

**CCT College Dublin Continuous Assessment**

<b>Programme Title:</b>	BSc (Hons) in Computing in Information Technology		
<b>Cohort:</b>	September 2020 – Year 2 – Full Time		
<b>Module Title(s):</b>	Object Oriented Constructs (OOC) Linear Algebra (LA) Databases (DB)		
<b>Assignment Type:</b>	Pair or Individual/ Integrated	<b>Weighting(s):</b>	OOC (50%) LA (30%) DB (20%)
<b>Assignment Title:</b>	System of Linear Equations Calculator		
<b>Lecturer(s):</b>	Amilcar Aponte ( <a href="mailto:amilcar@cct.ie">amilcar@cct.ie</a> ) Aldana Louzan ( <a href="mailto:alouzan@cct.ie">alouzan@cct.ie</a> )		
<b>Issue Date:</b>	2 <sup>nd</sup> November 2021		
<b>Submission Deadline Date:</b>	Saturday, 18th December 2021 @ 23:59		
<b>Late Submission Penalty:</b>	Late submissions will be accepted up to <b>5</b> calendar days after the deadline (Friday, 24 <sup>th</sup> December 2021 @ 23:59). All late submissions are subject to a penalty of <b>10% of the mark awarded</b> . Submissions received more than 5 calendar days after the deadline above <b>will not</b> be accepted and a mark of 0% will be awarded.		
<b>Method of Submission:</b>	<b>Moodle</b>		
<b>Instructions for Submission:</b>	Single ZIP file that contains: <ul style="list-style-type: none"> <li>• Source code (Full NetBeans Project).</li> <li>• Documentation in PDF format.</li> </ul>		
<b>Feedback Method:</b>	<b>Results posted in Moodle gradebook</b>		
<b>Feedback Date:</b>	21st January 2021		

**STUDENTS:**

- **HEBER MOTA** (2020317)
- **YAN OLIVEIRA** (2020336)

**NOTE:** This project was developed in collaboration with two other students. Together, we discussed and researched on how to develop the many different challenges of the program. Students *Diego Lucas & Marcelo Urbano*.

---

# TABLE OF CONTENTS

<b>INTRODUCTION</b>	<b>3</b>
<b>CONCEPTUAL DIAGRAM</b>	<b>4</b>
Entities and Attributes	4
Relationships	5
<b>LOGICAL DESIGN</b>	<b>6</b>
<b>CHALLENGES FACED</b>	<b>7</b>
Graphical User Interface (GUI)	7
Security Measures	7
Debugging	8
Timing	8
Difficulty Level	8
<b>DATABASE SCRIPT</b>	<b>9</b>
<b>REFERENCES</b>	<b>10</b>

---

# INTRODUCTION

In this assignment, we were asked to **develop a program that** could be able to **solve and manage linear algebra equations** from both 2x2 and 3x3 variables by using matrices. The requirement was that first the user should login and depending on the permissions granted to them (Admin or Regular user) they would be able to perform tasks accordingly.

**The admin user** was pre-registered and with the right credentials (*Username: CCT / Password: Dublin*) they could **have access to administrator options**, such as history, modify their profile, see and manage users in the system. Whereas **the regular user**, who has no need to be pre-registered (as they don't have administrator permissions) could be registered with their personal data following the login criteria and **use the calculator to make operations** in the system.

Alongside with the application, it was also required to:

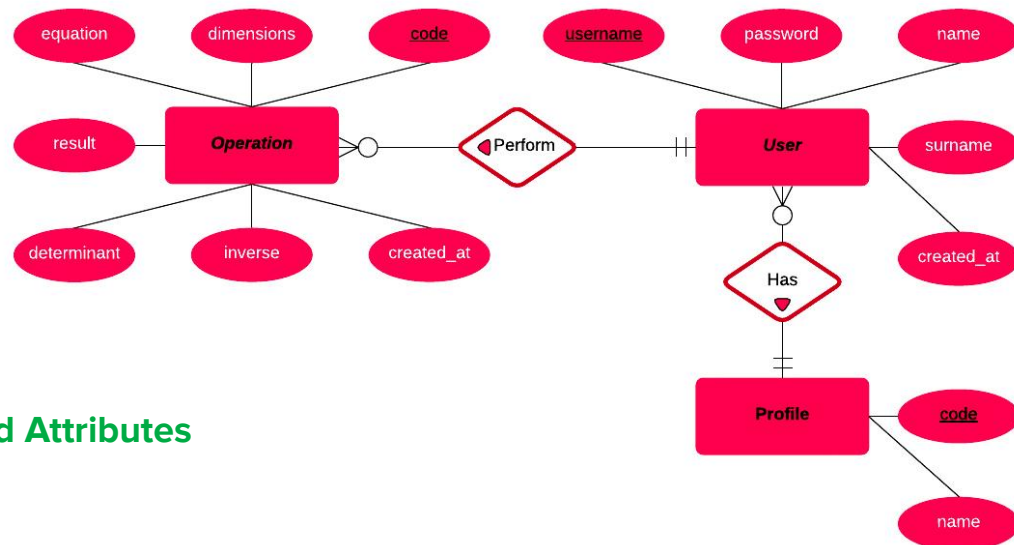
- provide a conceptual design diagram using CHEN notation which was made in [Lucidchart.com](https://lucidchart.com);
- develop a logical design and structure in the relational model;
- validate and determine the 3 normalization forms;
- explain the rationale of the design;
- make use of Java to build code and user interface;
- validate user inputs such as login details, passwords, etc;
- solve equations of 2x2 and 3x3 using matrices;
- keep a record of the operations and users in the database;

In this document, we will discuss the thinking process for the program as well as challenges faced from the beginning.

---

# CONCEPTUAL DIAGRAM

By using CHEN notation, the very first thing we did was creating the conceptual design of our application so we had a better idea of how our program would work/look like and so clarifying some of our doubts. Below can be seen that the main entities are Profile, User and Operation and so we built the system around these objects and how they are linked to each other.



## Entities and Attributes

1. **Profile:** this entity has as its primary key “code” or “profile-code” which defines the activity of the user being stored in the database as well as if they are a Regular user or an Administrator.
2. **User:** the User entity with the primary key “username” - once it is not possible for two users to have the same username - gathers the following data in order to create a new user in the system: name, surname, password and automatically, the date; if these criteria are not met, the system needs to show an error to show user that the fields need to be filled in thoroughly.
3. **Operation:** in the Operation entity, we have a few more attributes to consider once the operations have different things to consider, such as the dimensions (2x2 or 3x3), the equation, determinant, inverse (if there is any), result and why not, the date of creation. The “code” PK sets the criteria of the equations and also for the user.

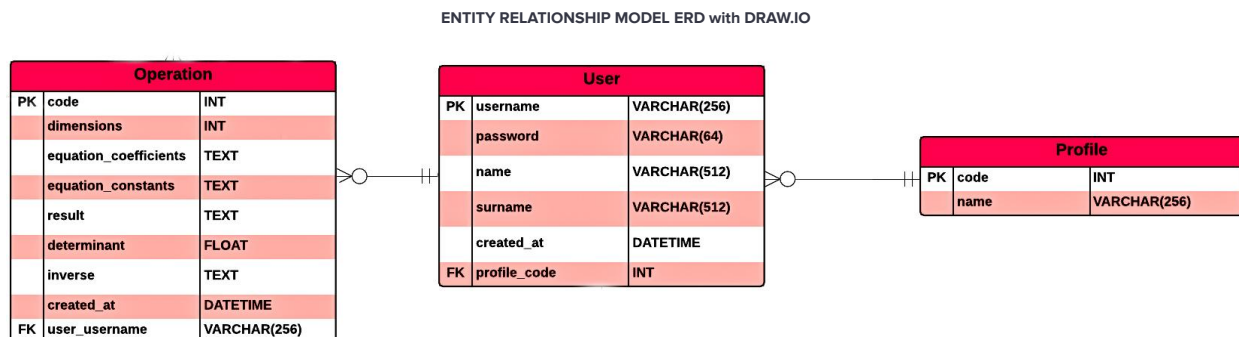
---

## Relationships

When it comes to the relationships among the entities, we realized that the first one to be considered would be the User entity. Because without having a user logged into the system, one would not be able to access the functionalities of the program, so the **User** entity is the one that leads the person **to have a profile** (and records of it) as well as to be able to **perform algebra operations** in our program.

# LOGICAL DESIGN

Here is the logical structure we developed basing on the previous conceptual diagram. Once again, the User entity is the one to start the connection with Profile (many to one) and Operation (one to many).



By following the **3 Normalization Forms** we made sure that:

1. All the attributes in the tables should be atomic and not be repeated;
2. The Primary Key has all of its attributes dependent on it;
3. All the attributes (not being the PK) are independent from each other but dependent on the primary key.

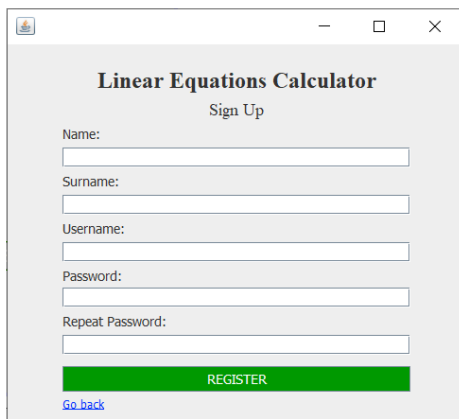
---

# CHALLENGES FACED

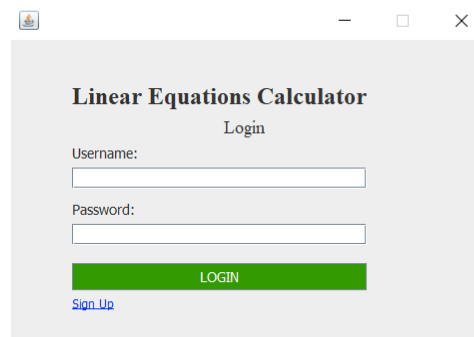
## Graphical User Interface (GUI)

As it was something new for us, it took some time to get the hang of the GUI Builder in Netbeans, but watching some tutorials and doing some research we managed to create an (almost) fully working interface. Not fully as we did not have enough time to increment it with all the functionalities, such as user management for the administrator, operation history and accessing all the users in the system, etc. The reason why will be listed in the topics that follow.

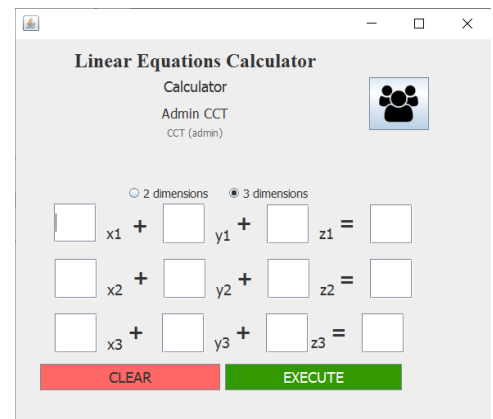
The design below was the final version of the UI developed for user input. All the respective errors related to input validation for the program criteria (ex.: fill in all gaps) are fully working as well.



The screenshot shows a window titled "Linear Equations Calculator" with a subtitle "Sign Up". It contains five text input fields: "Name:", "Surname:", "Username:", "Password:", and "Repeat Password:". Below the fields is a green "REGISTER" button and a blue "Go back" link.



The screenshot shows a window titled "Linear Equations Calculator" with a subtitle "Login". It contains two text input fields: "Username:" and "Password:". Below the fields is a green "LOGIN" button and a blue "Sign Up" link.



The screenshot shows the main interface of the "Linear Equations Calculator". It has a title bar and a subtitle "Calculator". Below the subtitle are two radio buttons: "2 dimensions" and "3 dimensions", with "3 dimensions" selected. There are three rows of input fields for coefficients:  $x_1, y_1, z_1$ ;  $x_2, y_2, z_2$ ; and  $x_3, y_3, z_3$ . Each row is followed by an equals sign and a result field. At the bottom are two buttons: a red "CLEAR" button and a green "EXECUTE" button.

## Security Measures

In fairness, we did not think about setting up security for the user password as it is not mandatory for this project, but during our research, we found out that it could be possible to **encrypt user password** for better security **via SHA-256** (*Secure Hash Algorithm*), so when a user logs in, we have the encryption of this password (within 64 characters) saved into a 256 digest size in bits encryption. So we linked the SHA-256 instance to a

---

class named Security.java which worked successfully easier than we expected.

## Debugging

Without a shadow of a doubt, debugging was the greatest challenge faced as it is very time consuming to manage to solve even small errors while coding. Thankfully with help of each other and research we got them fixed and kept going with the project.

A valid example of when we got stuck was:

When having technical issues with both Netbeans and Workbench. Which was mainly in my case (Heber). For an unknown reason some libraries would not work as well as the connection with the database that would keep asking for a password that I could not figure out. It took about 4 days for me to solve this problem after uncountable unsuccessful attempts by simply uninstalling both of them - this time with all their files that were found left in my machine and reinstalling again, and this time with a newer version (Apache Netbeans IDE 12.6) and again the Workbench and the server.

Apparently, some files were interfering in the functioning of my Netbeans as well as the Workbench that was misconfigured and just when they were fully uninstalled and downloaded again I could get them to work normally. After doing that, I was able to make use of external library such as JSON that was used to save matrix structures into the database in a more practical way to save and retrieve data.

## Timing

Although more than one month was given to accomplish this work, it was still not enough for us. At the beginning we had no idea of how to carry out a project with this complexity and it made us very overwhelmed and feeling at sea. But once we kickstarted with the conceptual diagram that gave us a brief of how it should look like, we had then a better idea of it.

---

Then, it took us a long time to discuss how it would be done. Days of research were taken in order to give us a northstar as well as in many other steps of the program such as the calculator, the database script, the user interface and the many different classes it has for each functionality. We just managed to get as far as we got with the help of two other students who worked with us to solutionate the issues found along the way, otherwise we probably would not be able to get it done the way it was.

For this reason, we were not able to finish the project as a whole, and as soon we realized we did not have enough time for doing so, we decided to polish it as we could to deliver a **fully working program, even that not fully completed.**

## Difficulty Level

I could not finish this report without mentioning how difficult it was to achieve what was requested from lecturers - especially from the programming part. Most of our headache was due to the fact that we had never been taught how to bring all of this together and now we were just expected to accomplish it somehow - extremely frustrating!

**We spent the longest time doing research of what should have been taught in class** looking for it on google and developer websites such as *Stackflow* and *GeeksforGeeks*. So I hope lecturers take it into consideration when grading because a fully working program with such complexity cannot be expected to be accomplished perfectly by most students who never had experience with Java or MySQL before.



```

/* CREATE DATABASE IF DOES NOT EXIST */
CREATE DATABASE IF NOT EXISTS linear_equations_calculator_db;

USE linear_equations_calculator_db;

/* CREATE TABLES IF DOES NOT EXIST */
CREATE TABLE IF NOT EXISTS profile(
    code INT NOT NULL,
    name VARCHAR(256) NOT NULL,
    PRIMARY KEY (code)
);

CREATE TABLE IF NOT EXISTS user(
    username VARCHAR(256) NOT NULL,
    password VARCHAR(64) NOT NULL,
    name VARCHAR(512) NOT NULL,
    surname VARCHAR(512) NOT NULL,
    created_at DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
    profile_code INT NOT NULL,
    PRIMARY KEY (username),
    CONSTRAINT FK_user_profile FOREIGN KEY (profile_code) REFERENCES profile(code)
);

CREATE TABLE IF NOT EXISTS operation(
    code INT AUTO_INCREMENT NOT NULL,
    dimensions INT NOT NULL,
    equation_coefficients TEXT NOT NULL,
    equation_constants TEXT NOT NULL,
    result TEXT NOT NULL,
    determinant FLOAT NOT NULL,
    inverse TEXT,
    created_at DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
    user_username VARCHAR(256) NOT NULL,
    PRIMARY KEY (code),
    CONSTRAINT FK_operarion_user FOREIGN KEY (user_username) REFERENCES user(username)
);

/* INSERT MANDATORY DATA */
INSERT INTO profile (code, name) VALUES(1, 'admin');
INSERT INTO profile (code, name) VALUES(2, 'regular');

INSERT INTO user (username, password, name, surname, profile_code)
VALUES('CCT', '0a8ad51fea30c29587f61bd8420a49602aa78304833627829cd29d7594ce3c57', 'Admin', 'CCT', 1);

/* INSERT DATA FOR EXAMPLE */
INSERT INTO operation (dimensions, equation_coefficients, equation_constants, result, determinant, inverse, user_username)
VALUES(
    3,
    '{"matrix":["[1,2,1],[2,-1,1],[-1,3,1]]'}',
    '{"matrix":["0,1,2]]'}',

```

---

```

        '\matrix\':[-1.4,-0.8,3]],
        -5.0,
        '\matrix\':[[0.8,-0.2,-0.6],[0.6,-0.4,-0.2],[-1,1,1]]',
        'CCT'
    );

```

```

INSERT INTO operation (dimensions, equation_coefficients, equation_constants, result, determinant, inverse, user_username)
VALUES(
    2,
    '\matrix\':[[1,0],[1,0]]',
    '\matrix\':[1,2]',
    '\matrix\':\''inconsistent\'',
    0.0,
    null,
    'CCT'
);

```

```

INSERT INTO operation (dimensions, equation_coefficients, equation_constants, result, determinant, inverse, user_username)
VALUES(
    3,
    '\matrix\':[[0,0,0],[0,0,0],[0,0,0]]',
    '\matrix\':[0,0,0]',
    '\matrix\':\''infinity solutions\'',
    0.0,
    null,
    'CCT'
);

```

```

INSERT INTO operation (dimensions, equation_coefficients, equation_constants, result, determinant, inverse, user_username)
VALUES(
    2,
    '\matrix\':[[1,-5],[3,5]]',
    '\matrix\':[15,-5]',
    '\matrix\':[2.5,-2.5]',
    20.0,
    '\matrix\':[[0.25,0.25],[-0.15,0.05]]',
    'CCT'
);

```

```

INSERT INTO operation (dimensions, equation_coefficients, equation_constants, result, determinant, inverse, user_username)
VALUES(
    3,
    '\matrix\':[[2,1,-1],[-3,-1,2],[-2,1,2]]',
    '\matrix\':[8,-11,-3]',
    '\matrix\':[2,3,-1]',
    -1.0,
    '\matrix\':[[4,3,-1],[-2,-2,1],[5,4,-1]]',
    'CCT'
);

```

```

INSERT INTO operation (dimensions, equation_coefficients, equation_constants, result, determinant, inverse, user_username)
VALUES(
    2,
    '\matrix\':[[-1,1],[1,-1]]',
    '\matrix\':[0,0]',
    '\matrix\':\''infinity solutions\'',
    0.0,
    null,
    'CCT'
);

```

---

);

# REFERENCES

- Wagner, L., 2020. How SHA-256 Works Step-By-Step. [online] qvault.io. Available at: <<https://qvault.io/cryptography/how-sha-2-works-step-by-step-sha-256/>> [Accessed in December 2021].
- Youtube.com. 2020. Java Connect to MySQL Database Step by Step. [online] Available at: <<https://www.youtube.com/watch?v=duEkh8ZsFGs>> [Accessed in December 2021].
- Youtube.com. 2016. Java Calculator App Development Tutorial 1 | Swing | GUI. [online] Available at: <[https://www.youtube.com/watch?v=\\_ZW4ktG1DEE](https://www.youtube.com/watch?v=_ZW4ktG1DEE)> [Accessed in December 2021].
- Sokoloski, P., 2016. Validating Linear Equations and Functions with Matrix. [online] Matrixcalc.org. Available at: <<https://matrixcalc.org/>> [Accessed in December 2021].
- Mkyong.com. 2020. Java SHA-256 and SHA3-256 Hashing Example - Mkyong.com. [online] Available at:

---

<<https://mkyong.com/java/java-sha-hashing-example/>> [Accessed in December 2021].

- GeeksforGeeks. 2021. Gaussian Elimination to Solve Linear Equations - GeeksforGeeks. [online] Available at: <<https://www.geeksforgeeks.org/gaussian-elimination/>> [Accessed in December 2021].
- GeeksforGeeks. 2021. Java Program to Find the Determinant of a Matrix - GeeksforGeeks. [online] Available at: <<https://www.geeksforgeeks.org/java-program-to-find-the-determinant-of-a-matrix/>> [Accessed in December 2021].
- GeeksforGeeks. 2021. Adjoint and Inverse of a Matrix - GeeksforGeeks. [online] Available at: <<https://www.geeksforgeeks.org/adjoint-inverse-matrix/>> [Accessed in December 2021].
- baeldung. 2021. The DAO Pattern in Java. [online] Available at: <<https://www.baeldung.com/java-dao-pattern>> [Accessed in December 2021].
- Caelum. Undated. Caelum Escola de Tecnologia Cursos Online. [online] Available at: <<https://www.caelum.com.br/apostila-java-web/bancos-de-dados-e-jdbc#exercicios-javabeans-e-contatodao>> [Accessed in December 2021].
- Posts, B., 2021. Java JSON Tutorial and Example: JSON-Java (org.json). [online] Codevoila.com. Available at: <<https://www.codevoila.com/post/65/java-json-tutorial-and-example-json-java-orgjson>> [Accessed in December 2021].

- 
- Jdoodle.com. 2021. JDoodle - free Online Compiler, Editor for Java, C/C++, etc. [online] Available at: <<https://www.jdoodle.com/online-java-compiler/>> [Accessed in December 2021].
  - GeeksforGeeks. 2021. Java Program to Represent Linear Equations in Matrix Form - GeeksforGeeks. [online] Available at: <<https://www.geeksforgeeks.org/java-program-to-represent-linear-equations-in-matrix-form/>> [Accessed in November 2021].
  - Ejml.org. Undated. Solving Linear Systems - Efficient Java Matrix Library. [online] Available at: <[https://ejml.org/wiki/index.php?title=Solving\\_Linear\\_Systems](https://ejml.org/wiki/index.php?title=Solving_Linear_Systems)> [Accessed in November 2021].
  - pothe, o., 2021. Solving linear eqn by efficient-java-matrix-library. [online] Stack Overflow. Available at: <<https://stackoverflow.com/questions/22395533/solving-linear-eqn-by-efficient-java-matrix-library>> [Accessed in November 2021].
  - NetBeans, A., 2021. Designing a Swing GUI in NetBeans IDE. [online] Netbeans.apache.org. Available at: <<https://netbeans.apache.org/kb/docs/java/quickstart-gui.html>> [Accessed in December 2021].
  - Youtube.com. 2019. IE 3 Java Swing GUI Programming Tutorial | Java Swing (Graphical User Interface) Tutorial. [online] Available at: <[https://www.youtube.com/watch?v=mDxEgtMNPtA&ab\\_channel=ProgrammingKnowledge](https://www.youtube.com/watch?v=mDxEgtMNPtA&ab_channel=ProgrammingKnowledge)> [Accessed in December 2021].

- 
- Lee, A., 2020. Java GUI Tutorial - Make a Login GUI. [online] Youtube.com. Available at: <[https://www.youtube.com/watch?v=iE8tZ0hn2Ws&ab\\_channel=AlexLee](https://www.youtube.com/watch?v=iE8tZ0hn2Ws&ab_channel=AlexLee)> [Accessed in December 2021].
  - Willems, K., 2019. SQL Tutorial: How To Write Better Queries. [online] datacamp. Available at: <<https://www.datacamp.com/community/tutorials/sql-tutorial-query>> [Accessed in November 2021].
  - Docs.oracle.com. n.d. Using SQL Scripts. [online] Available at: <[https://docs.oracle.com/cd/E14373\\_01/user.32/e13370/sql\\_rep.htm#A EUTL192](https://docs.oracle.com/cd/E14373_01/user.32/e13370/sql_rep.htm#A EUTL192)> [Accessed in November 2021].
  - Youtube.com. 2018. SQL Tutorial - Full Database Course for Beginners. [online] Available at: <[https://www.youtube.com/watch?v=HXV3zeQKqGY&ab\\_channel=freeCodeCamp.org](https://www.youtube.com/watch?v=HXV3zeQKqGY&ab_channel=freeCodeCamp.org)> [Accessed in November 2021].
  - Youtube.com. 2019. Advanced SQL course | SQL tutorial advanced. [online] Available at: <[https://www.youtube.com/watch?v=2Fn0WYyZV0E&ab\\_channel=GEEK%27sLesson](https://www.youtube.com/watch?v=2Fn0WYyZV0E&ab_channel=GEEK%27sLesson)> [Accessed in December 2021].
  - Youtube.com. 2018. How to Solve a System of Equations Using Cramer's Rule: Step-by-Step Method. [online] Available at: <[https://www.youtube.com/watch?v=LprQ\\_Id-8hE](https://www.youtube.com/watch?v=LprQ_Id-8hE)> [Accessed in November 2021].