Sheffield Hallam University

Faculty of Science, Technology and Arts

Department of Computing

Project (Technical Computing)

[55-604708]

2019/20

Author:	Bartosz Jurus
Student ID:	26009240
Year Submitted:	2020
Supervisor:	Martin Cooper
Second Marker:	Dave Thornley
Degree Course:	BSc Computer Science Hons
Title of Project:	AUTOMATIC SOCIAL MEDIA MANAGEMENT FOR CONTENT CREATORS

Confidentiality Required?

NO □

YES □

AUTOMATIC SOCIAL MEDIA MANAGEMENT FOR CONTENT CREATORS

22/04/2020

Bartosz Jurus
b6009240@my.shu.ac.uk

Abstract

As the society embraces internet more and more, the rise of user on social media platforms increases. Some of these platforms focus on interaction with audiences using videos. Users whose videos are popular are able to monetise their content allowing them to earn money and make a living out of uploading videos to their fans worldwide.

However, there have been events which showed how unstable the income is as the relationship of power is heavily leveraged to the platform advantage ("The golden age of YouTube is over", 2020). These events show how dangerous it is to stake a whole livelihood in only one place.

This report will investigate ways in which this can be remedied by allowing content creators to spread their content across multiple platforms allowing for multiple revenue streams by automating interactions with multiple platforms.

Table of Contents

Introduction

This project aims to create a website allowing content creators to upload a video to multiple websites simultaneously and share the URL of uploaded video with the fans. To achieve this the project will utilize Application Programming Interfaces provided by the biggest social media platforms like YouTube, Facebook and Twitter. Because this website could be used by large amount of people, scalability of the solution is important to accommodate sudden spikes in usage.

While achieving the goals set out above a new technology called Blazor will be used to provide dynamic User Interface. Additionally, another new technology gaining traction created by Facebook called GraphQL will be used to interface with the database. Furthermore, the quality of the code will be measured and is required to be above 65 out of 100 for all parts of the solution enforcing good practices and well-structured project. To ensure that the solution is resilient a goal of 90% code coverage by unit tests has been established. In order to allow flexibility in hosting the server the .Net Core will be used to create a cross platform solution.

Research

This section of the report covers different decisions that had to be considered for key parts of the overall project. All parts investigate different aspect of the system which will influence what technologies will be chosen and systems used. Furthermore, research provides good opportunity to avoid going down the wrong path and learning from mistakes done by other people.

Analysis of similar products

transferring-videos.com

After investigating product that may fulfil similar role which this project aims to achieve, a website www.transferring-videos.com was found. One of the biggest advantages of the software is allowing its users to transfer already uploaded videos to other platforms. This is especially useful to someone who has been uploading for extended periods of time. If the author did not store original copies there is no need to download them to upload them to a different site. However, the issue with this approach is that the video has been processed by one platform already. This leads to a worse looking video than the original upload as most platforms do additional processing on the video ("VIDEO ROOM 1000 COMPLETE MIX -- All 1000 videos seen in sequential order!", 2020). Another limit of this website is locking the quality of the video to SD or HD depending on the subscription level chosen.

Furthermore, there is no way for a user to share their videos automatically after upload. If a user would set out to do this, they would have to manually go to each site to copy URL and then manually go to each social media to post the links to their fans.

During exploration of the website one jarring flaw was revealed in the upload process. When YouTube was chosen as the destination a pop up with 14 steps has appeared asking user to create and enter client id and secret for OAuth communication for this website to use for uploading of the video. This approach seem not user friendly and this project will need to make sure to avoid solutions like this.

SyVid

Another website that allows for distribution of videos to multiple platforms is SyVid. Unfortunately, it is not possible to have access to the website without paying first so the analysis is based on claims of the authors and screenshots available on the internet. From the main website it looks very close to what this project aims to achieve. It has a modern looking user interface and allows for sharing links of uploaded media. There are two main issues that can be seen without using the website.

Firstly, the website main demographic seems to be people working in marketing and not content creators. This could make potential users look for other alternatives because their specific need might not be catered to.

Secondly, there is a lack of integration for newer social media like discord which in 2019 surpassed 250 million users (Koetsier, 2020). As discord has a big userbase this project will attempt to integrate it into the final solution.

Conclusion

Both of the products show that the aims set out in this project are achievable and each product has flaws that can be improved. The SyVid is particular is a stand out in terms of features it provides and wide range of platforms that are integrated into it. The insight gained by looking at these product will aid in steering this project to avoid their problems while trying to learn from their positives.

One of the noteworthy outcomes of this research is a realization that both solutions are webpages rather than a standalone desktop application. One of the reasons why this might be the case is to increase their outreach by allowing people using different operating systems to use their programs without producing bespoke solution for each system. This further verifies that a website is the correct approach for achieving the objectives set out by this project.

Internet requirements

As this project aims to save time by automatically propagating the videos in the web it is important to understand how long the process would take if done manually. According to Ofcom ("UK home broadband performance, measurement period November 2018", 2020) in 2018 the average upload speed of home broadband was 7.2 Mbit/s. To be able to find average upload time of a video a length is required. A report by J. Clement (2020) suggests that depending on a category of video and average length is between 6.8 to 24.7 minutes. Figure 1 shows estimated upload time for an upload of a single video using the 7.2 Mbit/s upload speed. As can be seen from the chart a long 2160p video can take up to 39 minutes to finish an upload (see Appendix B for exact data). If a user would want to upload the video to other platforms the total time would be multiple time larger. However, it is important to notice that there are still a lot of people in the United Kingdom who are on much slower speed than that which exacerbates the issue even further.

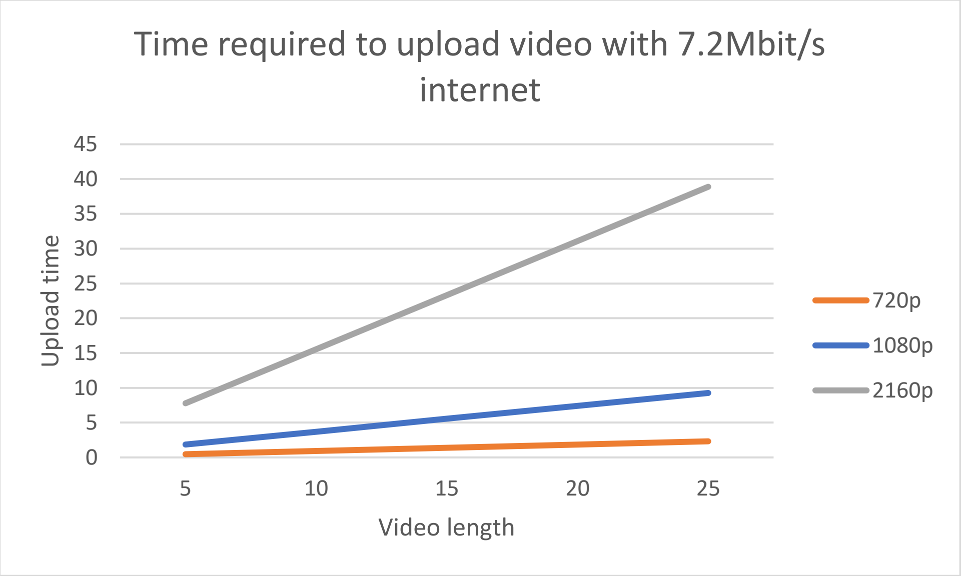


Figure 1. Graph showing time required to upload different videos

Cloud providers

There are multiple cloud providers that this project could use to scale with demand. Among these the biggest ones are Amazon Web Services, Google Cloud and Microsoft Azure. All of these providers have the services needed to accomplish the goal of this project. This makes the decision of chosen provider not important in the short term, but it still is important to consider long term consequences before a decision is made. To decide multiple factors were taken account.

The first one is market share where AWS had 42% in 2019 (Stalcup, 2020). Market share is a good indicator of acceptance and preference for a product, meaning that nearly half of companies prefer to use AWS over its competitors. Moreover, market share can used to gauge how much help can be expected from other customers. Large market share increases the likelihood that any problems occurring during development were found and solved by someone else already.

Another metric considered is downtime. This specific metric is important because as a business it is important that the customers are always be able to use the service offered without gaps in availability which can cause them to move to other competitors. The data shows that in the period between 2015 to 2017 AWS had 448 minutes of downtime in comparison to 1652 and 506 by Microsoft and Google respectively ("How AWS Stacks Up Against Rivals on Downtime", 2020). Furthermore, in the period between 2018 and 2019, AWS was first again with Google Cloud being close second in terms of lowest downtime ("Comparing the cloud giants: Uptime and reliability", 2020). However, in mission critical situations multiple providers should be utilized to lower potential downtime by having a backup. Due to above reasons Amazon Web Services will be used to provide scaling to this project.

An alternative to both approaches is to buy the servers and take care of hosting the website independently. This approach has benefits like independence and complete ownership of the service. However, once rapid spikes of interest happen it may become difficult to buy and set up new servers in order to keep the experience of using the service pleasant. If the spike is temporary the business might end up with multiple servers that are no longer needed. An example of such spike would be Christmas when people are buying a lot of gifts online and the websites selling items are experiencing higher traffic than through the rest of the year. Furthermore, this approach requires significant upfront investment which can be too much to bear for an emerging business.

Project Management

Waterfall

The waterfall approach is the traditional way to develop an application. It consists of gathering requirements from the user. After all requirements have been finalized a design is created specifying all aspect of the solution. The development process uses said design to create a deliverable which is then tested against requirements defined in the first stage of the project. The final stage is maintenance which involves fixing bugs that slipped through testing. This approach is quite beneficial when there is limited time for delivery of the solution as it imposes strict deadlines which cannot be altered. Furthermore, because the requirements are set in stone it prevents overengineering of the solution to predict all possible areas where new requirements will need to be accommodated. This stable base for the project also make verifying that the set out aims have been meet is easier in contrast to Agile where the requirements need to be reviewed and changed as the needs of the business/stake holder change ("The Pros and Cons of Waterfall Methodology", 2020).

However, this is also a detriment as it doesn’t allow any flexibility and going back to a previous stage can be costly in comparison to other approaches which embrace change (Alshamrani & Bahattab, 2015). This also makes developing a solution a difficult endeavour as the system needs to be designed at the beginning.

Agile

The agile methodology is an opposite of the waterfall approach. There are multiple systems under the agile banner with the standouts being Scrum and Kanban. There are major differences when it comes to the approach to the sprints for both of them. However broadly speaking Kanban main focus is on a persistent board across sprints with limited items that can be simultaneously worked on. In Scrum there is a larger overhead required as items are added to the board for the duration of a sprint. Each item on a board has to be able to be finished by the end of the sprint and each team member has to decide their commitment level to the sprint. The items are also marked with score called story points that represents amount of effort and time needed to finish said task. This quantization of sprints allows for better estimates of future sprints and ability to compare performance between sprints. Another benefit of agile is higher success rate of projects. As Schwaber & Sutherland (2012) point out in their book, based on data from CHAOS project database Agile projects are 3 times more likely to succeed than Waterfall projects.

However, one of common complaints raised against Agile approach is that the documentation is often lacking and created as an afterthought. This is could be fine for short lived projects but for longer projects when the developers who created the system get replaced by new hires, the insufficient documentation may become an issue directly impeding work. Another issue is difficulty of predicting results of the project. It is often not clear how many sprints developing the program will take, and this methodology doesn’t provide good means of predicting length of the whole project. This can oftentimes cause budget overruns (Frank & Carmi, 2014). Once the works start there are ways to use past performance in order to estimate how much can be done in a sprint using things like momentum and story points, but long-term planning is still difficult.

Tracer bullet development

The term Tracer Bullet Development was coined by David Thomas and Andrew Hunt in the book The Pragmatic Programmer. Its key idea is that when a system is created all the elements should be able to interact with each other as quickly as possible. The rationale for that is even though the class might be functional based on design until all the units are integrated there is no evidence justifying the belief that the system works as expected. This approach also has an additional advantage of quick visible improvements as the features go from user interface all the way to the back end when implemented (Thomas & Hunt, 2020). This means that after a sprint the feature can be provided to the client for evaluation. A key advantage is that the client will be able to quickly spot misunderstandings that might have been made when gathering requirements (Richardson & Gwaltney, 2005). This in turn allows to rectify the wrong assumptions while the solution is still in early stages rather than when the development has long stopped. Furthermore, this approach can also be incorporated with an Agile approach which already heavily depends on client involvement and iterative development.

Conclusion

The project management chosen is Agile, utilising Kanban and tracer bullet development. Agile has been chosen, because this project involves a large amount of development in a not well researched scenario and it is unfeasible to assume that all of the pitfalls can be predicted before development. The agile approach will add flexibility required in working with lack of domain knowledge and with addition of tracer bullet development it will allow quick understanding if there are any mistakes in the architecture of the project.

Issue Tracking

Trello

Making sure that different tasks and bugs found are not forgotten is a common problem. Over the years there have been multiple attempts to tackle this problem. One such solution is a Trello board inspired by Kanban it is a project board were many items can be placed within different swim lanes. The items can be moved between swim lanes indicating progress in completing these tasks. This type of visual feedback is beneficial when trying to identify how much of sprint/project is done and how much work is still needed. Trello has clean user interface and allows users to have multiple boards which can be used in larger teams where each team would have their own separate board. Another positive side of Trello is the integration with GitHub which allows for automatic creating of items on board when a new issue is created in the repository. Small thing like this remove manual and tedious tasks allowing developers to focus on programming.

Glo Boards

One of the competitors of Trello is Glo Boards which takes integration even further by completely integrating itself into a git client. This means that there is no need for a browser to interact with the project. Tasks such as adding an issue, creating a pull request and committing a bug fix can all be done from the same program. Not only that it also has features that allow you to set milestones for major releases and has powerful filtering system allowing the user to drill down through all of the cards to the ones that are relevant.

Source Control

Subversion

The subversion is a source control system that utilizes a single central repository for managing code. It is currently developed by apache. One of its advantages is the performance with large repositories as only the current working version is downloaded by the client. Another benefit is how well SVN handles large binary files which is unparalleled so far by other version control software. However, this also has a disadvantage of the need for constant connection between the client and the server. Furthermore, subversion requires a dedicated machine hosting it if multiple developers need to work together. Moreover, with modern DevOps approach to developing software the architecture is not lending itself well for automation which is partially due to waning community base.

These problems might prove a deciding factor due to extra expense and set up needed to start working with subversion. Although, this might prove a worthy investment if the project is expected to be taking place over multiple years where the performance is a big factor.

Git

The most popular among developers tool for source management is git. A website recording data of open source project show that at the time of writing this article git is used in 71% of all open source projects ("Compare Repositories - Open Hub", 2020). A survey conducted in 2016 by RhodeCode also found that 87% of developers prefer Git over other version control systems ("Version Control Systems Popularity in 2016", 2020). Git is a distributed source control system that allows each user to store a copy of the whole repository and its history. This has an advantage of allowing users to see previous revisions and go back the history of commit locally without the need for internet connection allowing to remain productive under internet outages or during travel.

Conclusion

For this project git will be used due to being well integrated into modern tools with the possibility to be added into a continuous integration/continuous deployment pipeline. Such solution will also work well with the “fail fast, fail often” mindset of agile methodology as long as well-designed tests are present. This is the case because the CI/CD pipeline should detect any bugs present in the new version of the software and raise an alert forcing the developer to fix the bug introduced. The program chosen to be the git client is GitKraken as it not only is a fully featured git client, it also integrates issue tracking along with other features.

Technologies

C# .Net Core

The C# language was chosen as the programming language of choice because it is one of the most popular languages with mature ecosystem and is constantly improved by Microsoft. It is one of the most popular programming languages according to the survey performed by stack overflow ("Stack Overflow Developer Survey 2019", 2020).

There are two main branches one can use when developing with C#. The older one is .Net Framework. It has been the main way to program using .Net since its release in 2002 as Visual Basic has been slowly phased out since. Its main advantage is wealth of libraries which have been created in the ecosystem. The .Net Framework additionally allows for developing graphical user interface programs on Windows. This is also one of the flaws as there is no way to utilize all the libraries on any other operating system. This is what led Microsoft to develop .Net Core which solves exactly this issue. It trades off wealth of the libraries and ability to develop GUI native application in favour of cross compatibility. This new development allows targeting any supported operating system and compile the program for it.

Entity Framework Core

The Entity Framework Core is a cross platform library that allows interactions with a database of choice purely using C#. It is especially great because it allows the use of models that represent tables in database as the schema, meaning that any change to a model also modifies the corresponding table. It has another benefit of allowing developers unfamiliar with writing SQL queries to interact with the database in a familiar way as from their point of view the database is just another class in the system.

ASP .Net Core

The ASP .Net Core was chosen as it is the only cross platform option of hosting dynamic webpage in C# ecosystem. However, that does not mean it is lacking features or that it has bad performance. On the contrary it contains all functionality required to develop this application and plenty more, should they be needed in the future as this solution is expanded upon. The performance also is impressive as it ahead by a lot of other competitors such as NodeJS on benchmarks where each holds 3rd and 52nd place respectively ("TechEmpower Framework Benchmarks", n.d.). What is interesting is that some benchmarks found that performance was not sacrificed when porting ASP to be cross platform and in most cases both version are close to each other in terms of performance.

Identity

Identity is a library which handles Authorization and Authentication of users. It is the preferred way of solving these two issues in .Net ecosystem. Because it was created and is constantly improved by experts there is a high degree of confidence that it is a secure system. It also abstracts all of the detailed and error prone aspects of hashing passwords, checking if user is logged in which speeds up development and prevents the developer from creating security problems.

Blazor

Blazor is a new technology developed by Microsoft allowing for running C# code on in the web browser. There are two modes available client side and server side Blazor. Client side Blazor compiles all of the dependencies using Web Assembly and allows the browser to run the code. It is good because it offloads majority of work to the client machine. However, this could also be a negative if the typical user has a slow computer. This should not apply to the users of this website as editing and rendering videos is very resource demanding. Additionally, if a content delivery network is used there is a chance that the Blazor Web Assembly code has been cached speeding up the load times.

The server side Blazor lets the server run the code and transport the rendered component to the user. This means that even websites with a lot of complicated components can still be run on weak computers increasing the number of potential users. Overall, the sever side version will be used as it's the only version currently supported. The client side is still in development and the first stable release is planned for May 2020.

Design

User interaction

Before creating the user interface, the tasks that a user would want to accomplish were considered and following were identified:

1. Register
2. Log in
3. Add a social media platform to the account
4. Upload a video to one or more platforms
5. Share links of the videos to text based social media

There have also been activities that are considered out of scope of this project:

1. Password recovery
2. Password change
3. Modifying specific settings on platform by platform basis

Given the essential goals that the users might have, happy paths were created reflecting them. A happy path is a set of steps that represent the default way in which the user will interact with the system. These paths are used to identify flow between different pages in the system and can be used for testing purposes to verify that the users can operate the system as expected. For the purposes of this project it is expected that the user will use keyboard and mouse to navigate the website. Mobile devices and touch screen interaction were considered out of scope of this project.

Architecture

Overview

One of the key aspects of the project was to create a solution which will be scalable and easily extendable. This remained key when designing the application and attempting to identify and remove all bottlenecks. Figure 2 demonstrates the infrastructure of the services in the cloud as well as the order of operation during upload operation. Following sections will cover each element, explaining its purpose and why this particular service was chosen.

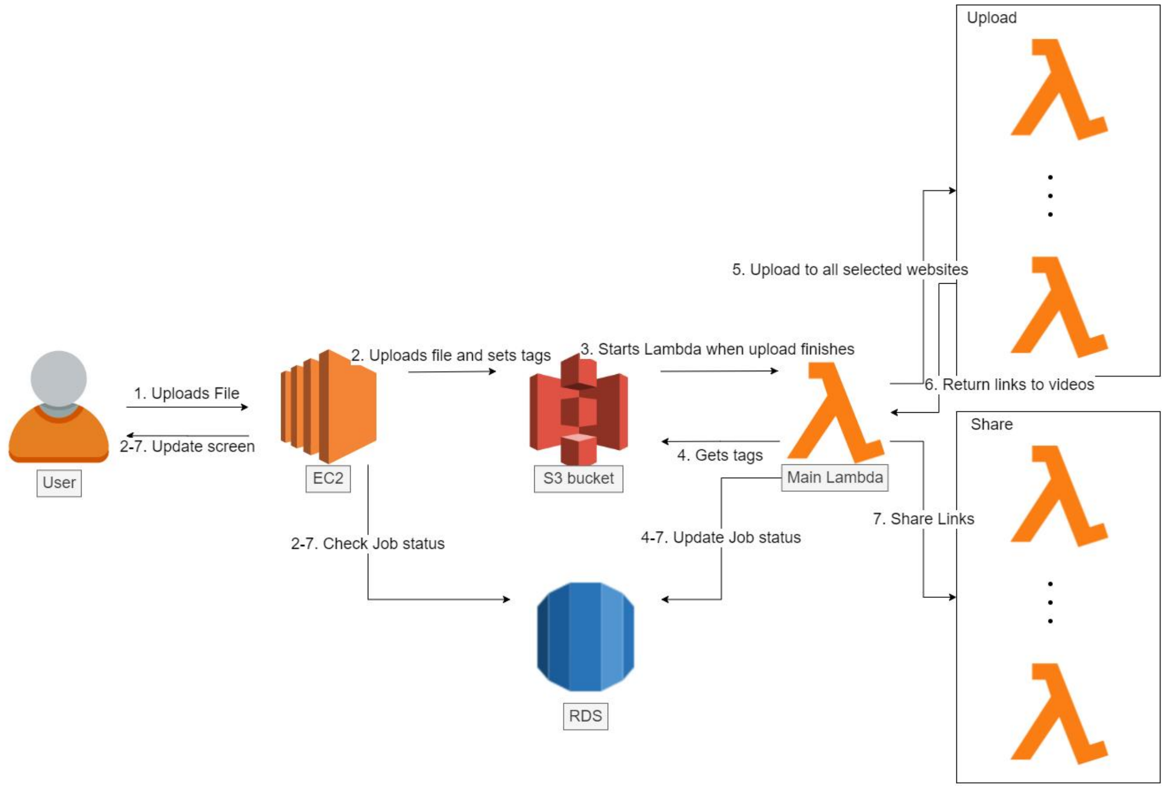


Figure 2. Diagram showing the architecture of the system.

Elastic Compute Cloud

The EC2 allows for scalable general-purpose computing environment. Its main role in this application is hosting a persistent ASP.Net Core server which provides user facing interface in a form of a website and is the only way for user to interact with the system. The performance can be improved by changing the type of the EC2 instance to one with higher core count and/or RAM. Alternatively, another instance can be created but due to limited budget of this project this have not been covered by this solution.

An alternative to an EC2 is a lambda function which is also used in this project. The main reason why it has not been used is due to extra time it takes for it to get ready to run code. This leads to noticeable delay when navigating to a website, negatively impacting user experience.

S3

The S3 service is what is commonly referred to as object storage which means that the data is treated as object. This has key advantage as opposed to different storage architectures which is better interface that can be directly programmable by applications ("Object storage", 2020). This service is also important due to ability to set tags on objects. In this solution they are utilized to save information, which services use to know where to upload the video and where to send final message. Moreover, it automatically scales when more objects get uploaded removing any need for manual involvement once it is set up (Gulabani, 2017).

The S3 bucket is used as temporary storage of videos to be uploaded for the user. Once the upload to the bucket is finished the bucket triggers Main Lambda function as the file is ready to be further processed. The limit on the files is 5 GB which is around 13 minutes of a 4k video and 38 minutes of

1080p video. The exact size is affected by more than just the resolution, so the figures above are meant as a rough estimate rather than an absolute maximum amount.

RDS

RDS stands for Relational Database Service is the database chosen for this project, because all the data stored in the system is related and structured making the RDS the best choice. Furthermore, this service is compatible with Entity Framework Core which means that it will fit with the design and technology decisions made earlier. This database stores user accounts requires for secure use of the system as well as their different tokens and statuses of requested upload jobs. All parts of the system besides the S3 bucket interact with the database. This is one of the bigger disadvantages of this design as it could potentially cause a bottleneck with large number of users. Figure 3 shows the Entity Relationship Diagram of the database exluding athentication entities genarated by Indentiy.

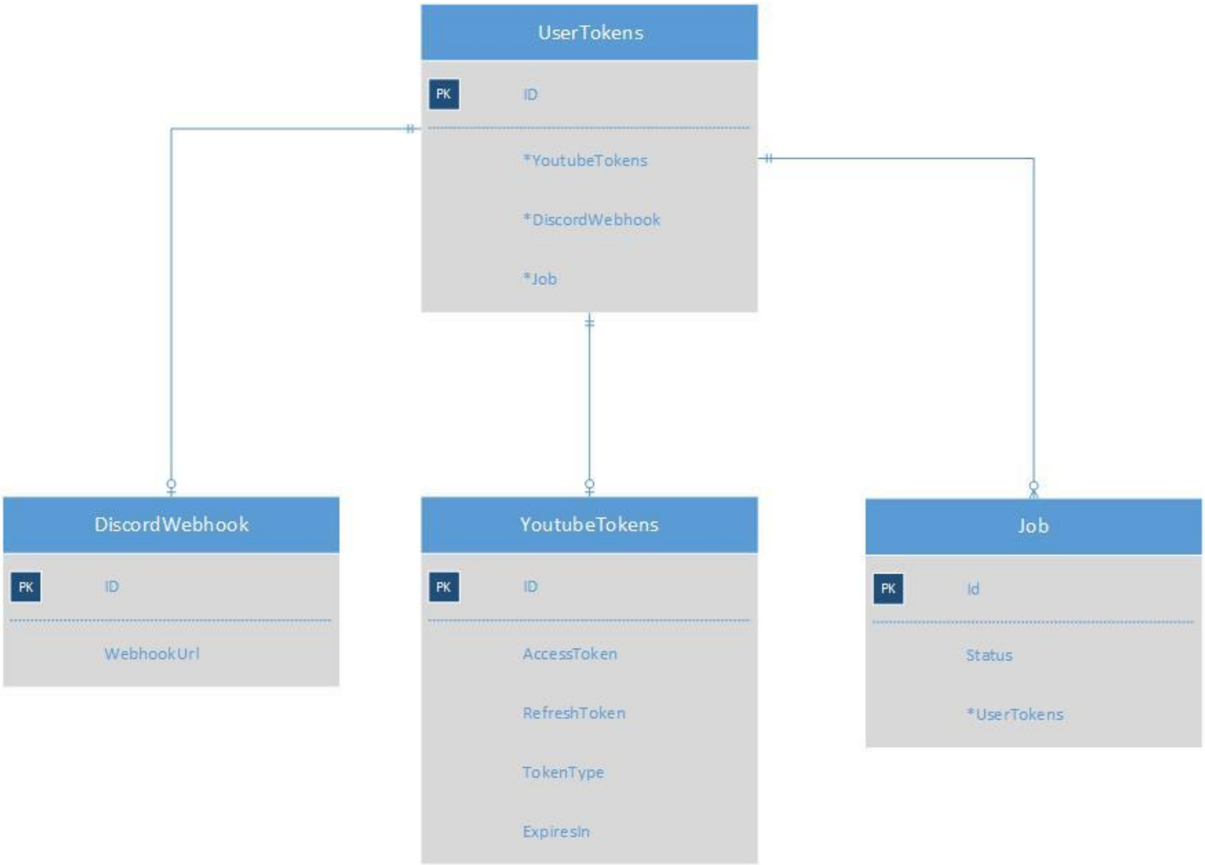


Figure 3. ERD diagram showing relationship between entities in the database.

Lambda functions

There are 3 main groups of lambda functions:

1. Main lambda
2. Uploading lambdas
3. Sharing lambdas

Firstly, when main lambda function is triggered by S3 bucket it retrieves the tags on the item to upload and uses them to modify its behaviour. As it proceeds through execution it updates the status of the upload job in the RDS database. The main two roles of this function are to start the correct lambda functions in parallel manner for uploading of the file to video streaming websites and in similar manner share URLs of the uploaded files to different social media.

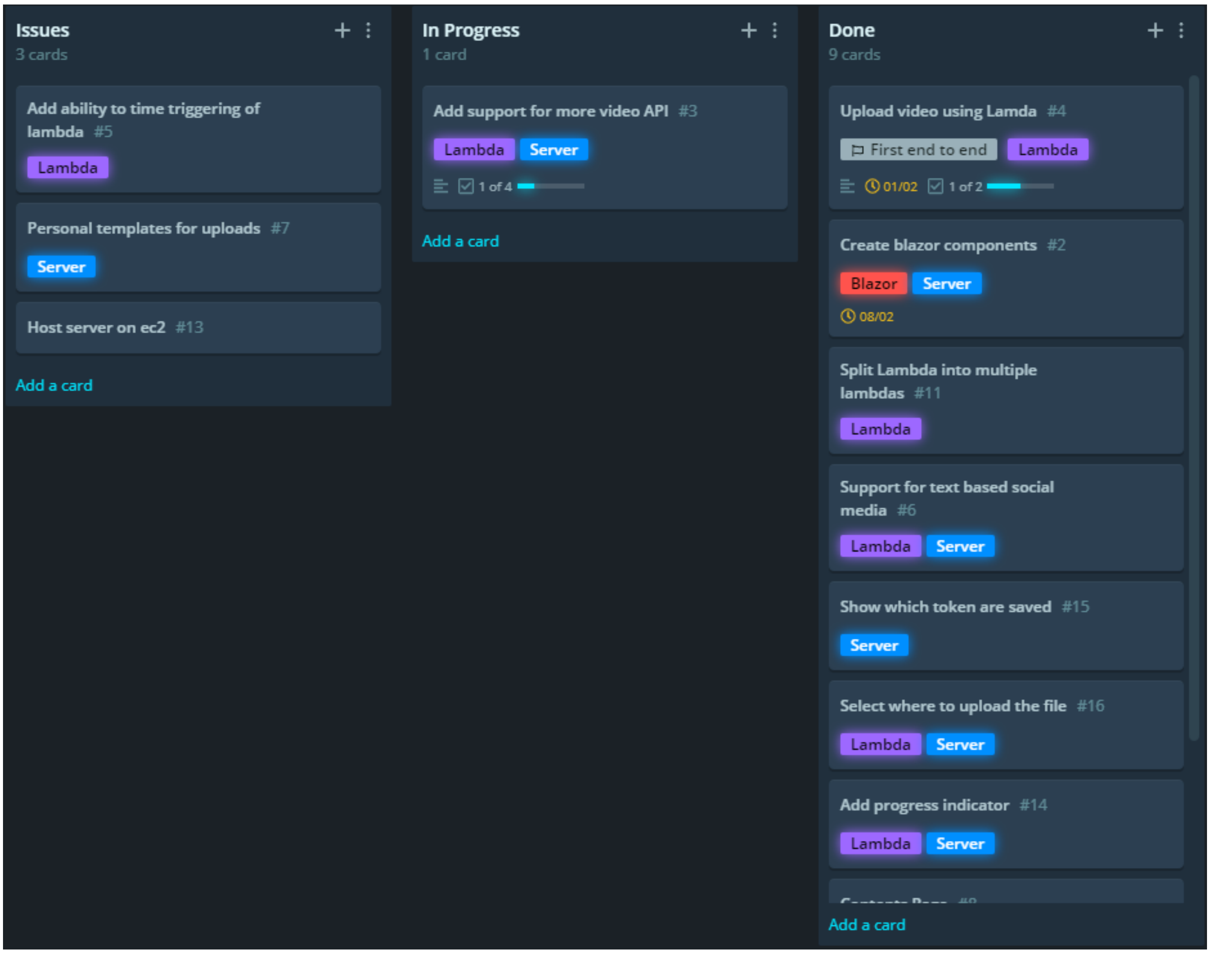
Secondly, lambda functions in the uploading group have a single responsibility of uploading the file to only one predetermined website. They retrieve required data for the upload like access token from the database and after the upload is finished, they return the URL of the video to the main lambda.

Thirdly, sharing lambda functions create posts and send messages to different social media and messaging applications. Similarly, to the uploading lambda functions they also retrieve any necessary data from the database to fulfil their task. Depending on the service this might be OAuth tokens or URL to which a post request needs to be made.

Due to the fact that each of the uploading and sharing lambda is used for interacting with one external service it quite simple to extend the system by creating a new lambda to add support for another service. This is one of the key goals of this project as it can be extended with rise of new social platforms.

Development

While working on producing the deliverable GitKraken and Glo boards were used as planned in the research section of this project. The figure shows how the board looked near the end of the project. All issues were moved from issues columns which was integrated with the GitHub issues towards the Done columns as the project progressed. New items were added when bugs or additional feature ideas were discovered. Figure 4 shows how the look of the board nearing the end of the project.



Issues
3 cards

Add ability to time triggering of lambda #5
Lambda

Personal templates for uploads #7
Server

Host server on ec2 #13

Add a card

In Progress
1 card

Add support for more video API #3
Lambda Server
1 of 4

Add a card

Done
9 cards

Upload video using Lamda #4
First end to end Lambda
01/02 1 of 2

Create blazor components #2
Blazor Server
08/02

Split Lambda into multiple lambdas #11
Lambda

Support for text based social media #6
Lambda Server

Show which token are saved #15
Server

Select where to upload the file #16
Lambda Server

Add progress indicator #14
Lambda Server

Add a card

Figure 4. Kanban board used in the project.

The process of using the git repository was a GitHub flow approach which is a workflow which is well suited for websites which only have one version in production. This approach involves a master branch which should always be in a state where it can be deployed and a develop branch where the developers make changes. Once the develop branch is ready it is merged with master and manually published to a production environment. This way of using git does also allow for a change to other approaches such as Git-flow or Gitlab-flow as the needs change or number of developers working on the project increases.

Connecting to platforms

One of the biggest obstacles of this project is that it is completely reliant on functionality exposed by an APIs of each platform. Different platforms take different approaches to authentication and basic flow of interacting with the API. This section will show how varied the discrepancy can be between them by using two examples.

YouTube

In order to use API provided by YouTube multiple steps need to be taken. Firstly, a developer account needs to be made which will allow for creation of projects in the google developer console. Secondly, after creating a project the next step is to connect it to the APIs that it is going to use. After that has been accomplished the OAuth authentication needs to be configured with allowed URLs, Icons, etc. Upon successful configuration a client id obtained will allow for making obtaining access tokens for the user enabling the use of API.

When the user decides to connect to YouTube the method shown in Figure 4 is run. A builder pattern is used to construct a URL containing client id, requested scopes, return Uri among other which are required by the API. The redirectUri is the method set earlier in the configuration where the user will be redirected to after successful or unsuccessful connection. After the user is redirected to google OAuth service a prompt appears asking if the user wishes to allow this application access to perform actions requested in the scopes.

```
[HttpGet]
public IActionResult YoutubeToken(string redirectUri=null)
{
    var url = new YoutubeTokenUrlBuilder(_configuration)
                .WithRedirectUri(redirectUri)
                .Build();
    return Redirect(url);
}
```

Figure 5. Function used to redirect user to obtain OAuth token from YouTube API.

The scopes are permissions to certain types of functionality provided API. They allow for more granular control of permission and give more power to the user to control how much access do they feel good about granting to a given program. The scope requested by the application is youtube.upload which grants ability to upload a video but nothing besides that. This increases security of the system as even if the tokens were stolen, the only thing that a potential hacker could do is upload more videos. There would be no way to delete videos or access any other private information on the channel limiting among of potential damage.

When the users are redirected back to the server after the operation function showed in Figure 6 is run. The function uses parameters provided in the URL to create and send a POST request which returns the tokens required to interact with the API. The tokens are returned as a JSON object which gets deserialized into a model and stored in the database.

```
[HttpGet]
public async Task<IActionResult> YoutubeTokenRedirect(string code = null, string scope
= null)
{
    using (var client = new HttpClient())
    {
        var body = new Dictionary<string, string>()
        {
            {"code", code},
            {"client_id", _configuration["Tokens:Youtube:client_id"]},
            {"client_secret", _configuration["Tokens:Youtube:client_secret"]},
            {"redirect_uri", "https://dev.local.com/Token/YoutubeTokenRedirect"},
            {"grant_type", "authorization_code"}
        };

        var result = await client.PostAsync($https://oauth2.googleapis.com/token
                                       , new FormUrlEncodedContent(body));
        var json = await result.Content.ReadAsStringAsync();
        var youtubeTokens = JsonConvert.DeserializeObject<YoutubeTokens>(json);
        var userId = User.FindFirstValue(ClaimTypes.NameIdentifier);
        var user = _dbContext.Tokens.FirstOrDefault(x => x.ID.Equals(userId));
        if (user != null)
            user.YoutubeTokens = youtubeTokens;
        else
            _dbContext.Tokens.Add(new UserTokens() {ID = userId, YoutubeTokens =
youtubeTokens});
        await _dbContext.SaveChangesAsync();
    }

    return RedirectToAction("Settings", "Account");
}
```

Figure 6. Function used to save obtained tokens to the database.

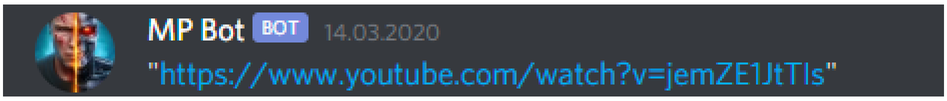
Discord

There is a big contrast with the way that Discord integration has been handled. First of all, there is no need for any tokens to use its API which massively simplifies the process. What is required is that a user adds a webhook to a channel. This is done in Discord and it generates a URL which needs to be entered into the field on the website. To send a message to a channel with a webhook all that is needed is to send a POST request to the URL with correctly formatted JSON body modifying certain fields based on preference such as content field. Before code was written for making the request a software called Postman was used to verify that the request will work. Postman has the advantage of easy modification of the request in contrast to trying to attempt to do it in code from the get go which makes any change required to fix any mistake costly timewise, as each time the program needs to be recompiled before another attempt can be made. The finished function can be seen in Figure 7.

```
public async Task<HttpResponseMessage> SendMessage(string webhook, string message)
{
    using (var client = new HttpClient())
    {
        var body = new Dictionary<string, string>()
        {
            {"content",  message},
            {"username", "MP Bot"}
        };
        try
        {
            var result = await client.PostAsync(webhook,
            new StringContent(JsonConvert.SerializeObject(body), Encoding.UTF8,
"application/json"));
            LambdaLogger.Log($"Message sent with result {result}");
            return result;
        }
        catch (Exception e)
        {
            LambdaLogger.Log($"Message failed to send");
            return new HttpResponseMessage(HttpStatusCode.SeeOther);
        }
    }
}
```

Figure 7. Function used to send message to a Discord channel.

After Discord receives the request a message is added and marked as a BOT message indicating it was sent by a program rather than human. An example message can be seen in Figure 8.



MP Bot BOT 14.03.2020
"https://www.youtube.com/watch?v=jemZE1JtTIs"

Figure 8. Screenshot showing an example message containing link to a YouTube video.

User Interface

The program is using Model View Controller paradigm for user interface. The ASP .NET is routing requests using /controller/action format to the correct action on a controller. The type of action run also depends on the request type set as attributes above functions such as [HttpPost] and [HttpGet]. After a matching action is found it runs prerequisite code and stores any data needed by the view in a ViewBag which is then passed to the view automatically. The view uses passed data to alter its contents using Razor and is converted to HTML code. The styling of view was handled by Bootstrap which is a library used to achieve a consistent look across multiple pages.

As set out in the aims, Blazor has been used to create dynamic progress bar which polls the database for the status of current job. It then uses received response to determine how much of the progress needs to be displayed and updates the component. The Blazor code is run by the server and data needed to be displayed is transferred using protocol called SignalR which allows server to push new data to clients. Without using Blazor similar solution would have to be either insecure direct call to database from client-side JavaScript which would make the credentials easy to fide. Alternative would be calling a REST API to make the server do the database call hiding the credentials from the outside world and the return response from the database. Both solutions have no obvious benefits and they require extra development time to make them work which illustrates why Blazor is an interesting technology.

During research of other similar products, a user interface issue was found where attempting to connect with YouTube has been incredibly unfriendly for users. The attempted way to handle this scenario managed to simplify this process to simply few mouse clicks. The process is starts when user navigates to settings page after logging in. Then when user clicks on YouTube button and then on “Get Youtube Token” button, the user gets redirected to Google OAuth consent screen. This screen is a familiar screen controlled entirely by Google which asks the user to verify that they want to grant shown permissions to this program. After the user selects the account and grant permission a redirection back to the server happens where the token data is stored in the database and becomes tied to the user account. This approach to obtaining token is better as the user is likely to have witnessed the same screen before when using other websites that also support ability to sign in using a Google account.

Accounts

The user accounts are mainly handled by the Identity library. Firstly, when the program is run the database connection for authentication is created. This database stores all of the user data required for authentication and authorization. Two screens were added to allow users to login and register. The register screen requires email, password and confirmation of the passwords. If the inputted password matches the requirements set on the model which is used to display the form a post request is made to Accounts controller. The controller adds the user to the database which now enables the user to log in. Similarly, after entering credentials in the login screen a post request is sent to validate them. If they are correct Identity library sets session cookie which enables user to access controllers and actions with [Authorize] attribute. If a user is not logged in or the session cookie expired the user is redirected to login page.

Additionally, to ensure that the credentials of users are not vulnerable to man in the middle attacks and packet sniffing the solution forces https by redirecting to https version of the website if it is accessed over http. The https protocol encrypts the packets which means that simple packet sniffing will not be able to obtain meaningful data from them. If a man in the middle attacks happens majority of the browsers will notice invalid signature on the SSL certificate and warn the user that the security might have been compromised.

Furthermore, all the password that are stored in the database are hashes ensure that if a hacker would gain access to the database there would be no easy way to access any account. This is since algorithm employed is HMACSHA256 which has no known exploits as of the time of writing this report. The only way to attempt to gain the password through a brute force attack which is unfeasible in most scenarios due to length required to run the attack which can go up to multiple years for a single password depending on resources used and the password length.

Uploading

The upload process will be described in stages as it happens in the expected scenario.

User upload

The process begins when the user navigates to upload page which is available on the navigation bar after logging in. Then user has an option to select file form local storage to be uploaded and select to which services the file ought to be uploaded and to which platforms the links ought to be sent. After the files is uploaded to the server, it is immediately uploaded to the S3 bucket along with tags containing the id of the job and where to which platform the file needs to be uploaded.

The S3 bucket wait until the upload process finishes and then based on the configuration it starts a lambda function and passes a S3Event to it which contains information explaining which file was uploaded and from which bucket.

The main lambda function uses the event given to retrieve the tags with the vital information. The user id is used to identify the user and retrieve relevant OAuth tokens required for interaction with chosen social media platforms. Additionally, the job status is changed in the database using job id from the tags to let user know that the file is being processed. Afterwards a two-step process occurs.

Firstly, all lambdas that correspond to video platforms are started in parallel. This is where the time saving gain happens as each lambda function has its own connection and is able to use maximum bandwidth allocated for the sole purpose of the upload. The main lambda waits for the finish of all the invoked lambda where each of them return an URL to view the video.

Secondly, the URLs returned in the first step are sent to text based social media to inform all of the followers about new video release. This step also happens in parallel to save even more time.

While the lambdas all do their work the job status is being updated constantly. The updates is what allow the progress bar to inform the user about what is currently being done on the file.

Debugging

During the development of this project there were multiple occasions where bugs have been introduced. If it was on the ASP .Net Core server normal debugging took place using tools provided in Visual Studio. However, when the issue was present in Lambda functions same approach was unavailable because of .Net still being in early adoption in AWS. In these scenarios logs were used in Amazon CloudWatch to find at what point the function breaks and what is causing the issue.

Testing

Website

The website has been tested in a multitude of ways. Firstly, unit tests were utilized to ensure correctness of algorithms developed (see Appendix C for all unit tests). These tests also enabled easy way to identify if a function works with expected parameters. This leads to a more robust code as range of expected values is clearly defined and validated. Furthermore, it gives confidence to developers to refactor code with less fear of introducing undefined behaviour which without testing might cause undetected bugs.

Secondly, user interface testing has been added to verify that user is able to navigate correctly between pages and to make sure there are no missing elements on the page (see Appendix D for screenshot from test runner showing the UI tests). This type of testing also works as an integration test as it simulates user clicking buttons rather than checking specific class. The test library used to simulate user interacting with the system is called Selenium web driver. Majority of the functionality was tested only excluding action which would permanently alter the state. An example of such action is registering a new user. Multiple tests were made to verify that all the fields are visible and that errors are displayed when there are values missing or the values do not meet the requirements. However, the act of successfully creating a user is not tested because a new user would permanently get added to the database every time, which would also mean that that user needs to get deleted after the test is run. This would lead to unreliable result as there is a chance that there would be a collision with actual user and potential data loss could happen.

Lambda Functions

All of the lambda functions were tested using exploratory testing after each deployment and major code change. Logs available on Amazon CloudWatch were used to verify that the functionality works as expected.

Additionally, unit tests were developed where possible to cover as much as of the function as possible. This oftentimes required introduction of interfaces to allow for dependency injection. Furthermore, some function had to be modified to decrease direct coupling between classes which alongside with increased use of interfaces allowed for use of Mocks for objects which interacted with 3rd party services. Additionally, fake database class has been created which was injected into the function as using the Moq library to mock the database proved unfruitful.

Critical analysis

Ethical Issues

There are not many ethical issues that are apparent, but there are questions that should be answered to have a clearer picture of the consequences of this project. Currently there are no checks to verify that a user is not using this website in order to spam other platforms. This should be remedied as it could lead to the abuse of terms and condition of most platforms. This could cause this project to be blocked from using the API completely stopping the usefulness of this solution.

Furthermore, there has been no terms and conditions created along with privacy notice to name a few. These documents need to be created before the website goes public to conform to the law. Additionally, no simple way that would all deletion of all of user data has been made. This makes conforming to the GDPR law a manual tedious process which should also be addressed.

Development issues

Along the way of creating the solution git has been used in an agile fashion. Although, the GitHub flow has been used there was no obvious benefit to it over just using one branch in this project. The main reason identified was lack of other collaborators and only one final deadline. There was no reason to merge into master branch if the program was not going to be used in production. Similarly, the sprints were used but the benefit is hard to judge. With the deadlines for other courses it gave more flexibility to the development process, but this could have been achieved in other methodologies if the deadlines were considered to begin with.

During development of the application there were multiple roadblocks which impeded progress. To begin with in the early stages of this project there was an idea to use GraphQL technology. GraphQL is a way a query language which allows for optimized data transfers because it only returns the data requested and no other fields that might have been parts of the requested object. The aim was to implement it to highlight another new and interesting technology. While trying to implement it, it became clear that there was lack of understanding in how the technology works which led to nearly two weeks of work being removed. The issue was caused by misunderstanding in how the GraphQL would fit in the whole architecture which at the time of attempting to implement it meant that it was no longer feasible to completely restructure all of the connections between the different elements.

Similar issues happened when there was an attempt to obtain access to API of other social media platforms. For example, Twitter requires an approval for the developer account which takes up to two weeks and can still be declined, Facebook also has a similar process and additionally when browsing through their API it was not clear which in particular is most fitting to this project and to use Reddit API there is only a need for a normal account but in order to post somewhere a new subreddit would have to be created which is locked for new accounts until a non-disclosed level of reputation is reached. These restrictions were not foreseen and decreased the ability to showcase and test the scalability and achieve set out aims even though the code has been written to allow for it.

There have also been issues due to never working with cloud before. The main issue has been not being used to turning off resources after developing sessions. This led multiple times to going over the budget and having to incur costs that could have been avoided by being more organised. Another issue with cloud is that there is a delay between turning on a database and being able to use it. While working on this project there have been time were due to lack of availability the waiting time for database to turn on would even go to 20 minutes.

One of the aims was 90% of unit test coverage. This aim was not met because there is high percentage of code which directly interacts with 3rd party services making it impossible to test using unit tests. The final score for coverage is 28% after excluding the test projects (see Appendix E for coverage for all project in the solution). This number is much lower than the goal. However, where there was no way to test using unit tests other form of testing was used like exploratory and UI testing which would increase the coverage percentage if it was possible to include it in the coverage report.

Another goal of this project was to create a well written maintainable solution. This has been achieved with all of the solution with the lowest Maintainability Index of 72 and highest of 95 (see Appendix F for all metrics). This shows that if this solution would be worked on for longer there is a low technical debt which means there is no need to rewrite code to improve quality.

Future Improvements

There are multiple ways in which this project could get improved. One of the major ways is improving the overall user interface design. The user experience hasn't been the main focus of this project and therefore requires additional work. One way to achieve this would be to conduct user testing and ask for feedback after initial redesign. User feedback is valuable as it can shed light on seemingly obvious things that were not notice due to constant exposure to the system by people working on it. Furthermore, localization and accessibility should be considered as currently neither are implemented. Adding localization will allow users from other countries use the system massively boosting potential user base of the solution. Similarly, adding accessibility features like alt text and optimising the layout to make sure that screen readers read the page correctly goes a long way towards allowing disabled people to operate the system.

From the architectural point of view configuring the EC2 instance to scale automatically depending on demand should be set. This will allow to not only optimize running costs, but it will also allow for optimal performance due to automatic scaling of servers. Another boon of such solution is higher uptime as all of the servers should never be down.

Additionally, load balancer should be added to spread the load evenly across all of the instances. This will make sure that users will be redirected to new/underutilized EC2 instances further optimizing the performance and cost of the solution.

Moreover, an investigation should take place to check how big of a bottleneck the database is. It could be possible that a distributed database could be a better solution with larger number of users. Another possible solution would be to utilize a cache like Redis to speed up access to frequently queried items in the database.

Another way to improve this project would be to add integration to larger number of different platforms. Currently there are only two, but to compete with any other solution out there most other major platforms need to be added to look like an attractive alternative for potential users. Such addition should not be a difficult achievement as all of the foundation has been laid out. The things that are needed to be added are platform specific user interface, lambda function for interacting with the platform and some trivial minor modifications in other places.

Should this project be released to the public the costs required to run it should be investigated to determine if it is commercially viable. Different monetisation methods need to be examined to determine which is preferred by the target demographic and will allow long term growth.

Eventually it would also be potentially beneficial to add support for other cloud providers which should be relatively easy due to care for low coupling during development. This will allow for quick migration in case of drastic cost changes or changes how some services behave.

Personal Improvements

During the development of the project substantial personal improvements have been made. Since this project involved work with new technologies there were multiple scenarios were met and the road the meet the aims had to be revaluated. In those cases, it was important to remain steadfast and realistically decide which solution makes most sense given the timeframes. One of the takeaways from the gained experience is to think more about other external factors that might affect the time to work on the project. During the planning other assignments were not taken into consideration which skewed the timeframes and increased the perceived pressure of the project. Furthermore, things like lack of access to the library in order to get sources was not considered which was revealed to be an oversight in the light of a coronavirus outbreak. Luckily this did not heavily impact the result thanks to large amount of resources available through the Sheffield Hallam Library website.

References

Alshamrani, A., & Bahattab, A. (2015). A Comparison Between Three SDLC Models Waterfall Model, Spiral Model, and Incremental/Iterative Model. International Journal of Computer Science Issues (IJCSI), 12(1), 106–111.

Clement, J. (2020). Youtube average video length by category 2018 | Statista. Retrieved 21 April 2020, from https://www.statista.com/statistics/1026923/youtube-video-category-average-length/

Compare Repositories - Open Hub. (2020). Retrieved 21 April 2020, from https://www.openhub.net/repositories/compare

Comparing the cloud giants: Uptime and reliability. (2020). Retrieved 21 April 2020, from https://www.linkeit.com/blog/comparing-the-gigants-of-cloud-uptime-and-reliability

Frank, M., & Carmi, B. (2014). Implications of Pressure for Shortening the Time to Market (TTM) in Defense Projects. International Journal Of Information Technologies And Systems Approach, 7(1), 23-40. doi: 10.4018/ijitsa.2014010102

How AWS Stacks Up Against Rivals on Downtime. (2020). Retrieved 21 April 2020, from https://www.theinformation.com/articles/how-aws-stacks-up-against-rivals-on-downtime

Koetsier, J. (2020). Private Social: Discord's 250M Users Are Already Where Facebook Wants To Go. Retrieved 21 April 2020, from https://www.forbes.com/sites/johnkoetsier/2019/06/12/private-social-discords-250m-users-are-already-where-facebook-wants-to-go/#3371489441fd

Object storage. (2020). Retrieved 21 April 2020, from https://en.wikipedia.org/wiki/Object_storage

Richardson, J., & Gwaltney, W. (2005). Ship it ! (pp. 104-128). Raleigh (N. C.): Pragmatic Bookshelf.

Schwaber, K., & Sutherland, J. (2012). Software in 30 Days: How Agile Managers Beat the Odds, Delight Their Customers, And Leave Competitors In the Dust (1st edition). Wiley.

Stack Overflow Developer Survey 2019. (2020). Retrieved 21 April 2020, from https://insights.stackoverflow.com/survey/2019

Stalcup, K. (2020). AWS vs Azure vs Google Cloud Market Share 2020: What the Latest Data Shows - ParkMyCloud. Retrieved 21 April 2020, from https://www.parkmycloud.com/blog/aws-vs-azure-vs-google-cloud-market-share/

TechEmpower Framework Benchmarks. Retrieved 21 April 2020, from https://www.techempower.com/benchmarks/#section=data-r17&hw=cl&test=fortune

The golden age of YouTube is over. (2020). Retrieved 21 April 2020, from https://www.theverge.com/2019/4/5/18287318/youtube-logan-paul-pewdiepie-demonetization-adpocalypse-premium-influencers-creators

The Pros and Cons of Waterfall Methodology. (2020). Retrieved 21 April 2020, from https://www.lucidchart.com/blog/pros-and-cons-of-waterfall-methodology

Thomas, D., & Hunt, A. (2020). The Pragmatic Programmer: journey to mastery, 20th Anniversary Edition, 2/e (2nd ed., pp. 53-57).

UK home broadband performance, measurement period November 2018. (2020). Retrieved 21 April 2020, from https://www.ofcom.org.uk/research-and-data/telecoms-research/broadband-research/home-broadband-performance-2018

Version Control Systems Popularity in 2016. (2020). Retrieved 21 April 2020, from https://rhodecode.com/insights/version-control-systems-2016

VIDEO ROOM 1000 COMPLETE MIX -- All 1000 videos seen in sequential order!. (2020). Retrieved 21 April 2020, from https://www.youtube.com/watch?v=icruGcSsPp0

Bibliography

AWS Cloud Practitioner Essentials. Retrieved 21 April 2020, from https://aws.amazon.com/training/course-descriptions/cloud-practitioner-essentials/

Chadwick, J. Learning ASP.NET Core MVC. Retrieved 19 December 2019, from https://www.linkedin.com/learning/learning-asp-dot-net-core-mvc/use-identity-services-to-allow-users-to-log-in-to-your-site?u=69719634

Freeman, E., & Freeman, E. (2005). Head first design patterns. [Sebastopol, CA]: O'Reilly Media.

McConnell, S. (2016). Code complete. Redmond, Wash.: Microsoft Press.

Skeet, J. (2019). C♯ in depth (4th ed.). Shelter Island, NY: Manning.

Appendices

Appendix A – Project Specification

PROJECT SPECIFICATION - Project (Technical Computing) 2019/20

Student:	**Bartosz Jurus**
Date:	**29/09/19**
Supervisor:	**Martin Cooper**
Degree Course:	**BSc Computer Science**
Title of Project:	**Automating social media management for content creators**

Elaboration

Social media content creators need to utilize multiple platforms simultaneously, which is time consuming. This gave rise to freelancers whose sole job is going to different social media sites and posting announcements and videos. Doing so creates risk for the content creators because they have to give their account details to people they don't know in order to outsource the mundane and tedious work.

This project aims to solve this problem along with allowing them to increase their outreach by simplifying the process of uploading the videos to multiple websites. The project will provide one platform that will allow users to upload a video once which will then be propagated to all supported video hosting websites. This way content creators like youtubers will have an alternative income sources from other platforms and be able to post announcements to their followers at the same time.

Additionally, this project will research and use emerging technologies like GraphQL and Microsoft blazor to investigate

Since the system can potentially be used by large number of users simultaneously transferring large video files it is paramount that processing of request is able to scale on demand. For this reason cloud computing provider like AWS will be used to meet this requirement.

Project Aims

This should be a bullet-point list of aims for the project which describes what you are hoping to achieve. Your project report should refer back to this list to provide information on the extent to which these aims have been achieved.

- Produce a website for interacting with system
- Research and implement Microsoft blazor for front end user interface
- Research and implement GraphQL for Server API
- Research and implement Cloud computing for scalable server architecture
- Research and implement solution using .Net Core
- Integrate with Twitter API
- Integrate with YouTube API
- Integrate with Facebook API
- Unit testing for server components
- Code coverage of unit tests over 90% of lines of code
- Maintainability Index for all server modules at over 65/100

Project deliverable(s)

- The project will be using Kanban Board like Trello or GitKraken Glo board
- The project will be using Tracer bullet development for adding new features
- The project will aim to unit test as much as possible
- The project will deliver a cross platform server that will host a website that users will interact with

Action plan

This should be a **table** listing the jobs that need doing in order to succeed with your project. Against each job you should put a date by when it needs to be done. You and your supervisor will use this to ensure that the project remains on schedule. You could also use a graphical technique to present the information. Include optional deadlines such as the information review, contents page, and draft critical evaluation submissions.

Action	Due date
The Project Specification and ethics form	25/10/2019
Research cloud solutions	1/11/2019
Research GraphQL	8/11/2019
Research Blazor	15/11/2019
Plan server architecture	22/11/2019
Research ASP.NET Core	29/11/2019
Research Identity middleware	6/12/2019
The Information Review	6/12/2019
Research Entity Framework	13/12/2019
Produce UML diagrams explaining the system	20/12/2019
Design User Interface	27/12/2019
Implement backend	17/01/2020
Implement frontend	31/01/2020
The provisional Contents Page	21/02/2020
The draft Critical Evaluation	27/03/2020
Turnitin submission	22/04/2020
Physical submission	23/04/2020
Project demonstration	Before 12/05/2020

BCS Code of Conduct

I confirm that I have successfully completed the BCS code of conduct on-line test with a mark of 70% or above. This is a condition of completing the Project (Technical Computing) module.

Signature: B. Jurus

Publication of Work

I confirm that I understand the "Guidance on Publication Procedures" as described on the Bb site for the module.
Signature:

GDPR

I confirm that I will use the "Participant Information Sheet" as a basis for any survey, questionnaire or participant testing materials. This form is available on the Bb site for the module.
Signature:

Ethics

Complete the SHUREC 7 (research ethics checklist for students) form below. If you think that your project may include ethical issues that need resolving (working with vulnerable people, testing procedures etc.) then discuss this with your supervisor as soon as possible and comment further here.
Both you and your supervisor need to sign the completed SHUREC 7 form.
Please contact the project co-ordinator if further advice is needed.

RESEARCH ETHICS CHECKLIST FOR STUDENTS (SHUREC 7)

This form is designed to help students and their supervisors to complete an ethical scrutiny of proposed research. The SHU Research Ethics Policy should be consulted before completing the form.

Answering the questions below will help you decide whether your proposed research requires ethical review by a Designated Research Ethics Working Group.

The final responsibility for ensuring that ethical research practices are followed rests with the supervisor for student research.

Note that students and staff are responsible for making suitable arrangements for keeping data secure and, if relevant, for keeping the identity of participants anonymous. They are also responsible for following SHU guidelines about data encryption and research data management.

The form also enables the University and Faculty to keep a record confirming that research conducted has been subjected to ethical scrutiny.

For student projects, the form may be completed by the student and the supervisor and/or module leader (as applicable). In all cases, it should be counter-signed by the supervisor and/or module leader, and kept as a record showing that ethical scrutiny has occurred. Students should retain a copy for inclusion in their research projects, and staff should keep a copy in the student file.

Please note if it may be necessary to conduct a health and safety risk assessment for the proposed research. Further information can be obtained from the Faculty Safety Co-ordinator.

General Details

Name of student	Bartosz Jurus
SHU email address	B6009240@my.shu.ac.uk
Course or qualification (student)	BSc Computer Science
Name of supervisor	Martin Cooper
email address	M.J.Cooper@shu.ac.uk
Title of proposed research	Automating social media management for content creators

Proposed start date	25/10/2019
Proposed end date	24/04/2020
Brief outline of research to include, rationale & aims (250-500 words).	Social media content creators need to utilize multiple platforms simultaneously, which is time consuming. This gave rise to freelancers whose sole job is going to different social media sites and posting announcements and videos. Doing so creates risk for the content creators because they have to give their account details to people they don't know in order to outsource the mundane and tedious work. This project aims to solve this problem along with allowing them to increase their outreach by simplifying the process of uploading the videos to multiple websites. The project will provide one platform that will allow users to upload a video once which will then be propagated to all supported video hosting websites. This way content creators like youtubers will have an alternative income sources from other platforms and be able to post announcements to their followers at the same time. Additionally, this project will research and use emerging technologies like GraphQL and Microsoft blazor to investigate Since the system can potentially be used by large number of users simultaneously transferring large video files it is paramount that processing of request is able to scale on demand. For this reason cloud computing provider like AWS will be used to meet this requirement. This project aims to: • Produce a website for interacting with system • Research and implement Microsoft blazor for front end user interface • Research and implement GraphQL for Server API • Research and implement Cloud computing for scalable server architecture • Research and implement solution using .Net Core • Integrate with Twitter API • Integrate with YouTube API • Integrate with Facebook API • Unit testing for server components • Code coverage of unit tests over 90% of lines of code • Maintainability Index for all server modules at over 65/100
Where data is collected from individuals, outline the nature of data, details of anonymisation, storage and disposal procedures if required (250-500 words).	No user data will be collected. For testing and evaluating the solution testing account will be used by me.

1. Health Related Research Involving the NHS or Social Care / Community Care or the Criminal Justice Service or with research participants unable to provide informed consent

Question	Yes/No
1. Does the research involve? • Patients recruited because of their past or present use of the NHS or Social Care • Relatives/carers of patients recruited because of their past or present use of the NHS or Social Care • Access to data, organs or other bodily material of past or present NHS patients • Foetal material and IVF involving NHS patients • The recently dead in NHS premises • Prisoners or others within the criminal justice system recruited for health-related research* • Police, court officials, prisoners or others within the criminal justice system* • Participants who are unable to provide informed consent due to their	No
2. Is this a research project as opposed to service evaluation or audit? *For NHS definitions please see the following website* http://www.hra.nhs.uk/documents/2013/09/defining-research.pdf	No

If you have answered **YES** to questions **1 & 2** then you **must** seek the appropriate external approvals from the NHS, Social Care or the National Offender Management Service (NOMS) under their independent Research Governance schemes. Further information is provided below.

NHS https://www.myresearchproject.org.uk/Signin.aspx

***** All prison projects also need National Offender Management Service (NOMS) Approval and Governor's Approval and may need Ministry of Justice approval. Further guidance at:

http://www.hra.nhs.uk/research-community/applying-for-approvals/national-offender-management-service-noms/

NB FRECs provide Independent Scientific Review for NHS or SC research and initial scrutiny for ethics applications as required for university sponsorship of the research. Applicants can use the NHS pro-forma and submit this initially to their FREC.

2. Research with Human Participants

Question	Yes/No
Does the research involve human participants? This includes surveys, questionnaires, observing behaviour etc.	No

Question	**Yes/No**
1. *Note If YES, then please answer questions 2 to 10* *If NO, please go to Section 3*	
2. Will any of the participants be vulnerable? *Note: Vulnerable' people include children and young people, people with learning disabilities, people who may be limited by age or sickness, etc. See definition on website*	
3. Are drugs, placebos or other substances (e.g. food substances, vitamins) to be administered to the study participants or will the study involve invasive, intrusive or potentially harmful procedures of any kind?	
4. Will tissue samples (including blood) be obtained from participants?	
5. Is pain or more than mild discomfort likely to result from the study?	
6. Will the study involve prolonged or repetitive testing?	
7. Is there any reasonable and foreseeable risk of physical or emotional harm to any of the participants? *Note: Harm may be caused by distressing or intrusive interview questions, uncomfortable procedures involving the participant, invasion of privacy, topics relating to highly personal information, topics relating to illegal activity, etc.*	
8. Will anyone be taking part without giving their informed consent?	
9. Is it covert research? *Note: 'Covert research' refers to research that is conducted without the knowledge of participants.*	
10. Will the research output allow identification of any individual who has not given their express consent to be identified?	

If you answered **YES only** to question **1,** the checklist should be saved and any course procedures for submission followed. If you have answered **YES** to any of the other questions you are **required** to submit a SHUREC8A (or 8B) to the FREC. If you answered **YES** to question **8** and participants cannot provide informed consent due to their incapacity you must obtain the appropriate approvals from the NHS research governance system. Your supervisor will advise.

3. Research in Organisations

Question	**Yes/No**
1. Will the research involve working with/within an organisation (e.g. school, business, charity, museum, government department, international agency, etc.)?	No
2. If you answered YES to question 1, do you have granted access to conduct the research? *If YES, students please show evidence to your supervisor. PI should retain safely.*	

3. If you answered NO to question 2, is it because: A. you have not yet asked B. you have asked and not yet received an answer C. you have asked and been refused access. *Note: You will only be able to start the research when you have been granted access.*	

4. Research with Products and Artefacts

Question	**Yes/No**
1. Will the research involve working with copyrighted documents, films, broadcasts, photographs, artworks, designs, products, programmes, databases, networks, processes, existing datasets or secure data?	Yes
2. If you answered YES to question 1, are the materials you intend to use in the public domain? *Notes: 'In the public domain' does not mean the same thing as 'publicly accessible'.* • *Information which is 'in the public domain' is no longer protected by copyright (i.e. copyright has either expired or been waived) and can be used without permission.* • *Information which is 'publicly accessible' (e.g. TV broadcasts, websites, artworks, newspapers) is available for anyone to consult/view. It is still protected by copyright even if there is no copyright notice. In UK law, copyright protection is automatic and does not require a copyright statement, although it is always good practice to provide one. It is necessary to check the terms and conditions of use to find out exactly how the material may be reused etc.* *If you answered YES to question 1, be aware that you may need to consider other ethics codes. For example, when conducting Internet research, consult the code of the Association of Internet Researchers; for educational research, consult the Code of Ethics of the British*	Yes
3. If you answered NO to question 2, do you have explicit permission to use these materials as data? *If YES, please show evidence to your supervisor.*	
4. If you answered NO to question 3, is it because: A. you have not yet asked permission B. you have asked and not yet received and answer C. you have asked and been refused access.	**A/B/C**

Adherence to SHU policy and procedures

Personal statement

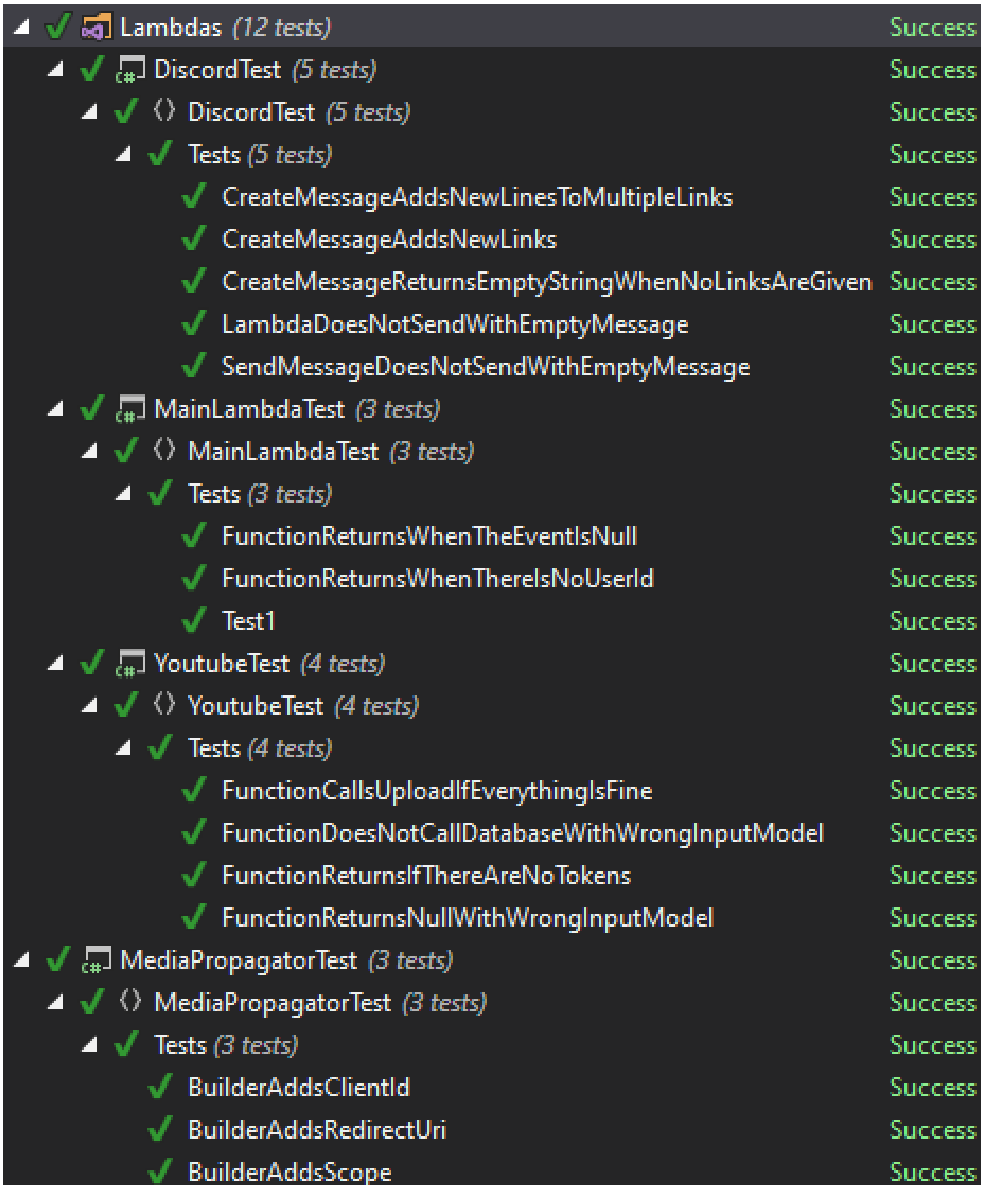
I can confirm that: – I have read the Sheffield Hallam University Research Ethics Policy and Procedures	
Student	
Name: Bartosz Jurus	Date:24/10/2019
Signature: B. Jurus	
Supervisor or other person giving ethical sign-off	
I can confirm that completion of this form has not identified the need for ethical approval by the FREC or an NHS, Social Care or other external REC. The research will not commence until any approvals required under Sections 3 & 4 have been received.	
Name:	Date:
Signature:	

Appendix B – Calculations for the length of upload

	Video size (in MB)			Time to upload (in minutes)		
Video Length (in minutes)	**720p**	**1080p**	**2160p**	**720p**	**1080p**	**2160p**
5	200	800	3360	0.462963	1.851852	7.777778
10	400	1600	6720	0.925926	3.703704	15.55556
15	600	2400	10080	1.388889	5.555556	23.33333
20	800	3200	13440	1.851852	7.407407	31.11111
25	1000	4000	16800	2.314815	9.259259	38.88889

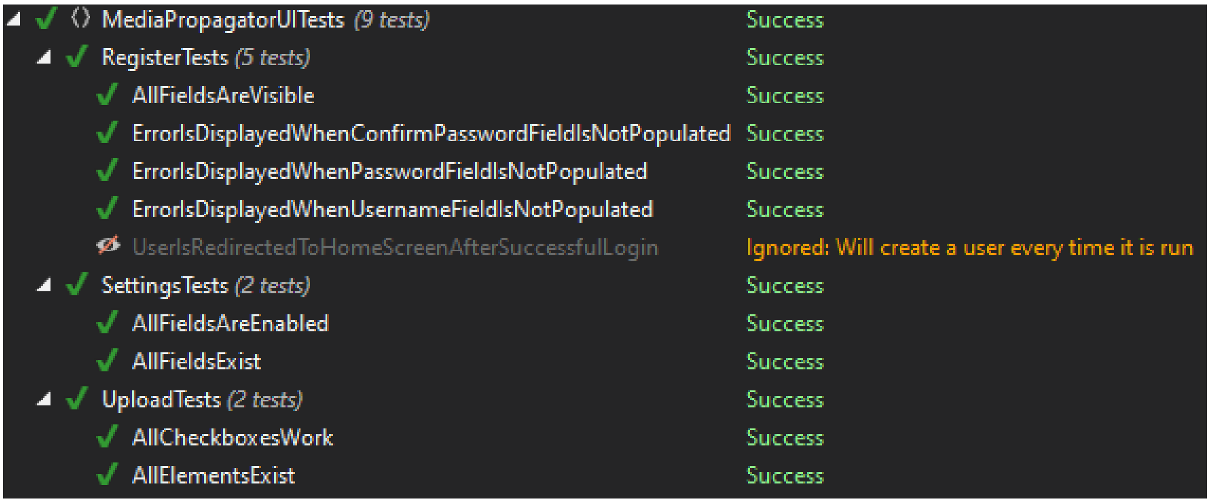
Internet speed
7.2

Appendix C – Unit tests



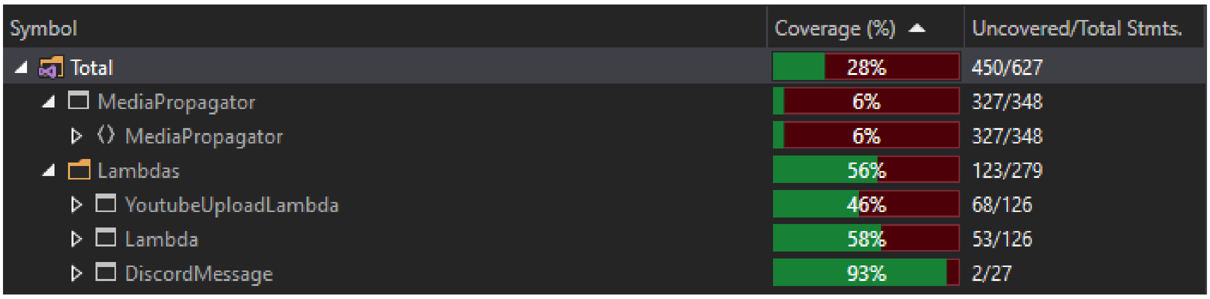
```
◢ ✓ Lambdas (12 tests)                                              Success
  ◢ ✓ DiscordTest (5 tests)                                         Success
    ◢ ✓ {} DiscordTest (5 tests)                                    Success
      ◢ ✓ Tests (5 tests)                                           Success
          ✓ CreateMessageAddsNewLinesToMultipleLinks                Success
          ✓ CreateMessageAddsNewLinks                               Success
          ✓ CreateMessageReturnsEmptyStringWhenNoLinksAreGiven      Success
          ✓ LambdaDoesNotSendWithEmptyMessage                       Success
          ✓ SendMessageDoesNotSendWithEmptyMessage                  Success
  ◢ ✓ MainLambdaTest (3 tests)                                      Success
    ◢ ✓ {} MainLambdaTest (3 tests)                                 Success
      ◢ ✓ Tests (3 tests)                                           Success
          ✓ FunctionReturnsWhenTheEventIsNull                       Success
          ✓ FunctionReturnsWhenThereIsNoUserId                      Success
          ✓ Test1                                                   Success
  ◢ ✓ YoutubeTest (4 tests)                                         Success
    ◢ ✓ {} YoutubeTest (4 tests)                                    Success
      ◢ ✓ Tests (4 tests)                                           Success
          ✓ FunctionCallsUploadIfEverythingIsFine                   Success
          ✓ FunctionDoesNotCallDatabaseWithWrongInputModel          Success
          ✓ FunctionReturnsIfThereAreNoTokens                       Success
          ✓ FunctionReturnsNullWithWrongInputModel                  Success
◢ ✓ MediaPropagatorTest (3 tests)                                   Success
  ◢ ✓ {} MediaPropagatorTest (3 tests)                              Success
    ◢ ✓ Tests (3 tests)                                             Success
        ✓ BuilderAddsClientId                                       Success
        ✓ BuilderAddsRedirectUri                                    Success
        ✓ BuilderAddsScope                                          Success
```

Appendix D – Automated UI Tests



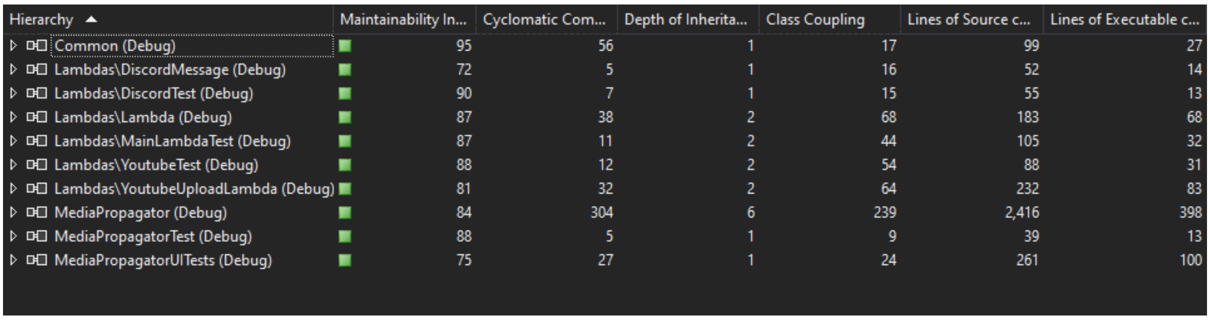
```
◢ ✓ {} MediaPropagatorUITests (9 tests)                     Success
  ◢ ✓ RegisterTests (5 tests)                               Success
      ✓ AllFieldsAreVisible                                 Success
      ✓ ErrorIsDisplayedWhenConfirmPasswordFieldIsNotPopulated Success
      ✓ ErrorIsDisplayedWhenPasswordFieldIsNotPopulated     Success
      ✓ ErrorIsDisplayedWhenUsernameFieldIsNotPopulated     Success
        UserIsRedirectedToHomeScreenAfterSuccessfulLogin    Ignored: Will create a user every time it is run
  ◢ ✓ SettingsTests (2 tests)                               Success
      ✓ AllFieldsAreEnabled                                 Success
      ✓ AllFieldsExist                                      Success
  ◢ ✓ UploadTests (2 tests)                                 Success
      ✓ AllCheckboxesWork                                   Success
      ✓ AllElementsExist                                    Success
```

Appendix E – Unit test coverage



Symbol	Coverage (%)	Uncovered/Total Stmts.
Total	28%	450/627
MediaPropagator	6%	327/348
MediaPropagator	6%	327/348
Lambdas	56%	123/279
YoutubeUploadLambda	46%	68/126
Lambda	58%	53/126
DiscordMessage	93%	2/27

Appendix F – Code Metrics



Hierarchy	Maintainability In...	Cyclomatic Com...	Depth of Inherita...	Class Coupling	Lines of Source c...	Lines of Executable c...
Common (Debug)	🟩 95	56	1	17	99	27
Lambdas\DiscordMessage (Debug)	🟩 72	5	1	16	52	14
Lambdas\DiscordTest (Debug)	🟩 90	7	1	15	55	13
Lambdas\Lambda (Debug)	🟩 87	38	2	68	183	68
Lambdas\MainLambdaTest (Debug)	🟩 87	11	2	44	105	32
Lambdas\YoutubeTest (Debug)	🟩 88	12	2	54	88	31
Lambdas\YoutubeUploadLambda (Debug)	🟩 81	32	2	64	232	83
MediaPropagator (Debug)	🟩 84	304	6	239	2,416	398
MediaPropagatorTest (Debug)	🟩 88	5	1	9	39	13
MediaPropagatorUITests (Debug)	🟩 75	27	1	24	261	100