

# EduGrove

## Group KI-35

1. Peter Adema 14460165
2. Gilles de Jong 14639084
3. Ruben Veltman 14649438
4. Jesse Oomen 14618494

Teaching Advisor: Rhijn Wiegman

FNWI/Faculty of Science

[3-2-2023]

# Table of content

1 Introduction .....	3
2 Design.....	4
2.1 Use case(s) .....	4
2.2 General requirements .....	5
2.3 Website/web application design .....	6
3 Implementation: .....	8
3.1 HTML build-up.....	8
3.2 PHP functions and function files .....	8
3.3 JavaScript, jQuery, and AJAX.....	9
3.4 Styling in CSS .....	10
3.5 Web scraping .....	10
5 Main achievements and advanced features .....	15
6 Discussion.....	16

# 1 Introduction

The goal for this project was to create a website where people can view educational videos on any scientific topic of their choice. These subjects include topics such as physics, biology, and geography. The platform's users would be the ones providing the content for other users so we decided to make a video sharing website where users can upload their own videos and make money doing so.

The website should thus allow its users to make and publish their educational content as well as view other people's content through several browsing features. Users are also able to construct courses out of their videos so their audience can easily follow along with an entire course's worth of material.

Users needed to be able to purchase videos and courses as well as view their purchases, which is where the monetary system came in. Users also needed to be able to interact with other users and the creators of the videos which is where the idea of a comment section underneath each video came into play. To further develop user interactions a rating system for the videos, comments, and courses was also put into place.

The website should also be easy to use and user friendly with a clear and obvious lay-out which any user should be able to navigate. Accessibility and adaptive responsive design were also must haves for Edugrove so that became a decently sized portion of the project.

These are the main ideas that were going around for the project at the start of the project period. The next step would be to divide these challenges up into bite-sized pieces and divide up the work so the project could be built up systematically and efficiently.

## 2 Design

On Edugrove there is a clear difference between users registered and unregistered users. Without an account or if you are not logged in you can explore the website, users are able to watch the first video of any course as well as any of the free videos and courses. Besides that, users can view ratings, comments, view-counts, and prices of the videos and courses as well.

For registered users there is a lot more to do on Edugrove. An account is connected to a bank account which holds the user's balance. With this balance users can buy premium videos and courses that would otherwise be inaccessible. When a user has access to a video or course, they can rate them and comment on them. This way users can see what other people think about the content on Edugrove. If you have an account, it is also possible to share content and you can upload courses and videos yourself.

Admins are special users on the site who have more permissions and abilities than normal users on the site. First off, they can gift items to users, this can be used when an admin wants to show content to friends or people they deem important. The admin can delete items as well, such as videos, courses, and comments. The admins need to have this ability because some users may upload content that is not user friendly, in case this happens it is possible for that item to be deleted.

If a user is behaving inappropriately the administrator can ban that user. This can happen when users behave poorly, an example of this would be writing a comment that does not confirm to the community guidelines. The admin has also the right to unban a user. If a user gets in contact about why they were banned, there is a possibility that after certain amount of time a user can be unbanned. The last special ability administrators have is to sudo as a user. This is important because that way the administrator can see what website experience is like as a normal user. In case there is a complaint about the functionality or the styling of a page the admin can take a look without making a user account.

### 2.1 Use case(s)

We developed our project for users that are interested in learning more about any subject of their choosing for an affordable price as well as for creative users that enjoy making educational content whilst being able to make a profit doing so.

That first user type, the average viewer, can also be split into two categories. On the one hand you have the dedicated learner who is willing to spend money on the content on the site, and on the other hand you have the user who wants to watch fun educational videos without spending any money. These users will be called the 'pay-per-view' user, and the free-to-use user, to make it easier to differentiate the two later.

It is because of these two different users that creators on our site get to choose to make their videos paid or free, with their free content they can draw in unaccustomed users that might be

interested in what they have to offer, whilst being able to sell their content to already established viewers.

These new viewers can then fall into either one of the two described categories and either decide to buy a video or course or watch all the free videos the creator has to offer. The site takes these two different sides into consideration by providing free and paid content instead of just paid content.

## **2.2 General requirements**

Creators and pay-per-view users need an account to either upload content, or purchase said content. So, like most websites Edugrove allows its users to make accounts, login, and logout, and manage their account through the form of changing their passwords or linked email accounts.

Like mentioned previously the content creators on Edugrove need to be able to upload their content which the site allows them to do. They can upload their videos and give them titles, descriptions, a main subject, as well as a thumbnail. Courses can also be created by selecting, dragging, and dropping previously uploaded videos in the preferred order on the course creation page.

The pay-per-view user needs to have a bank account with a balance with which they can purchase videos and courses. Edugrove is not able to connect with any real banks since the site does not work with real money. Instead, the site has a built-in fake bank called 'NietNG' which is obviously a play on ING. Users can see and manage their balance as well as view their past transactions and confirm currently pending transactions.

With this balance these users are now finally able to purchase videos and courses, if they have enough money left on their balance, that is. Since the site works with a fake bank it is not possible for users to top off their balances, so each new user has €100 to spend as they wish.

Both the free-to-use and pay-per-view users need to be able to browse the content on Edugrove which is where the multiple browse pages come into play. Users can sort the content by videos, courses, subjects, ratings, and price just to name a few options. Edugrove also includes a search function where users can look up specific topics that they are interested in and b Like mentioned previously the content creators on Edugrove need to be able to upload their content which the site allows them to do. They can upload their videos and give them titles, descriptions, a main subject, as well as a thumbnail. Courses can also be created by dragging and dropping previously uploaded videos in the preferred order on the course creation page. By allowing users to sort by price, free-to-use users are easily able to browse their preferred type of content as well.

Both types of viewers need to have the ability to interact with their favourite content creators and other users, this is why Edugrove features a comment section underneath every one of its videos. To give their opinions users are also able to rate comments, videos, and courses. These ratings also feed back into the diverse types of content sorting discussed earlier.

## **2.3 Website/web application design**

While creating the website, the majority of the design decisions were made rapidly due to the outcomes being fairly clear. The design for the authentication pages, as well as that of the video viewing page was not a point of discussion. However, the design decisions for how a new user should be guided towards a video and their first purchase was more contentious.

One of the ideas was to have the home page contain the actual site content, with links to videos and courses. However, it was decided that this might confuse or deter new users

The other proposed idea was to use the home page as an introduction to the concept of our site, and direct users towards a login form and paywall after they seek to view the actual content of the site. While this was more in line with what we had in mind for the user experience, it was still a concern that users might decide to leave the site without viewing any videos if the barrier to content involved creating an account and paying for videos.

As such, the final iteration of the website included a descriptive home page, along with a teaser video for all courses. This teaser video allows new users to view a course and experience the website without being forced to create an account or pay. The additional benefit to this choice is that more users may become interested in the courses they preview and as such more users may choose to purchase videos or courses from the site.

For the database, there was a clear idea of what tables were crucial and were necessary for the website. However, it later became clear that more tables were necessary as the scope of the project grew. This constant expansion eventually led to the final database model Edugrove uses in the definitive version of the site.

In this final database schema, 18 tables are used in order to store all relevant information. These tables are all in 3NF, utilising foreign keys to ensure data is not duplicated amongst tables. Every table is connected with at least one other, with most having multiple relations. The interconnected nature of the database schema allows for complex JOINS and analysis of the data natively within SQL. Finally, various attributes of the tables are encoded within the schema by declaring rows as unique, or by implementing triggers that ensure all data remains accurate. This has resulted in a database which can easily handle the complex web of information generated by the use of a video sharing platform, while being able to enforce internal consistency.



## 3 Implementation:

Every page on Edugrove has its own styling file, function file, API, and, if necessary, java script file. Most of the elements that are printed on each page are echoed through various PHP functions which are stored in the different PHP files.

### 3.1 HTML build-up

The HTML code itself was kept to a bare minimum, most of the HTML components were constructed with the PHP functions that would be called on the page. This was done to help keep the actual HTML code clean and easy to read. This had to be done since most pages on Edugrove feature automatically generated video boxes and course boxes.

Not every page on the website has this kind of clean HTML layout since this style was something that had to be learned and picked up by everyone first. This makes it quite easy to see which parts of the site were built early in the project and which ones were built later.

### 3.2 PHP functions and function files

As previously stated, every webpage has one or more unique functionality files where multiple functions are stored which do everything from gathering information from the database, to echoing different elements that are shown on the page. These functions are usually used only once since most of them are responsible for extremely specific tasks that are not necessarily required elsewhere on the website. If a function does prove to be required in a lot of separate places, it is moved to a separate file which can then easily be used by multiple different pages. This helps prevent the same function from being written multiple times in various places.

The PHP files within the project can be divided into four broad categories: page files, database functions, component functions and API endpoints.

The first type is that of all the pages on the website, and serve to provide structure. They combine database functions with component functions in order to show a responsive page to the end user, while containing relatively little actual PHP code.

The second type consists of all functions that interact with the database, and allow for a simpler interface that is more readable for within other PHP files. All the database interactions make use of a common database connection, speeding up the process of querying and inserting data. All the database interactions are done via PHP's MySQL PDO, which protects against SQL injection. Additionally, insertions to the database have HTML characters escaped, so that any



pages using user content can be certain that the text they seek to display is not filled with unwanted HTML nodes.

The third type consists of functions that create HTML elements for use within the pages. This type can be further divided into functions that 'render' HTML (generating strings for use in other component functions or API's) and functions that 'display' HTML (outputting to the page that is being generated). The utility of these functions lies in that extremely common web page components can be isolated and turned into reusable code fragments, resulting in less duplicate code.

The final type represents all pages that are meant to be interfaces by the JavaScript functions in the pages instead of by the user or other PHP functions. These pages allow for a page to update or load content without navigating to a separate page, and output JSON files for the JavaScript functions to interpret.

A good example of how these principles is used to deduplicate code is the `'html_header'` function, that generates the general boilerplate header along with all the linked stylesheets, JavaScript files to be loaded by the browser. It also displays the navigation menu, as that is common to all sites. By extracting these elements into a function and making the variable components parameters to it, the entirety of the common header can be added to a page in a single line of code.

### **3.3 JavaScript, jQuery, and AJAX**

Some pages on Edugrove require JavaScript and jQuery to function properly. These languages are used to change things on a page and send things from a page elsewhere in real time, instead of upon reloading the page. Things that change live on the screen at the click of a button are of course written in jQuery since that is one of the best ways to implement live CSS changes. jQuery is naturally also used to send information that is acquired on the page through AJAX post requests to the page's API, where that information can be cleaned and subsequently used in PHP functions to echo new html code or update tables in the database.

The API's that are used on the site receive information from the AJAX posts which is first checked thoroughly. The API checks to see whether the information fulfils the requirements. To elaborate further using an example, say an API expects a string which is supposed to be a comment a user has submitted, that string needs to fulfil a requirement. This requirement is that the comment shall not be an empty string. If it is an empty string, then this information is logged into an array which holds potential errors. These errors are then sent along an API fail response which tells the page and the users just exactly what went wrong.

In case nothing was wrong and all the information that was received turned out to fulfil the requirements, the information is sent along an API success response along with a success

message and any further data such as page elements that need to be displayed with the newly acquired information.

### **3.4 Styling in CSS**

Every page on Edugrove also has its very own styling file which tells the page's components how to look and behave. Most of this styling is unique per page since most pages have unique functions and different layouts. An obvious goal was to give the whole website a similar look and feel except for the fake bank that users can visit on the site, since that is supposed to feel like its own separate thing.

The order in which every part is styled within these CSS pages is made to mimic the layout the different components have on the actual webpage. This makes it easier to find and subsequently edit each components styling.

Every page has an additional section within the CSS file which tells the page what to do in case of someone visiting the page on a device with a smaller screen. This means that every page is equipped with at least some changes in layout in the event of such a case. This does mean that the layouts can change ever so slightly or more drastically in other cases. These changes allow mobile users to better interface with the website, and give the pages a look more native to their devices.

Additionally, while some CSS files have overlapping elements for common features, the numerous forms on the website share a single CSS file to ensure unified design and minimised code duplication.

### **3.5 Web scraping**

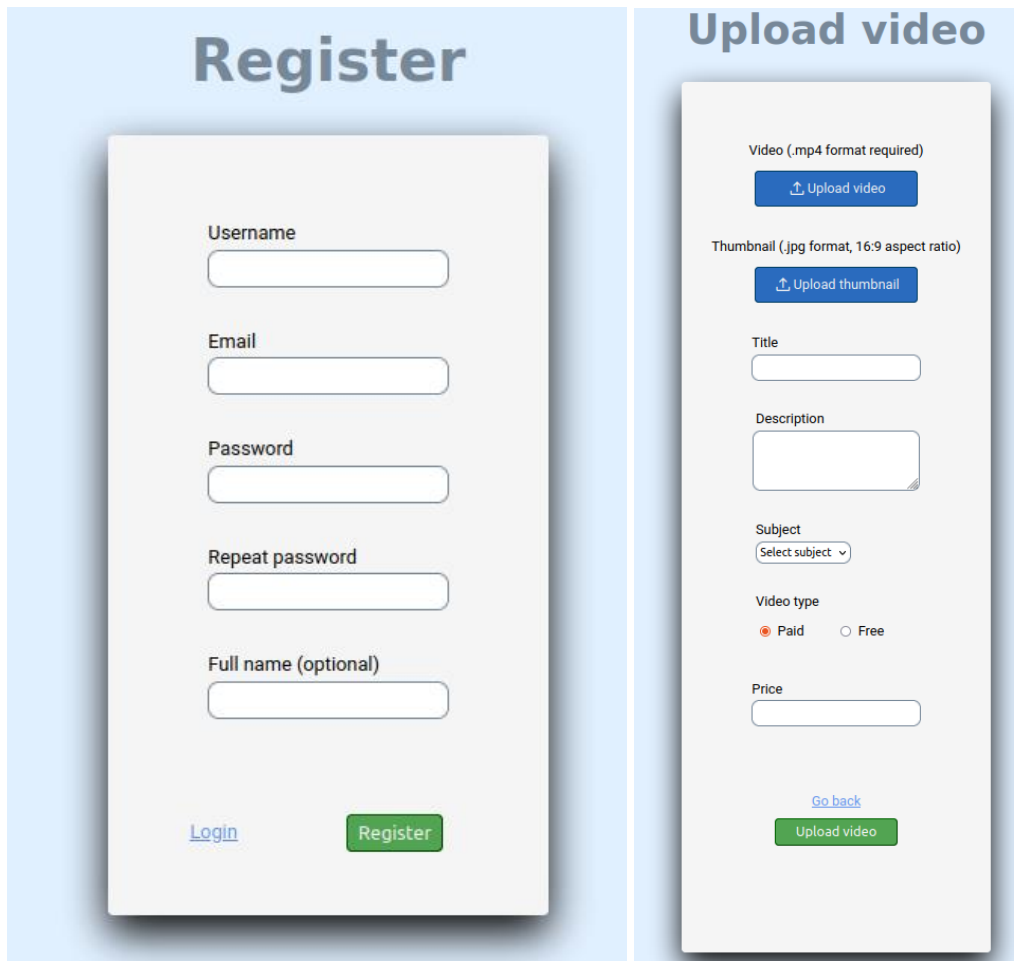
Edugrove needed to be filled with content so that users could have something to purchase and view. A large amount of content like that is not something one can just upload all by themselves, so to fill up the website with content web scraping was used.

With web scraping, playlists from YouTube were downloaded and converted into the eight courses currently present on the site. Numerous channels were victims of the web scraping efforts, and their content was used to fuel the Edugrove. The conversion from YouTube data to data that could be stored within Edugrove's database was done using multiple functions, rewriting the information taken from YouTube into information that the sites database would accept. The tool 'yt-dlp' was used to interface with YouTube's pages, while Python scripts were used to interpret the results and generate SQL queries.

Aside from the videos themselves and the playlists they belong to, other components of the videos were also scraped, such as the comments, likes, views, and descriptions. All these elements were saved within the database and displayed in Edugrove's different video pages. These features that were scraped from YouTube can also be interacted with and updated by the site's users. Ratings can be given, new comments can be placed as well as replies to other comments, and these comments can also be rated.

## 4 Results

At the end of the project's term the website was finalized, and the results were quite impressive. Edugrove looked and behaved exactly like the initial goals that had been laid out at the start of the project. Edugrove is a website with a lot of different pages and in order to give a glimpse of Edugrove, five screenshots from the website are displayed with a small explanation.



The image displays two side-by-side screenshots of web forms. The left screenshot, titled 'Register', features a light blue header and a white form box with rounded corners. It contains five input fields: 'Username', 'Email', 'Password', 'Repeat password', and 'Full name (optional)'. At the bottom left is a blue 'Login' link, and at the bottom right is a green 'Register' button. The right screenshot, titled 'Upload video', also has a light blue header and a white form box. It includes a blue 'Upload video' button at the top, followed by a 'Thumbnail (.jpg format, 16:9 aspect ratio)' section with its own 'Upload thumbnail' button. Below these are input fields for 'Title' and 'Description', a 'Subject' dropdown menu, a 'Video type' section with radio buttons for 'Paid' and 'Free', and a 'Price' input field. At the bottom, there is a blue 'Go back' link and a green 'Upload video' button.

Figure 2 Register and upload forms

All the following pages use the same CSS style, these pages are the registration page, login page, upload page, account management page and logout page, along with a couple smaller forms. With the video upload page users can upload videos themselves. If a user wants to make a course, they can use videos that they have already uploaded and put them in a course.

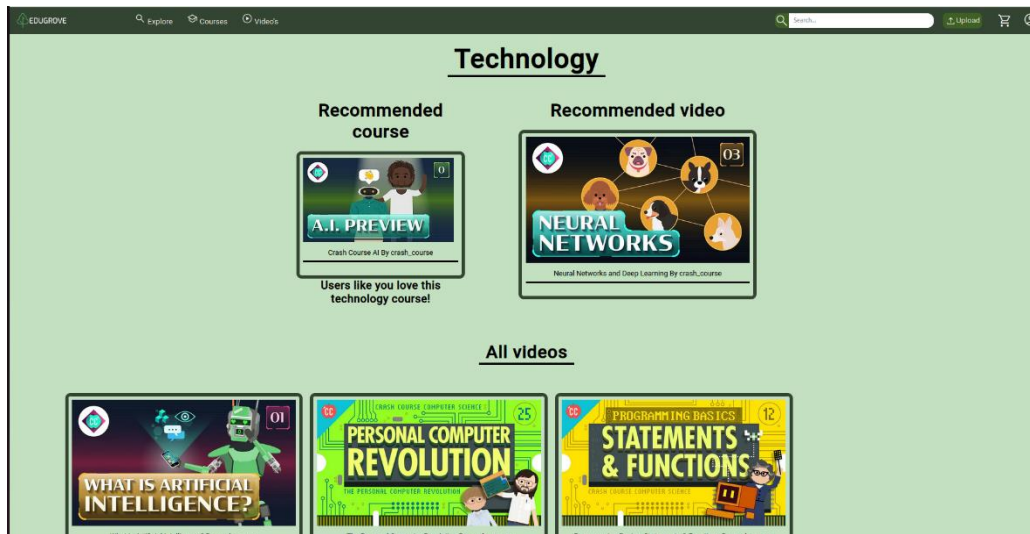


Figure 3 Subject page

If a user wants to explore a certain subject, for example technology, they can view all the videos, the recommend course, and the recommended video. The recommend course and video are based on rating.

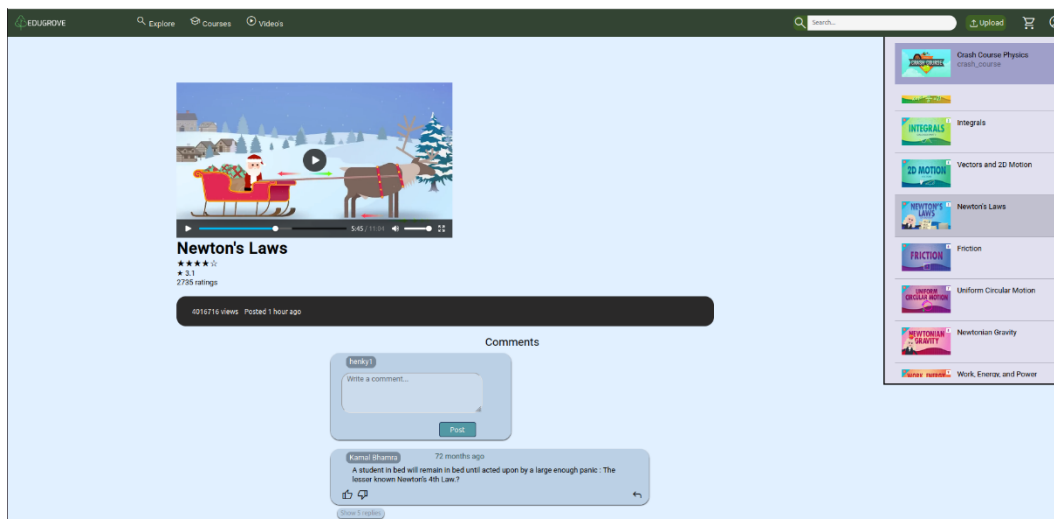


Figure 4 Video display page

When a user goes to a course, they can see all the videos of that course in order in the sidebar on the right. The video that is currently being watched is a bit darker, this way it is easier to see what the next video in the course is. If the course or video has been purchased, the user can watch the video, rate it, comment on it, comment on comments and rate the video between one or five stars. If a user stays on the video page for five seconds the view-count will be incremented by one. The views, the description and the comments are scraped from YouTube. The description becomes visible when you click on the black bar.

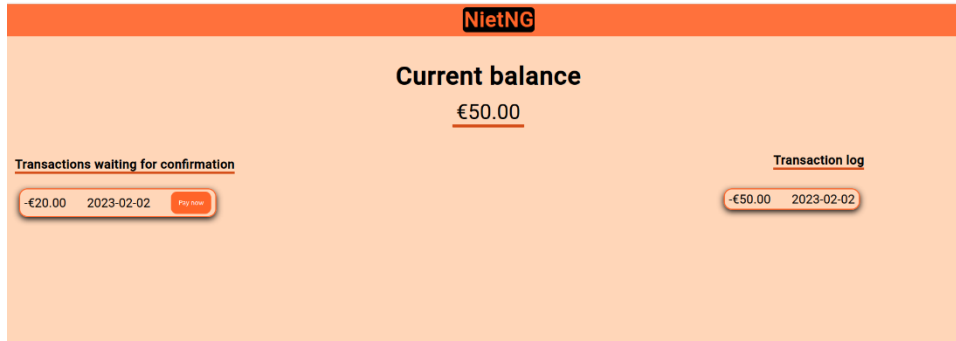


Figure 5 Bank page

When a user buys an item from Edugrove they will be redirected to the bank page 'NietNG'. On their account there is a certain amount of money. Upon buying an item, the price of the item will be withdrawn from the user's balance. The page also displays the pending and previous transactions.

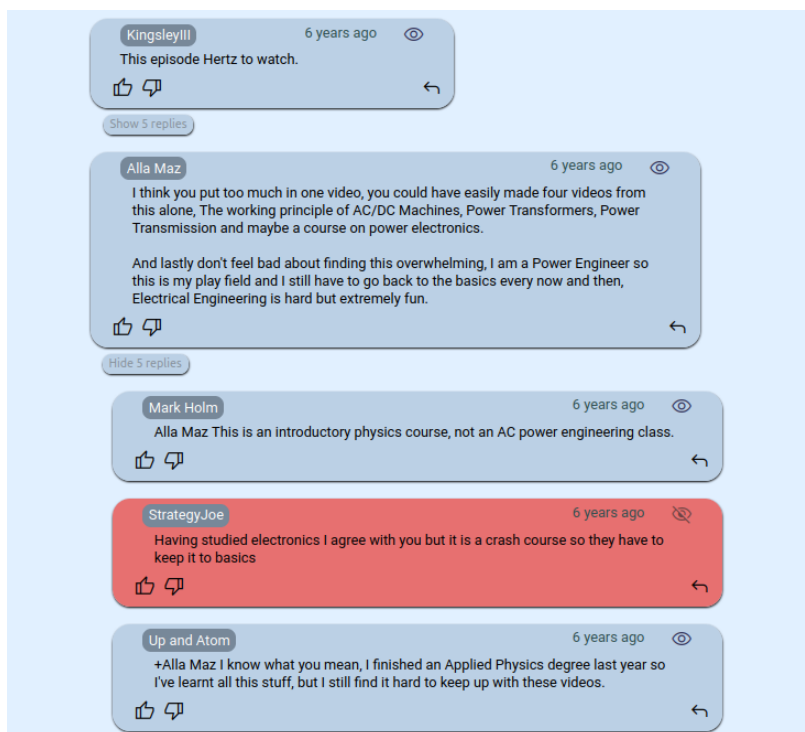


Figure 6 Admin view of comment section

Finally, administrators of the site are able to perform various tasks, such as:

- Restricting & unrestricting videos from the general public
- Banning & unbanning users, preventing them from using their account
- Hiding & unhiding comments that are deemed harmful (seen in red in the image)
- Gifting items, in case a purchase fails or compensation is required
- Authenticating as another user ('sudo') to check for potential problems with an account

## 5 Main achievements and advanced features

To create Edugrove a lot of time has been spent making the pages surrounding account functionality, such as the login, logout, and registration functions. As well as pages surrounding the courses, such as the explore page and the page where videos can be viewed. Although these pages can be seen as main achievements in creating Edugrove, they are not what makes the site shine. That title goes to the eight courses with over one hundred educational video's that are available on the site. The YouTube scraping that has been done to deliver all this content definitely takes the cake as the biggest achievement in the creation of Edugrove

Unfortunately, YouTube works with a like and dislike system to rate video's while on Edugrove you rate by giving one to five stars. To avoid a mismatch of 1.2 million views and two ratings, a function had to be created to artificially inflate the rating numbers. On server startup this function is run directly, it quadruples the startup time, from one to four seconds, because it is creating almost a million ratings and inserting them into the database.

Additionally, this website features both an online web shop and online forum even though only one of the two would have sufficed for this project, which is quite the accomplishment in and off itself

## 6 Discussion

As mentioned previously the definitive version of Edugrove included everything that it was supposed to have according to the initial expectations and goals, as well as lots of other unplanned functionalities which, nevertheless, made for an even better website.

First it is important to discuss the quality, and level of satisfaction surrounding the originally planned functionalities. To recap, those functionalities include but are not limited to: Account management, content creation, video viewing, course viewing, item acquisition, and simple content browsing.

Quite a few of the listed functionalities referred to remarkably simple and basic versions of functionalities you could encounter elsewhere on the web. Since the true scope of what could be achieved in four mere weeks was not entirely clear, most functionalities were jotted down with the idea that quite simple versions of them could certainly be implemented.

As time went on it became quite apparent that these basic functionalities could easily be expanded which is exactly what happened. So as time went on, the quality of the website improved which was great to see.

It is safe to say that every single function that was planned to be in the definitive version of the website has been completely and successfully realized. All the basic functionalities made it into the last version of Edugrove, and most of them are even more detailed than planned. Take the browse function for example, the plan was simply to create a page on the website where users could view every video on the platform, categorised by subject. This is the main video browse page in the last version, but on top of that users can sort the videos by rating, upload date, and views. The website also features a search bar as mentioned before, which was never really supposed to be a part of Edugrove since it seemed like it would have been too tricky to implement.

Something to mention is that the website currently houses videos and courses belonging to six unique subjects. It would be rather easy to add another subject or two to Edugrove so in that sense it would be quite simple to expand the website even further. It would also be quite easy to add other types of content outside of videos and courses. Something like a class, which would combine multiple courses belonging to the same subject so that viewers can get to experience more sides of a subject such as physics and learn how multiple parts of it play together.

That was just a quick idea, but it illustrates well how Edugrove could be expanded if someone really wanted to. The site is quite versatile and has a lot of room for potential growth and improvement. This is also in large part to the fact that the contents of the site are provided by the users themselves, so as more people use the site the larger its calibre of videos and courses grows. The site is self-sustaining if someone is willing to do the moderating work that is.

There is not a lot to change about the way the website was built up, looking back at the process in hindsight. A lot of time during the first week was spent on getting a few different applications



working so that the rest of the website would be easier to build. One of these applications was Docker, which allowed us to easily develop and iterate on designs locally. After a week of fiddling with it, it still was not in proper working order. This took quite a bit of work and cost a lot of time, so for a future project of this kind it would be advisable to put in some more preparation beforehand so no valuable time, that should be spend building the website, is wasted.