: https://www.irjet.net/archives/V7/i4/IRJET-V7I4714.pdf

[32] S. Chauhan, "Advanced amazon virtual private cloud (amazon vpc)," *Amazon Virtual Private Cloud*, vol. 03, 08 2018. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/9781119549000.ch3

[33] B. Saini, "A review of bot protection using captcha for web security," *IOSR Journal of Computer Engineering*, vol. 08, 01 2013. [Online]. Available: https://www.researchgate.net/publication/272719923_A_Review_of_Bot_Protection_using_CAPTCHA_for_Web_Security

[34] A. Sharma, "Python: The programming language of future," *IJIRT*, vol. 06, 09 2021. [Online]. Available: https://www.ijirt.org/Article?manuscript=149340

[35] N. Misra, "Deploy machine learning models using fastapi: A step by step walkthrough," *Medium*, vol. 06, 08 2021. [Online]. Available: https://medium.com/machine-learning-india/deploy-machine-learning-models-using-fastapi-a-step-by-step-walkthrough-f812ca8043ad

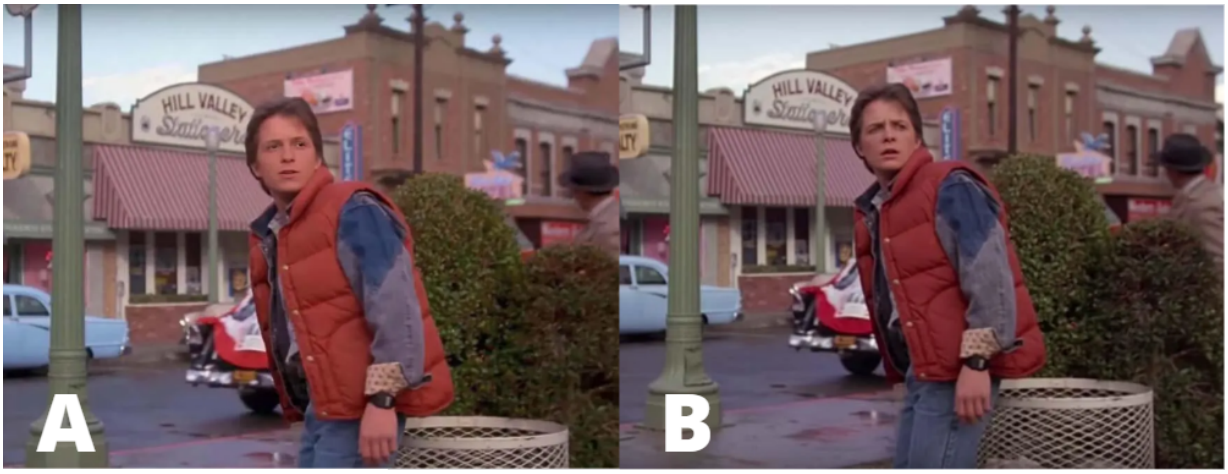
[36] A. Paszke, "Pytorch: An imperative style, high-performance deep learning library," *Facebook AI Research*, 12 2019. [Online]. Available: https://arxiv.org/abs/1912.01703

[37] L. Jiang, "Overview of serverless architecture research," *Journal of Physics Conference Series*, 01 2020. [Online]. Available: https://www.researchgate.net/publication/339680821_Overview_Of_Serverless_Architecture_Research

[38] M. Dworkin, "Sha-3 standard: Permutation-based hash and extendable-output functions," *NIST Pubs*, 08 2015. [Online]. Available: https://www.nist.gov/publications/sha-3-standard-permutation-based-hash-and-extendable-output-functions

Deepfake Hunter

An AI-enabled Software Platform for Detecting Deepfakes on the Internet



Deepfakes are synthetic media where parts of the original source is replaced with another version to seem real, but fool the viewer. Can you tell which image above is real and which is fake? This is a relatively easy example. Image A is the fake. It is Tom Holland's face within the famous movie 'Back to the Future' as Marty McFly. Seeing two images next to each other within a setting many people are familiar with makes it easy to tell the difference, but what if you were not familiar with the original and only saw the fake? And what if that fake is portrayed in a damaging manner to spread misinformation?

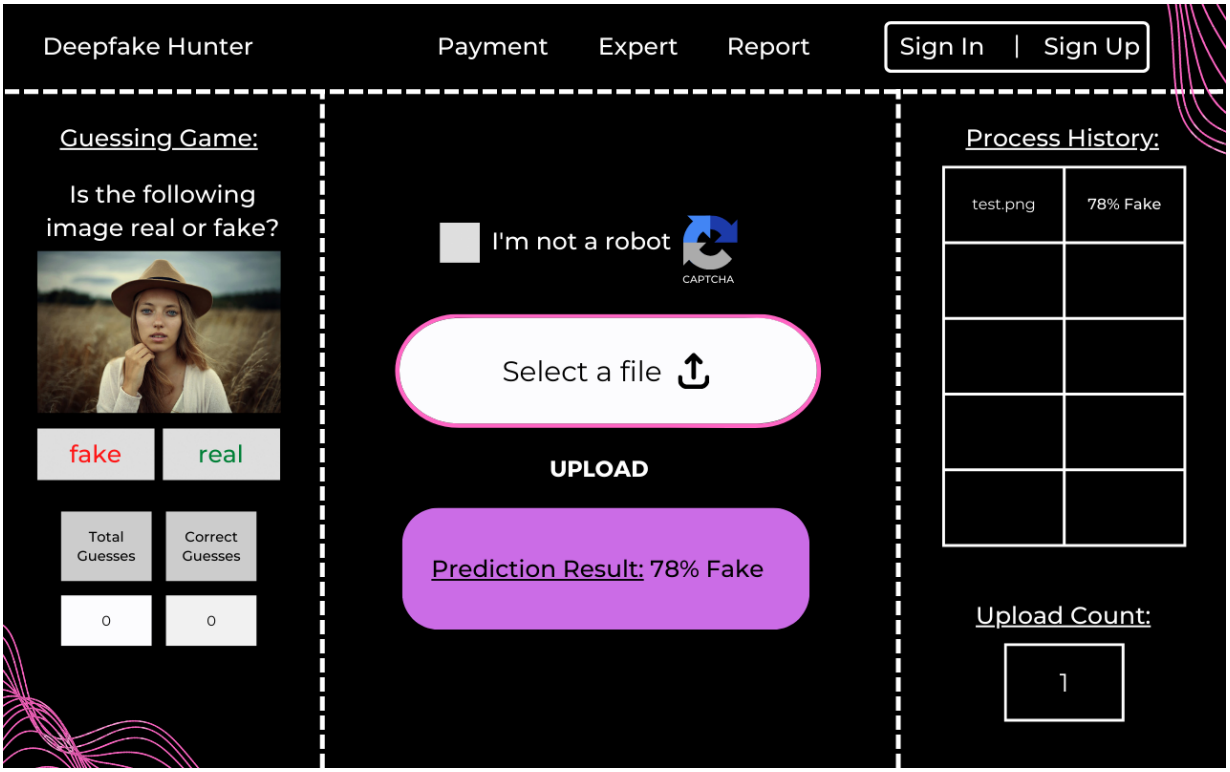
The advancement of deepfake technology has made it near impossible to tell by the human eye. Therefore the use of AI is vital to combat this problem in allowing detection of deepfakes. The Deepfake Hunter is a software platform that is open to the public for free use, so that people can verify if the media they receive is truly real. The website is straightforward to use, simply upload your file and the AI will do the rest.

There is paid version of the platform if you are interested in seeing how and where the AI detected the deepfake and if you would like to be apart of the community to ensure the AI keeps improving and growing. Within the platform you will notice there is a guessing game, try it out and see how well you can detect the fakes from the reals.

This platform aims to help society in a fundamental way. Everybody deserves to know if they are being lied to and if the media that is being spread has been tampered with. This is why everyone should use the platform. Deepfake Hunter will help stop the spread of misinformation and give clarity to the public regarding the images, videos and audio files they receive.

The addition of the paid community will ensure the platform remains sustainable long-term and will help in its environmental efforts. Deepfake Hunter has a net zero carbon policy to ensure you can verify fakes without harming the climate. Another feature that is present within the platform is the processing history log. For example, if you upload an image and the AI processes it to return a result to you, if someone else uploads the same exact image later on, the AI already knows the results of that image and will not need to process it again. This reduces the carbon footprint of the platform further.

The expert community that is within Deepfake Hunter will ensure the AI remains sustainable and



successful. Community members can upload known fake media files to the platform where the media will be confirmed and then used to train the AI. This will turn the AI model from solely artificial intelligence to a form of hybrid-intelligence. By having this human confirmation and training method, the experts can ensure the AI model does not develop any biases and remains as accurate as possible. From an ethical standpoint this is very important.

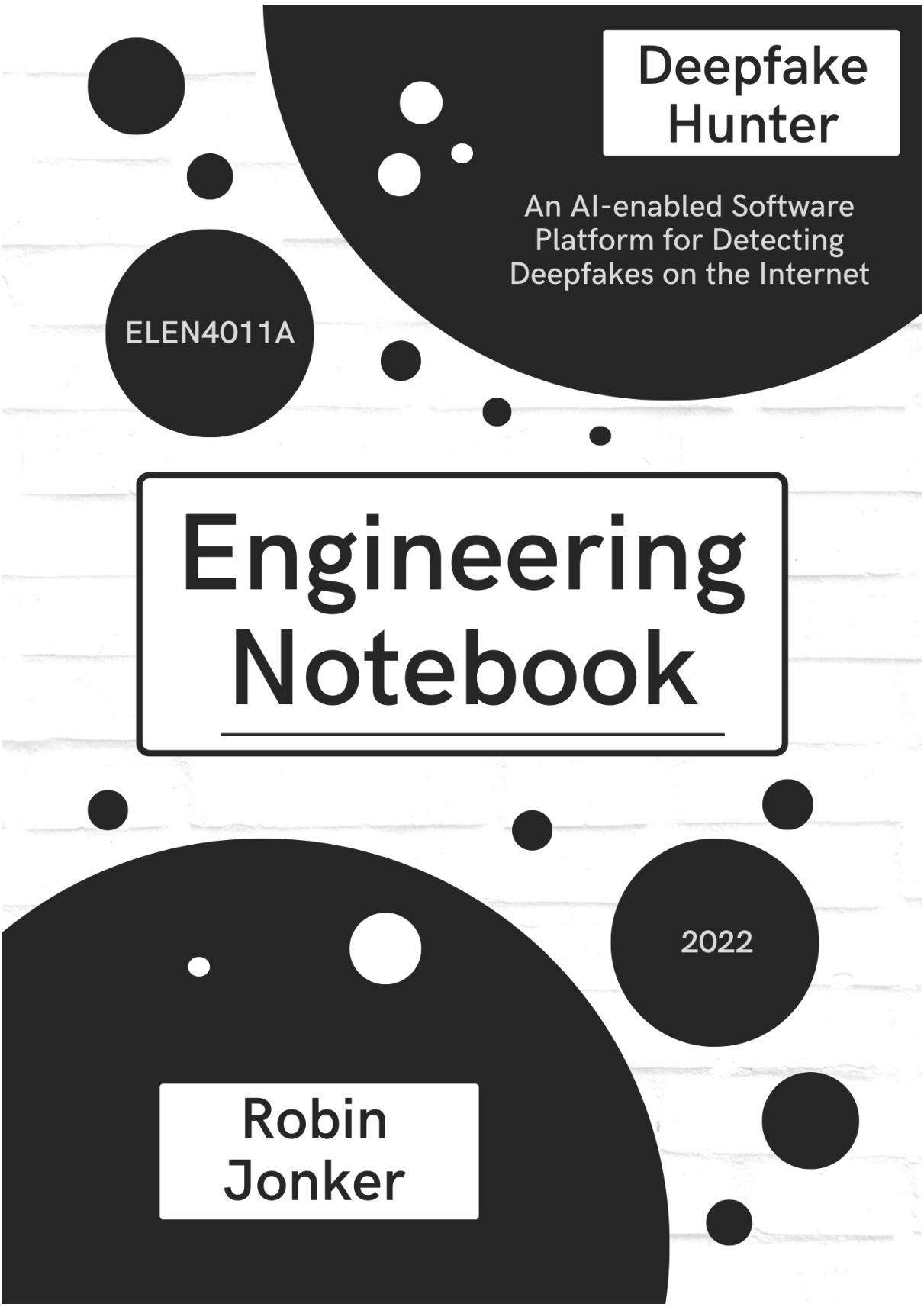
Deepfake Hunter is a platform that places a lot of focus in its security. As a platform that fights misinformation, it is crucial that it is not a source of any misinformation. Any data provided to the platform will face double-encryption along with many other features to ensure the platform is safe and secure. When the platform returns the results to users, it will do so in the form of a predication percentage. No AI model is ever 100% perfect, therefore it is important to rather give users a confidence level regarding the result. This confidence level will ensure the platform does not give definite false answers that can cause the spread of misinformation that can harm society.

The platform is built in such a way that it is economically very efficient. There is no overhead or long-term costs, with the expenses only increasing as the user base of the platform increases. This model ensures that the platform is economically self-sufficient and sustainable.

In summary, Deepfake Hunter is a platform that wants to provide a service that will greatly benefit society, while not causing any environmental harm, while being sustainable and secure, with proper guidance and maintenance to uphold the highest of ethical standards.

APPENDIX B

This appendix contains the engineering notebook that was used throughout the design project. This is not exhaustive of all the work that was done throughout the time but rather a combination of rough notes and thoughts. The notebook was often used an inspiration for work that was directly included in the report or diagrams that was then created externally and included in the relative sections of the report. This is included to showcase the continuous and effective use of time throughout the allocated period.


Deepfake Hunter
An AI-enabled Software Platform for Detecting Deepfakes on the Internet
ELEN4011A
Engineering Notebook
2022
Robin Jonker

Field of interest:
Topic 9: Software I

Proposed title of project:
Deepfake Hunter: An AI-enabled Software Platform for Detecting Deepfakes on the Internet

Supervisor:
Prof. T. Celik

Stream:	**Pre-requisite courses (if any):**
EE or IE	None

Brief description of content and scope of project:

The last decade has witnessed a revolution in Artificial Intelligence (AI), mainly due to advances in computing technologies, the availability of open-source data, and several breakthroughs in deep learning. Deep learning has successfully solved complex problems across domains and has overpassed humans' performance on specific tasks such as image recognition. Deep learning methods, such as deep neural networks, can now generate synthetic data that follows the distribution of information we may more likely observe in our natural world. Deepfake algorithms based on deep learning methods can generate fake images and videos that humans cannot distinguish from real ones. Deepfakes can cause threats to society and national security.

In this project, students will design an AI-enabled software platform that ordinary citizens can use to verify the authenticity of an image or video on the Internet. Some of the essential design considerations include:

1. Hardware requirements;
2. Data acquisition, storage, and processing;
3. Use of open-source data and software;
4. AI-model governance;
5. Sensitive user information must have adequate protection;
6. How will users interface with the system? Special attention must be given to the user interface and user experience;
7. Use of crowdsourcing to verify and rectify incorrect predictions from deepfake detectors;
8. Security of the system against cyberattacks;
9. System architecture;
10. Scalability and fault tolerance;
11. Maintainability of the system;
12. Estimated cost to develop such a system;
13. The carbon footprint of the platform.

Week 1: 25 July 2022

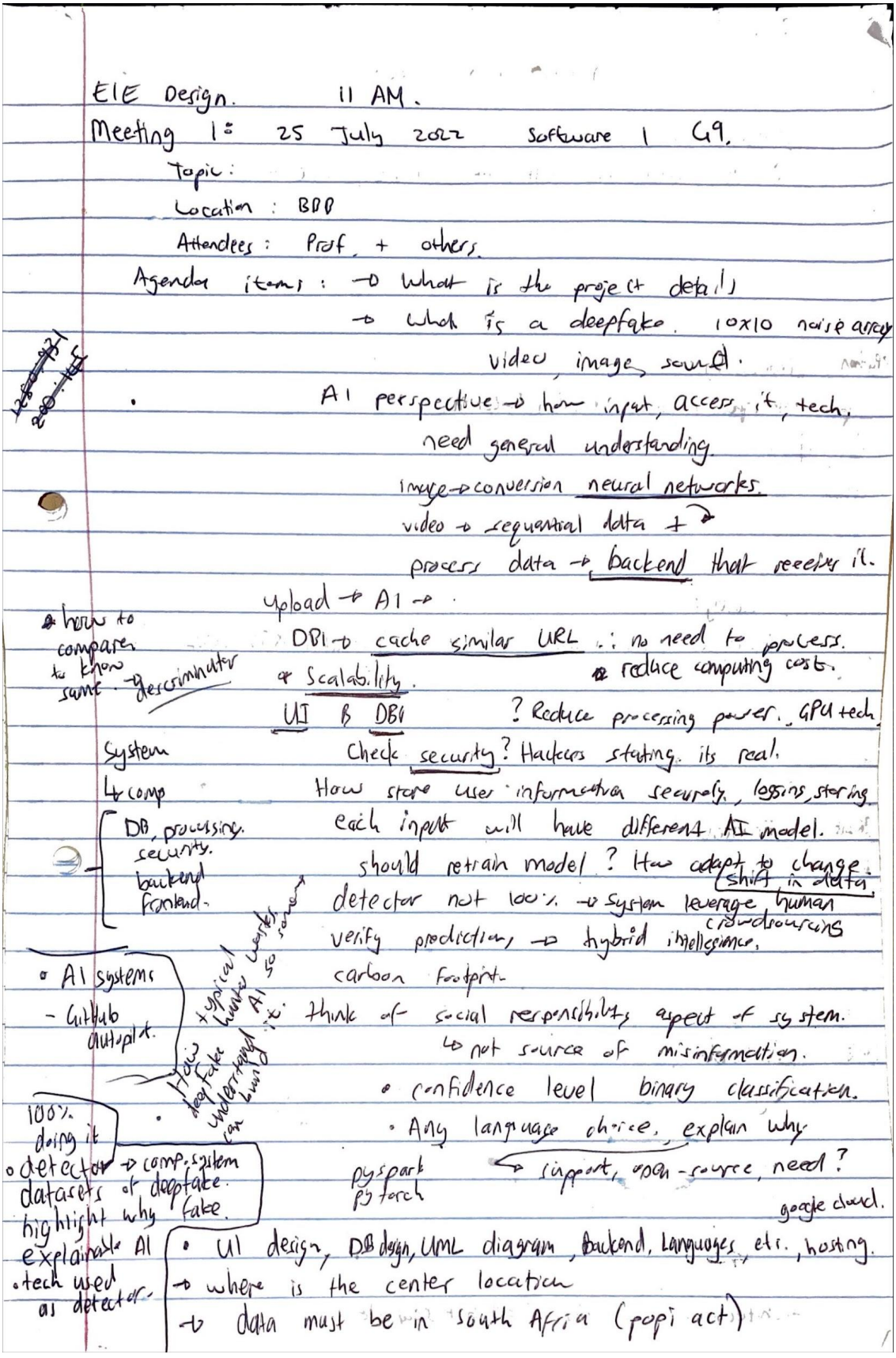
Summary: The tech stack that will be used within the development of the platform was decided upon.

It was important to decide early on what the intended development technology will be used as this will alter decision making moving forward in many aspects.

The use of containers was an early decision made which is important when working with every changing frameworks that having alterations in each version. The use of a software like docker will allow the code dependencies to only be setup once and then software initialization and downloading will not be a worry again moving forward regardless of who is working on the platform. This emphasizes the point of code being portable.

The stack was selectioned on a board range of reasons, such as being new and fully supported languages and frameworks that has a strong community, future growth and support. In summary the frontend will be made using React for the creation of the webapp. The backend will be created with Python. AWS and Vercel will be the hosting service of choice for the backend and frontend respectively. As mentioned above, Docker will be used for containerisation. Other libraries will also be used such as FastAPI to create RESTful API's and Tailwind CSS for styling, along with others that will not be the focal point in the broader architecture.

Meeting notes for this week will be presented below. These are not neat and is more used to write down any and everything that comes to mind.


EIE Design. 11 AM.
Meeting 1: 25 July 2022 Software 1 G9.
Topic:
Location: BDD
Attendees: Prof + others.
Agenda items: → what is the project details
→ what is a deepfake. 10x10 noise array
video, image, sound.
AI perspective → how input, access it, tech,
need general understanding.
image → conversion neural networks.
video → sequential data →
process data → backend that receives it.
Upload → AI →
DB → cache similar URL .: no need to process.
Scalability. reduce computing cost.
UI & DB ? Reduce processing power. GPU tech
Check security? Hackers stating its real.
How store user information securely, logging, storing
each input will have different AI model.
should retrain model? How adapt to change
shift in data
detector not 100% → system leverage human
crowdsourcing
verify prediction, → hybrid intelligence.
carbon footprint
think of social responsibility aspect of system.
→ not source of misinformation.
• confidence level binary classification.
• Any language choice, explain why.
→ support, open-source, need?
pyspark
pytorch
google cloud.
• UI design, DB design, UML diagram, backend, Languages, etc., hosting.
→ where is the center location
→ data must be in South Africa (popi act)
how to compare to know same → descriminator
System & comp
DB, processing, security, backend, frontend.
• AI systems
- GitHub output.
typical human AI
how deepfake understanding
100% doing it
• detector → comp. system
datasets of deepfake
highlight why fake.
explainable AI
• tech used as detector.

Week 2: 1 August 2022

Summary: The tech stack was further investigated, such as deciding upon the hosting architecture, along with the selection of the deepfake detection method of choice.

It was important to decide methods to use within the platforms creation considering many things. A notable question for next week would be about the use of a server. If processes and their respective results are stored on a database to reduce future processing, does that storage results not open the platform up for possible attacks? Is using a serverless process that is dormant and free when not in use not better than having a full server, this is apart of the hosting question and possible highlights scalability issues in hosting a server, eg use AWS Lambda?

Server vs Serverless? Serveless was chosen based on three key metrics, setup and maintenance, scalability, and cost. Latency is considered but with increased traffic, cold starts will not be a problem.

Considering Databases, there are 5 factors to consider:

Data model needs: The data model of choice will come with a variety of pros and cons and the decision will be made based on the other 4 factors.

Connection model: Instead of needing a persistent TCP connection to the database server that reuses this connection across multiple requests, using serverless removes the time and cost of use and reduces the connection resource that gets allocated to each open connection. It is inefficient to create a persistent DB connection for each request as you are paying the connection cost for something that may not be used again.

Infrastructure as code: It does not contain just the compute but also the queues, streams, blob storage and event trigger all together that allows the database to be created in a consistent repeatable way.

Fully managed: Using low-level infrastructure management, the developers avoid the maintenance associated with patching, upgrading and scaling a database.

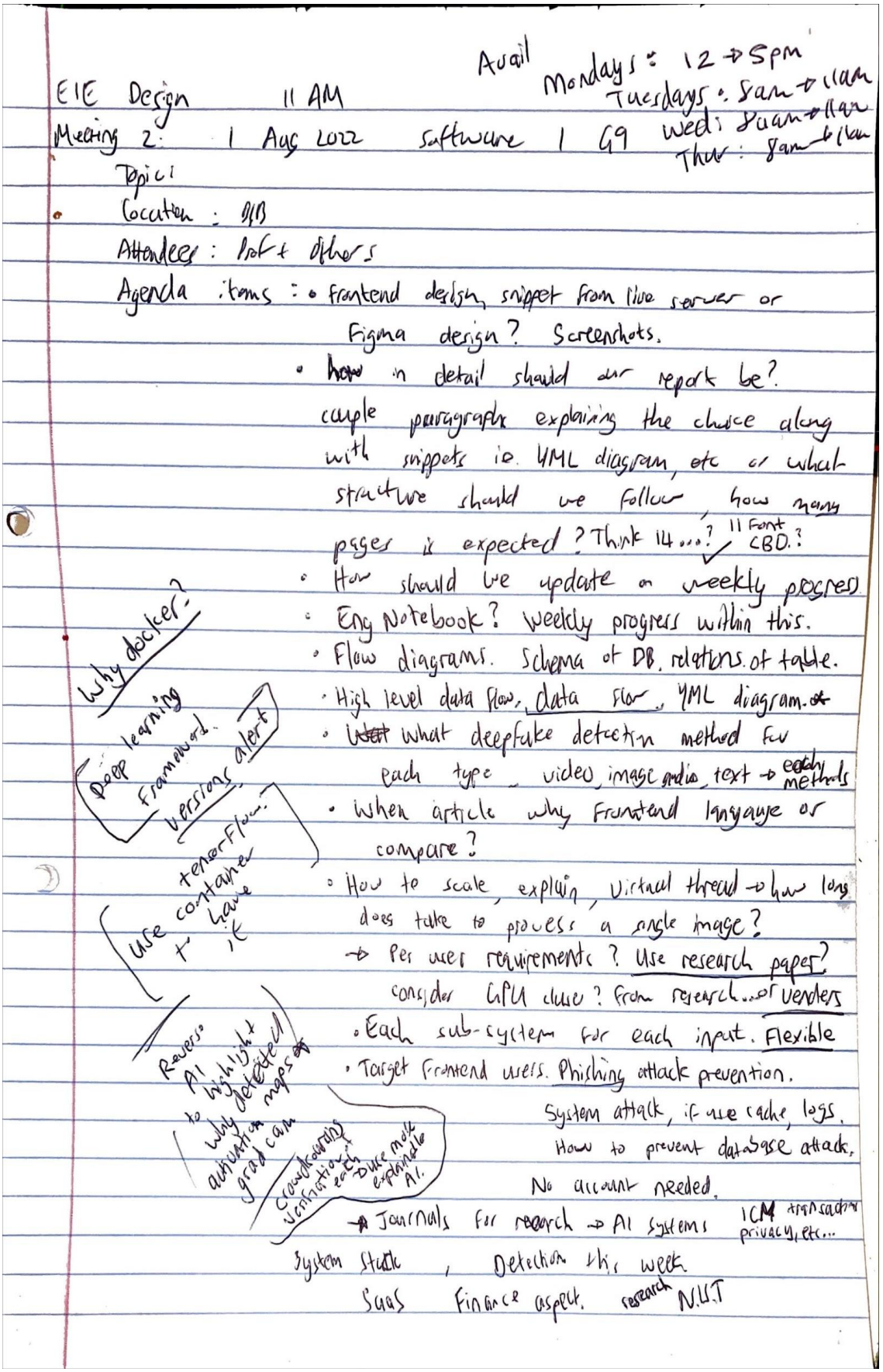
Pricing: Instead of paying a fixed rate regardless of traffic, a pay as you go pricing system avoids wasted funds. For DB its the same, only pay for the actual resource size that is stored.

Two different serverless database hosting frameworks were decided upon. Future considerations on deciding which one will be better will be done. Dynamo versus Aurora Serverless. Both are hosted by AWS which fits perfectly with the architecture of hosting the REST API using AWS Lambda. Currently I am considering Aurora as the favorite as it brings the flexibility of RDMS with the advantages of serverless hosting. Dynamo is the best option if the data access patterns are fixed and unknown. However due to the grouping constraint that Dynamo brings, and the application having clear groupings for eg between images/video/audio, a flexible SQL solution is needed so Aurora is the database hosting choice. Along with that Aurora also has a feature called Data API, this removes the need for worrying about connections and their resource usage but rather creates a REST endpoint to communicate with the database.

The deepfake detection was decided upon, however this is not a final decision and rather a proof of concept that will be investigated further in the future.

For the deepfake detection I have found a couple methods that has proved successful in the past. I researched different entries to the deepfake detection challenge that was found on Kaggle 2 years ago where the prize pool was $1mil. The software they used have MIT lisences and therefore could be used as the methods of choice. Further research will be done.

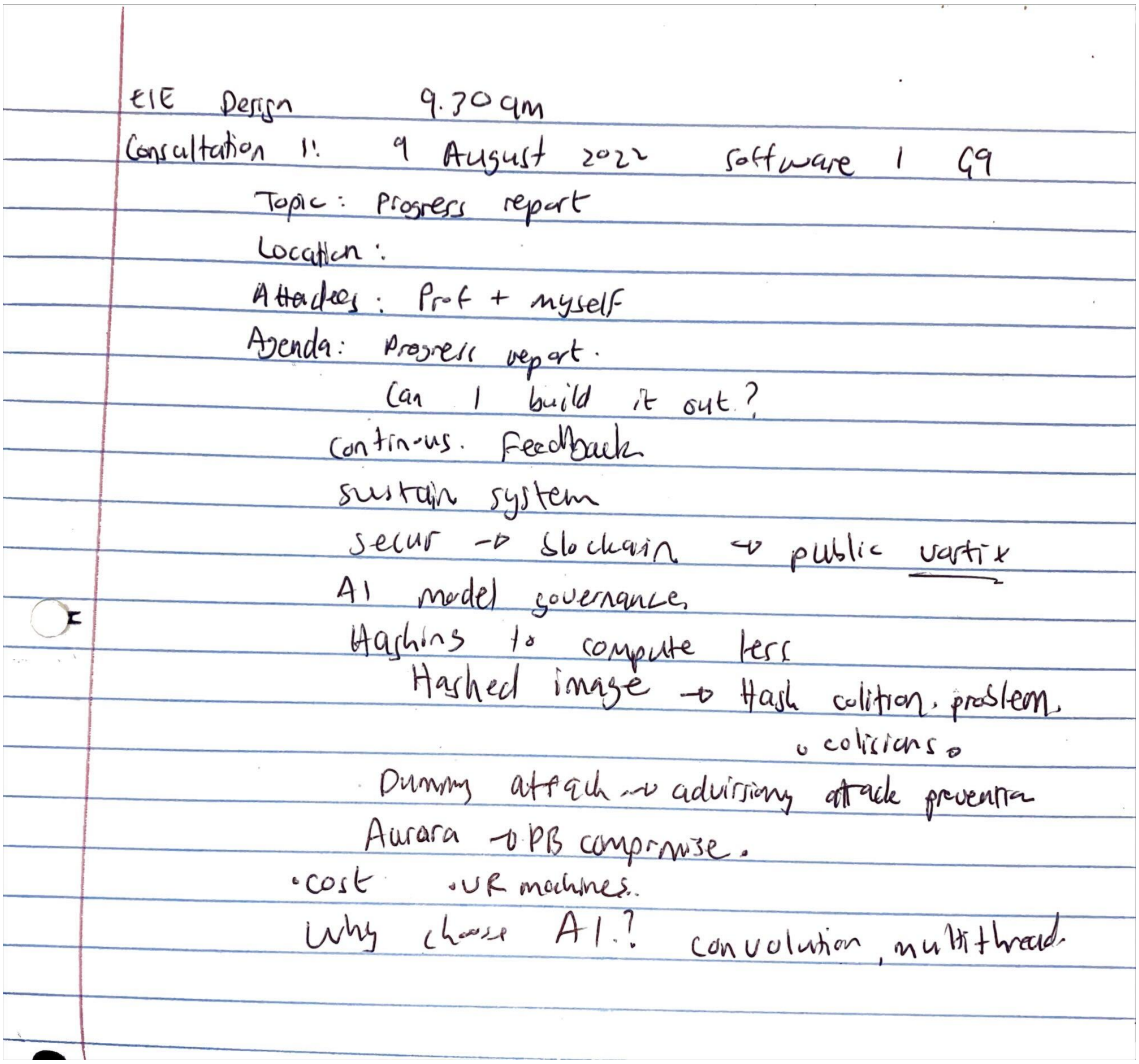
Meeting notes for this week will be presented below. These are not neat and is more used to write down any and everything that comes to mind.



Week 3: 8 August 2022

Summary: A slow week that was mostly set on doing more research, more particular research done into varying neural networks architectures

Within this week a consultation meeting was set up. The notes from that meeting will be included below.


EIE Design 9.30am
Consultation 11: 9 August 2022 software 1 G9
Topic: Progress report
Location:
Attendees: Prof + myself
Agenda: Progress report.
Can I build it out?
Continuous Feedback
sustain system
secur → blockchain → public vartix
AI model governance,
Hashing to compute less
Hashed image → Hash collition, problem,
↳ colisions
Dummy attack → advirsiony attack prevention
Aurara → PB compromise.
•cost •UR machines.
Why choose AI.? convolution, multithread.

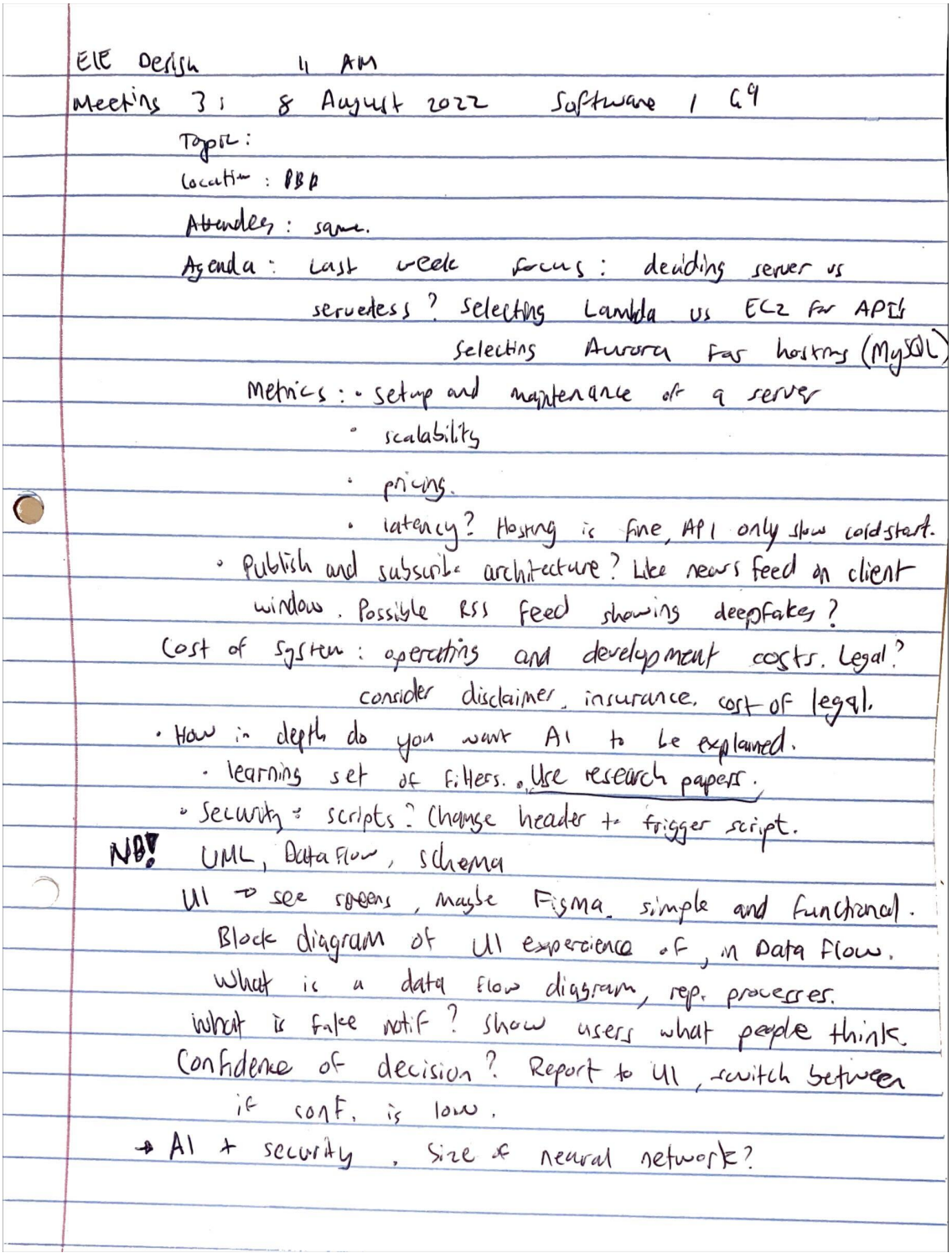
Research was done within selecting which AI method to use in detecting deepfakes for both video and images. No conclusion was made, more research is still needed. Using award winning methods such as those mentioned last week makes the most sense as there is a history of tests that occurred instead of just theoretical research.

https://www.researchgate.net/publication/328955911_Evaluating_the_Performance_of_ResNet_Model_Based_on_Image_Recognition
https://medium.com/mlearning-ai/understanding-efficientnet-the-most-powerful-cnn-architecture-eaeb40386fad
https://arxiv.org/abs/1809.00888
https://arxiv.org/abs/1609.03499

https://arxiv.org/abs/2111.02813

Most of the required questions for this project has been answered, future questions will only arrive once further progress has been made in the report. The current goal is to make substantial progress next week with the report in hopes that the final 2 or 3 weeks can be used on actually creating the platform.

Meeting notes for this week will be presented below. These are not neat and is more used to write down any and everything that comes to mind.

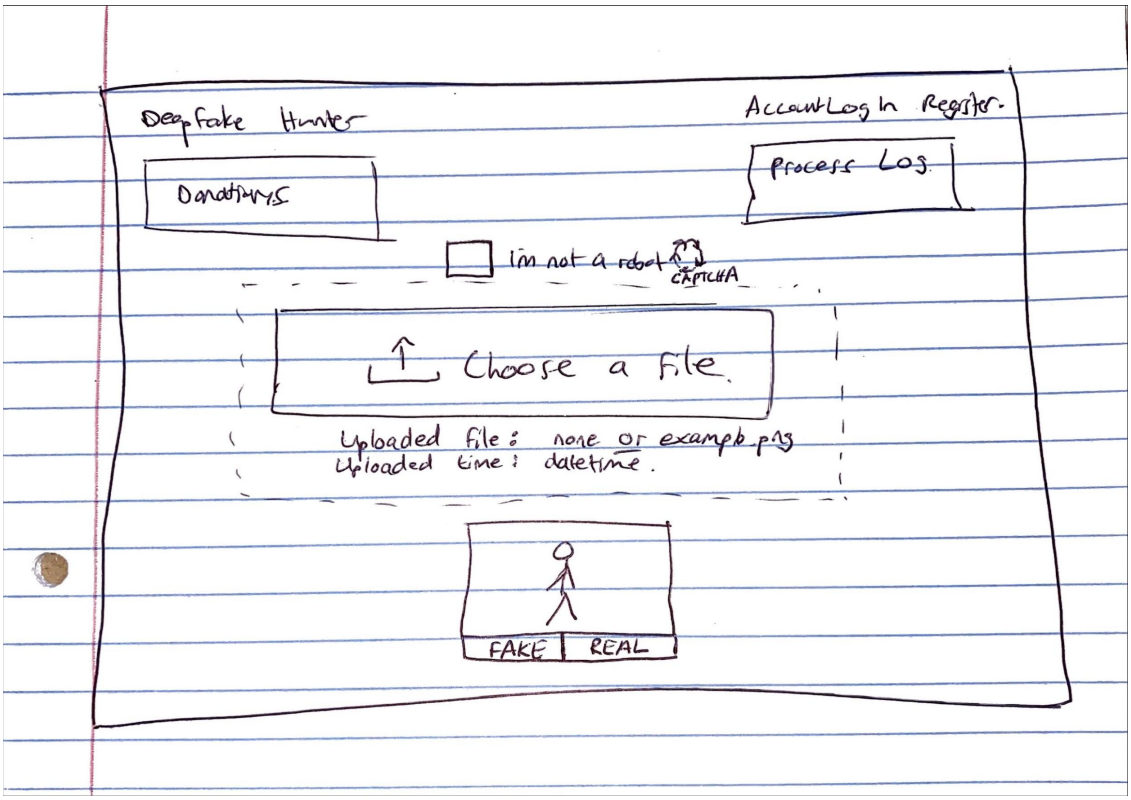


EIE Design 4 AM
Meeting 3: 8 August 2022 Software / G9
Topic:
Location: BBB
Attendees: same.
Agenda: Last week focus: deciding server vs serverless? selecting Lambda vs EC2 for API's
selecting Aurora for hosting (MySQL)
Metrics: • setup and maintenance of a server
• scalability
• pricing.
• latency? Hosting is fine, API only slow coldstart.
• Publish and subscribe architecture? Like news feed on client window. Possible RSS feed showing deepfakes?
Cost of system: operating and development costs. Legal? consider disclaimer, insurance, cost of legal.
• How in depth do you want AI to be explained.
• learning set of filters. • Use research papers.
• Security: scripts? Change header to trigger script.
NB! UML, Data Flow, schema
UI → see screens, maybe Figma. simple and functional.
Block diagram of UI experience of, in Data Flow.
what is a data flow diagram, rep. processes.
what is false notif? show users what people think.
Confidence of decision? Report to UI, switch between if conf. is low.
→ AI + security, size of neural network?

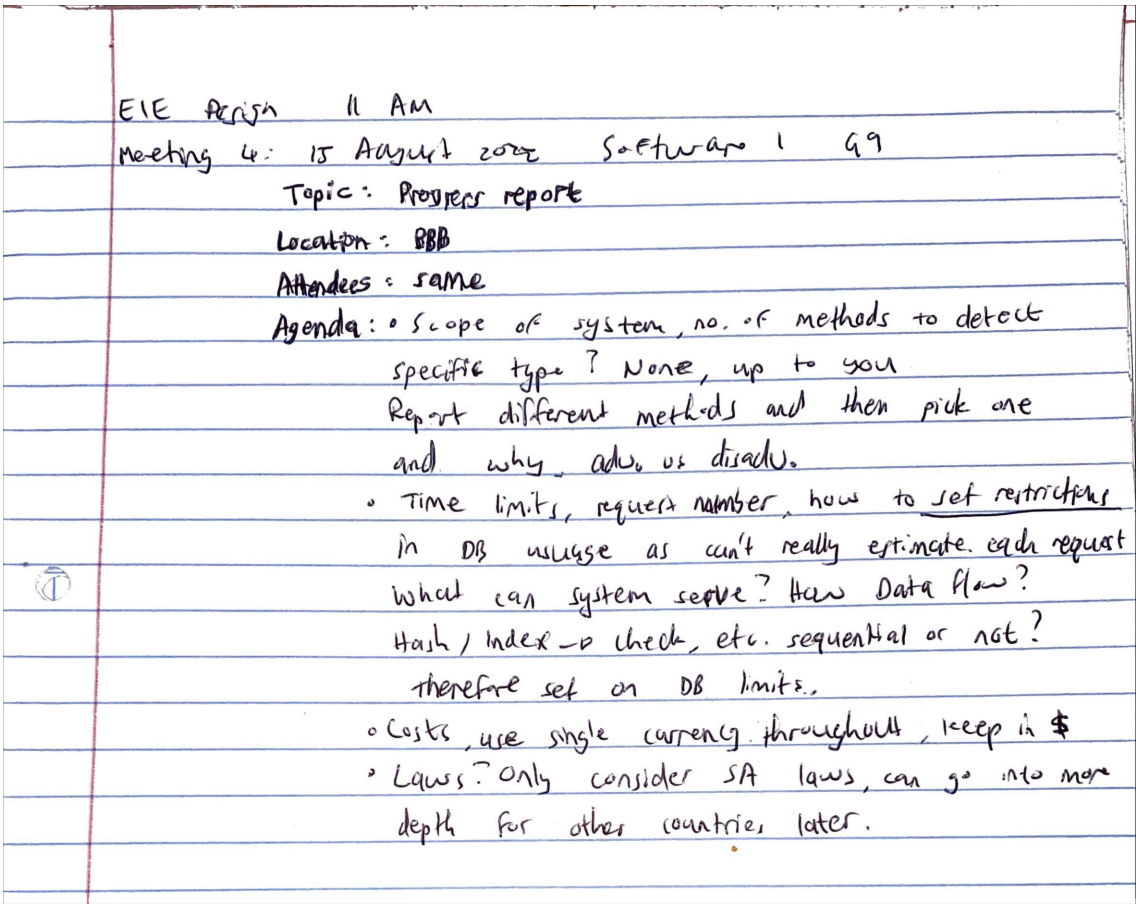
Week 4: 15 August 2022

Summary: A research focused week into aspects such as face detectors.

MTCNN is a python (pip) library written by Github user ipacz, which implements the paper Zhang, Kaipeng et al. “Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks.” IEEE Signal Processing Letters 23.10 (2016): 1499–1503. Crossref. Web.
S3FD
https://openaccess.thecvf.com/content_ICCV_2017/papers/Zhang_S3FD_Single_Shot_ICCV_2017_paper.pdf
Noisy student https://arxiv.org/abs/1911.04252



Meeting notes for this week will be presented below. These are not neat and is more used to write down any and everything that comes to mind.


EIE Area 11 AM
Meeting 4: 15 August 2022 Software 1 G9
Topic: Progress report
Location: BBB
Attendees: same
Agenda: • Scope of system, no. of methods to detect specific type? None, up to you
Report different methods and then pick one and why, adv. vs disadv.
• Time limits, request number, how to set restrictions in DB usage as can't really estimate. each request what can system serve? How Data flow?
Hash / Index-D check, etc. sequential or not?
therefore set on DB limits.
• Costs, use single currency throughout, keep in $
• Laws? Only consider SA laws, can go into more depth for other countries, later.

Week 5: 22 August 2022

Summary: A week focused on structuring the architecture of the system. Progress made, goal is to complete it next week.

Aspects such as hash collision can be prevent largely by using 64 bit sha-3 hash functions. https://crypto.stackexchange.com/questions/1170/best-way-to-reduce-chance-of-hash-collisions-multiple-hashes-or-larger-hash

Meeting notes for this week will be presented below. These are not neat and is more used to write down any and everything that comes to mind.

EIE Design 11 AM

Meetings 5: 22 August 2022 Software 1 G9

Topic: Progress report

Location = BBB

Attendees: name

Agenda: Main task: WHY for each
what is available? why choose? More detail on choice.

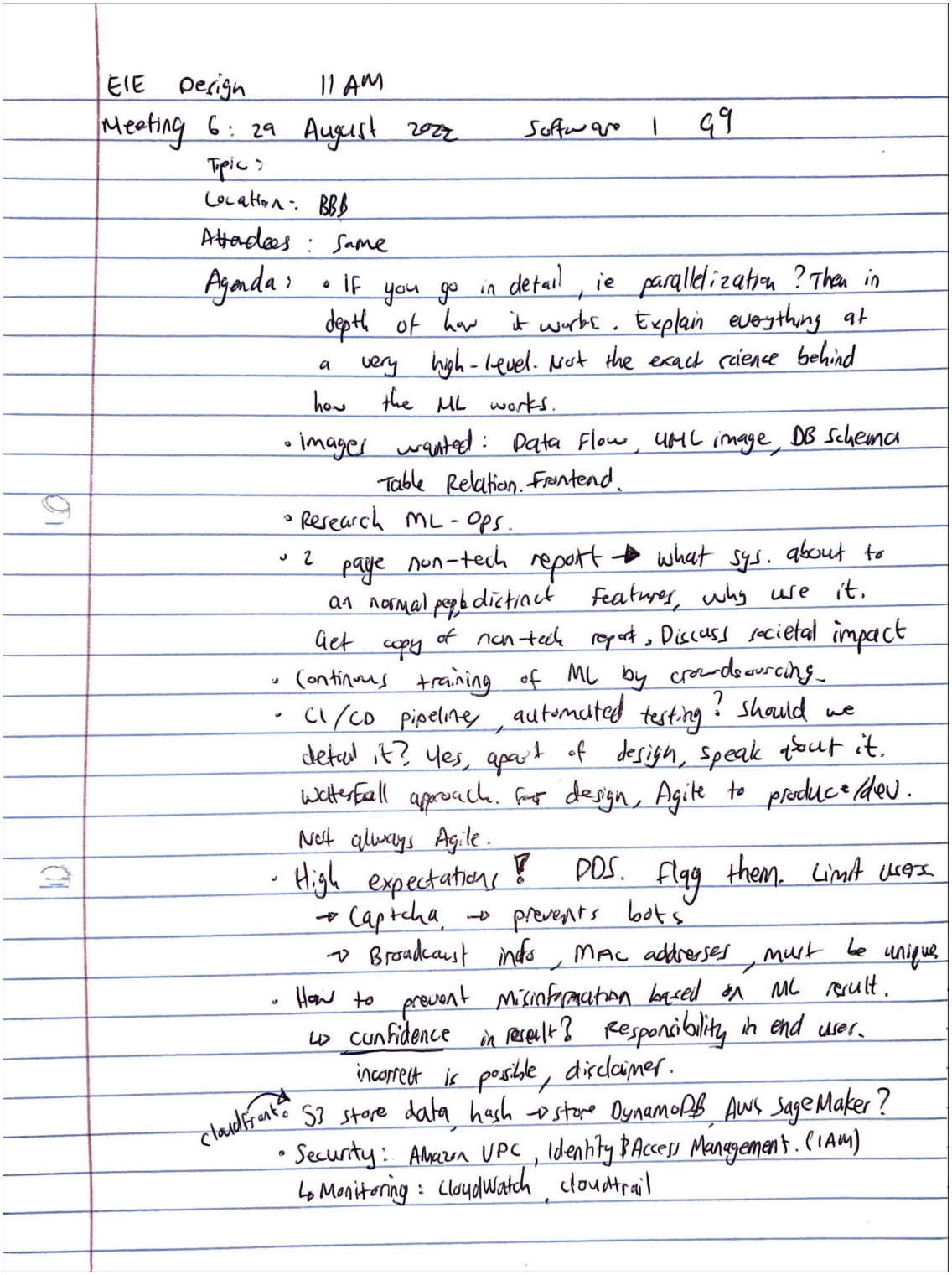
Detection → pytorch., refers to each but not detailed for each

- want to see system architecture, how they all connect with each other.
- To estimate users at a time? How?
  constrained by architecture of software, servering X users at a time, why is it so? what assumptions..?
  Possibly research backed. Should not be random → must be reasons behind it.
- Use blockchain to prevent attacks, store it.
- Put UI screenshots in report. Use data flow diagram.
- Blockchain vs ~~database~~ for storing results.
  Use SHA2 and SHA3 hash functions to avoid hash collisions.
- Use captcha for users input. Create money pool.
  Give people money for detecting images,
  Hybrid intelligiance. Not logged in → research
  active learning Logged in → crowdsourcing.
  Deepfake daily → 5 images → guess answers. (known)
- Research ML.ops. → use directly possible → research!
- Income generation. → ~~data~~ donation etc., pay for service to see how/why.
  ↳ brand. → therefore possibly license out technology
  ↳ payee stores images that was processed and results
  ↳ can be shared.
- User input → upload.

Week 6: 29 August 2022

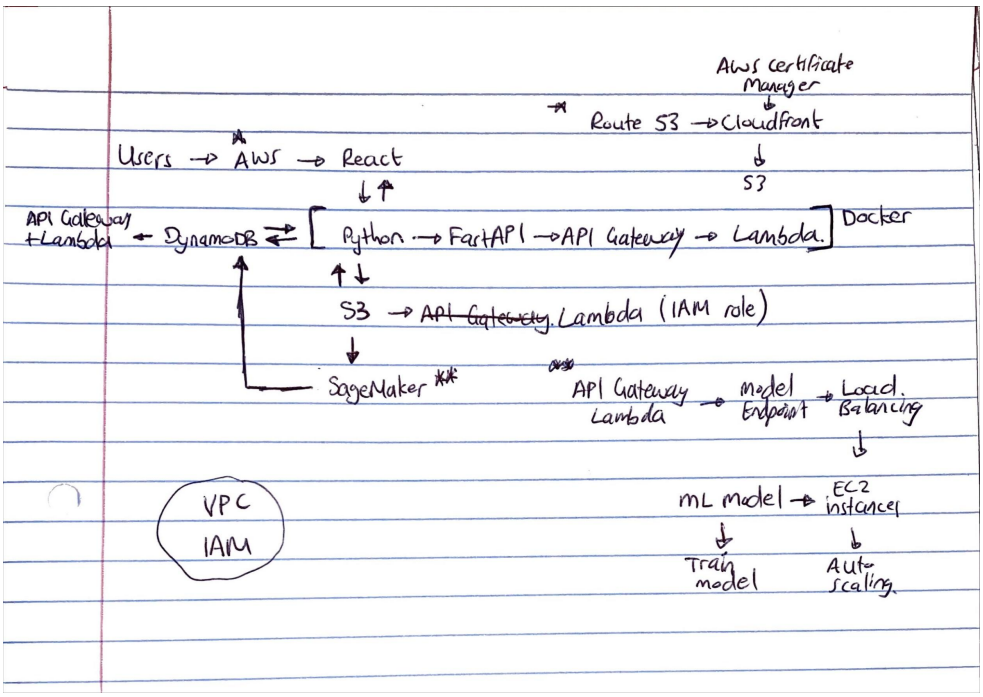
Summary: The completion of the system architecture was the main focus for this week along with designing the frontend, this will be completed next week.

Meeting notes for this week will be presented below. These are not neat and is more used to write down any and everything that comes to mind.



EIE Design 11 AM
Meeting 6: 29 August 2022 Software 1 99
Topic:
Location: BBB
Attendees: Same
Agenda:
- If you go in detail, ie parallelization? Then in depth of how it works. Explain everything at a very high-level. Not the exact science behind how the ML works.
- Images wanted: Data Flow, UML image, DB schema Table Relation. Frontend.
- Research ML-Ops.
- 2 page non-tech report → what sys. about to an normal ppl distinct Features, why use it. Get copy of non-tech report. Discuss societal impact
- Continous training of ML by crowdsourcing.
- CI/CD pipeline, automated testing? should we detail it? Yes, apart of design, speak about it. WaterFall approach. for design, Agile to produce/dev. Not always Agile.
- High expectations! DOS. flag them. Limit users → Captcha → prevents bots → Broadcast info, Mac addresses, must be unique
- How to prevent misinformation based on ML result. ↳ confidence in result? responsibility in end user. incorrect is possible, disclaimer.

cloudfront: S3 store data, hash → store DynamoDB Aws SageMaker?
- Security: Amazon VPC, Identity & Access Management. (IAM)
↳ Monitoring: CloudWatch, cloudtrail



Within this week a consultation meeting was set up. The notes from that meeting will be included below.

EIE Design 9 AM

Consultation 2 : 1 September 2022 Software 1 G9

Topic : Report check-up

Location : Teams

Attendees : Prof + myself

Agenda : check if report setup is good.

Skillset gained from software.

Holistic approach.

- Use case, distinct feature., gain depth

  cache similarity

  Measure data flow, computation..
- Design by testing. S.D. method

  ↳ Use case scenario. of implementation

  what test would put in place

  ↳ List of unit test, what can do. expected?

  Reliance of DB at time + snapshot.

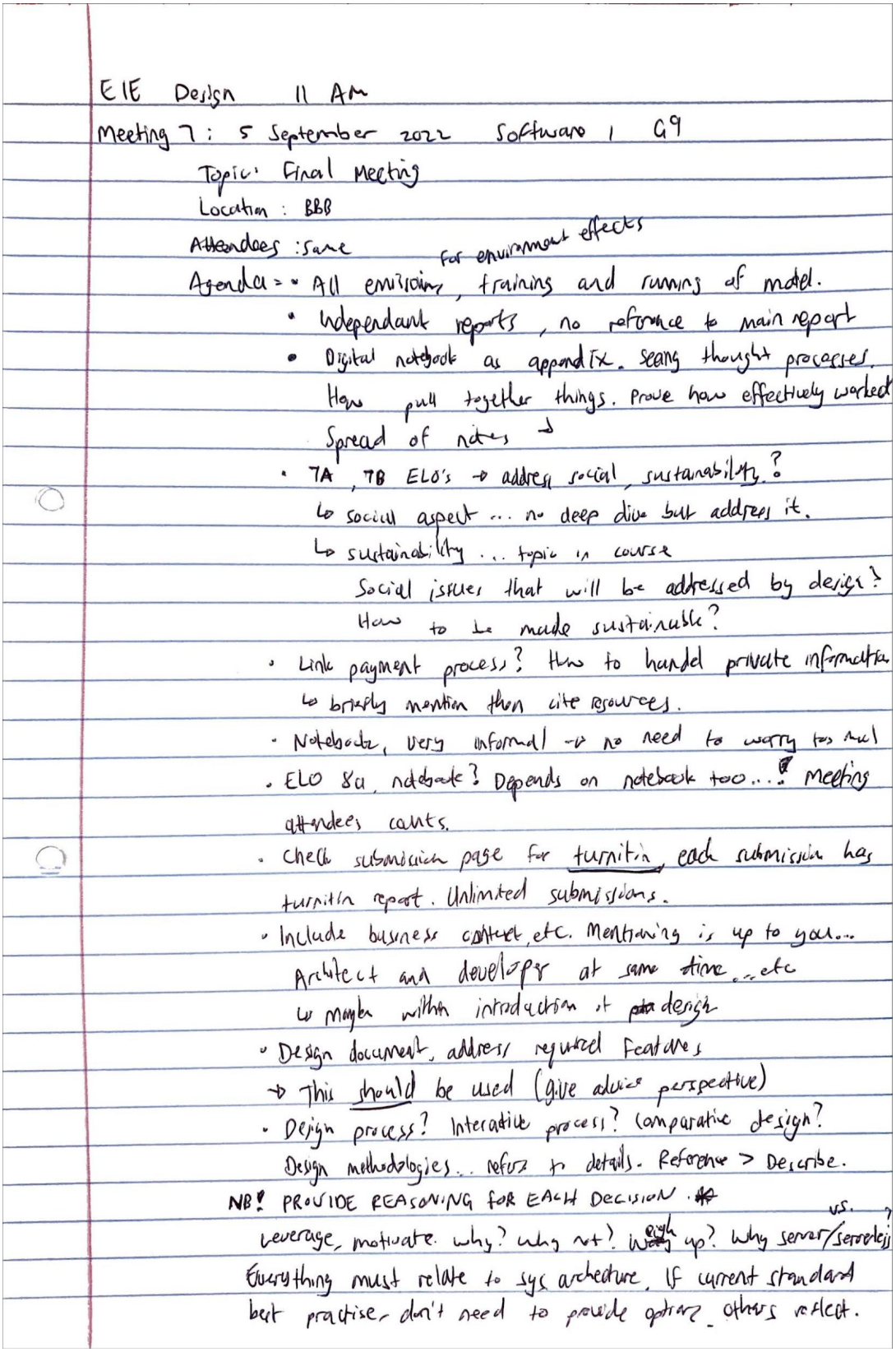
  what tests can be used to check ML model. security of DB.

  Test on adversarial attacks.

Week 7: 5 September 2022

Summary: The final week. Submission of friday. Completion of the report is the main focus.

Meeting notes for this week will be presented below. These are not neat and is more used to write down any and everything that comes to mind.



EIE Design 11 AM

Meeting 7: 5 September 2022 Software 1 G9

Topic: Final Meeting

Location: BBB

Attendees: same

Agenda: • All environs for environment effects, trainings and runs of model.

- Independant reports, no reference to main report
- Digital notebook as appendix. seeing thought processes. How pull together things. Prove how effectively worked
  Spread of notes →
- TA, TB ELO's → address social, sustainability?
  ↳ social aspect ... no deep dive but address it.
  ↳ sustainability ... topic in course
  Social issues that will be addressed by design?
  How to be made sustainable?
- Link payment process? How to handel private information
  ↳ briefly mention then cite resources.
- Notebooks, very informal → no need to worry too much
- ELO 8a, notebooks? Depends on notebook too...! Meeting attendees counts.
- Check submission page for turnitin, each submission has turnitin report. Unlimited submissions.
- Include business context, etc. Mentioning is up to you...
  Architect and developer at same time... etc
  ↳ maybe within introduction of design
- Design document, address required features
  → This should be used (give advice perspective)
- Design process? Interactive process? Comparative design?
  Design methodologies... refers to details. Reference > Describe.

NB! PROVIDE REASONING FOR EACH DECISION.

Leverage, motivate. why? why not? weigh up? Why server/serverless vs. ?

Everything must relate to sys architecture, If current standard best practise, don't need to provide options, others reflect.