"VIDEO STREAMING APPLICATION"

A CS814 Course Project Report

SUBMITTED BY

Mr. VIVEK JAIN (202IS029)

Mr. SHIVAM PANDEY (202IS024)

DEPARTMENT OF COMPUTER SCIENCE AN ENGINEERING

National Institue of Technology Karnataka

P.O. Srinivasnagar, Surathkal, Mangalore-575025 Karnataka, India

January 2021

Contents

List of Figures

Chapter 1

INTRODUCTION

1.1 Introduction

We have created an online video streaming web application "Video Streamer" as our project. It is an online video streaming site which will allow anyone to watch videos online in few clicks. We have put all our efforts to make the project user friendly by making him access and stream videos with great ease. The project is divided into different modules so that it can be easily maintained and can be scaled if needed for a larger user base.

1.2 Motivation

In early times of internet if a web administrator wants to add videos to his website, he has to post it as a link. After that the website visitors have to download the entire video before playing it. But this entire scenario changed once the live streaming web applications came into picture. Here, the content is served in a way that allows files to play almost immediately after the files begin to download. With streaming video or streaming media users will not have to wait for entire file to download before playing. With streaming they can watch any video with just a single click !

We have seen a lot of success in the recent times in the online video streaming segment with the advent of web applications like Youtube and Netflix and our project is targeted to ride on the current trend.

1.3 Use of Application

1.4 Types of Users

1.4.1 Free Users

Any User can Sign Up to our application by giving some basic details. By default, when a user signs up, he will be a Free User. A free user have only limited access to the web application. Once a free user takes the Subscription, he gets converted into a Premium User.

1.4.2 Premium Users

Premium Users are paid subscribers of our application. They can enjoy more functionalities than Free Users.

1.4.3 Admin Users

Admin Users can manage the application. Their roles and functionalities are discussed in ARBAC section.

1.5 Functionalities

1.5.1 Sign Up

- This functionality is added for user registration.
- Once a user signs up, he is known to our application.
- After Sign Up, he can log in any time and enjoy watching videos.
- It is the first and least thing required to access our application.

1.5.2 Browse Videos

- This is the basic functionality of our web application.
- It includes browsing through the list of videos present in the database.
- It is available for all category of users.
- Sign Up is required for a user to access this functionality.

1.5.3 Play Video

- This is also the basic functionality of our web application.
- It includes playing/streaming a video among the list of videos present in the web application database.
- It is also available for all category of users.
- Sign Up is required for a user to access this functionality.

1.5.4 Upload Video

- This functionality is available only to specific category of users.
- It includes uploading videos to the web application.
- It is available only for Premium users.
- Subscription is required for a user to access this functionality.

Chapter 2

Authentication and Authorization

2.1 Authentication

We have added functionalities to authenticate a user and prevent his account malicious use.

2.1.1 Log In

This is required every time a user tries to access our application. In this process we check the authenticity of a user by matching the details he provides at the time of Log In with the details he had already provided at the time of Sign Up that are already present in our database. If the details are matched then the access is granted otherwise its denied.

2.1.2 Log Out

This functionality is added to prevent misuse of our user's account. When a user wants to stop watching videos then he can simply Log Out from the application, then his account is no more active and anyone else can not continue using his account.

2.2 Authorization

We have different types of Users in our system. To control user functionalities on the basis of their roles and responsibilities, we need to have an access control mechanism for our application. Hence, we are implementing an authentication system.

2.3 Need for RBAC

We studied different access control systems and concluded that Role Based Access Control can solve our purpose. In our application, we have implemented $RBAC_0$.

2.4 Components of RBAC

2.4.1 Users

A person who sign up in our application is a User. Fig.2.1 shows a snapshot of Users table in our database. Column id is the primary key.

id	username	first_name	last_name	email
1	vivek			vivekj52@gmail.com
2	shivam	Shivam	Pandey	202is024@nitk.edu.in
5	vivekj52	vivek	jain	vivekj52@gmail.com
10	suryanitk	surya	gupta	surya@gmail.com
11	tridiv	tridiv	rabha	tridiv@nitk.edu.in

Figure 2.1: Users Table

2.4.2 Groups

Group is a set of Users. In our application, we have three groups, namely free, premium and admin. Column id is the primary key of table.

id	name
3	admin
2	free
1	premium
NULL	NULL

Figure 2.2: Groups Table

2.4.3 Permissions

Permissions are the operations a User can perform on a data object. Fig.2.3 shows a snapshot of permissions table in our database. Column id is the primary key of table.

2.4.4 User-Group Relation

We have a many to many relationship between Users and Groups. A Group can have many Users and a User can be part of many Groups.

2.4.5 Group-Permission Relation

We have many to many relationship between Groups and Permissions. A Group can have many permissions. A permission can be associated with many groups.

id	name	codename
5	Can add permission	. add_permission
6	Can change permission	. change_permission
7	Can delete permission	. delete_permission
8	Can view permission	. view_permission
9	Can add group	. add_group
10	Can change group	. change_group
11	Can delete group	. delete_group
12	Can view group	. view_group
13	Can add user	. add_user
14	Can change user	. change_user
15	Can delete user	. delete_user
16	Can view user	. view_user
25	Can add video	. add_video
26	Can change video	. change_video
27	Can delete video	. delete_video
28	Can view video	. view_video

Figure 2.3: Permissions Table

id	user_id	group_id
5	1	3
3	2	1
1	5	1
2	10	2
4	11	2

Figure 2.4: User-Group Relation Table

2.5 Need for ARBAC

In our application, there are different types of users. Some users can also upload videos which can be viewed by all the users.

To manage the user details, their groups and group permissions, we need a admin user for our application. Admin role is also needed to validate the video content and details. Some videos might not follow community standards. To manage all this, we need a admin control for our application.

2.6 Components of ARBAC

We have two components of Administrative Role Based Access Control:

2.6.1 Administrative Role

We have created a admin group in admin role. All the admin users will be added to admin group.

id	group_id	permission_id
1	1	25
2	1	28
3	2	28
4	3	5
5	3	6
6	3	7
7	3	8
8	3	9
9	3	10
10	3	11
11	3	12
12	3	13
13	3	14
14	3	15
15	3	16
16	3	25
17	3	26
18	3	27
19	3	28

Figure 2.5: Group-Permission Relation Table

2.6.2 Administrative Permissions

Administrative permissions are only assigned to admin group. In our project, we have given following permissions to admin group:

- Can add user
- Can change user
- Can delete user
- Can view user
- Can add group
- Can change group
- Can delete group
- Can view group
- Can add permission
- Can change permission
- Can delete permission

- Can view permission
- Can add video
- Can change video
- Can delete video
- Can view video

Chapter 3

IMPLEMENTATION OF PROJECT

3.1 Project Modules

We have two modules in the application:

3.1.1 Authentication and Authorization

All the authentication logic including Signup, Login and Logout are present in this module.

3.1.2 Streamer

Video browsing, streaming and uploading logic are present in this module.

3.2 Technologies Used

We have used following technologies for developing, testing and version control of our project with each one serves a different purpose.

Software Specifications

- Operating System: Ubuntu 20.04 LTS
- Languages Used: Python, HTML, CSS
- Framework: Django
- Database: MySQL
- IDE: PyCharm
- Version Control: Git
- Browser: Firefox, Chrome
- Text Editor: Sublime Text, VS Code

3.3 Database Connectivity

We have used relational database MySQL server in our application. We have used 'django.db.backends.mysql' package for database connection.

3.4 Screenshots of Project

3.4.1 Sign Up Page

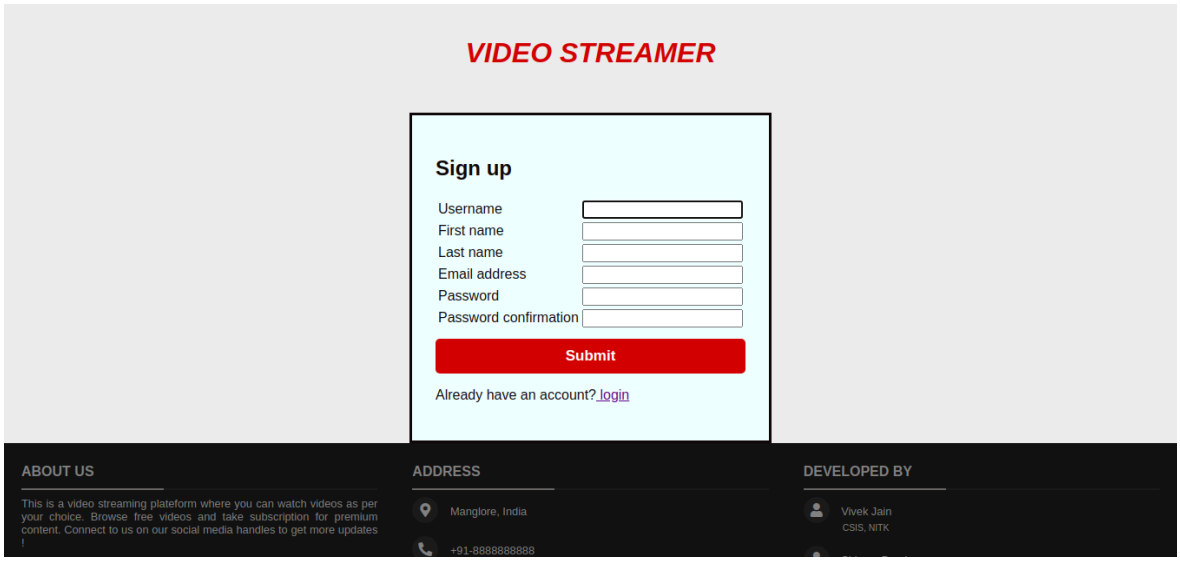

Figure 3.1: Sign Up Page

3.4.2 LogIn Page

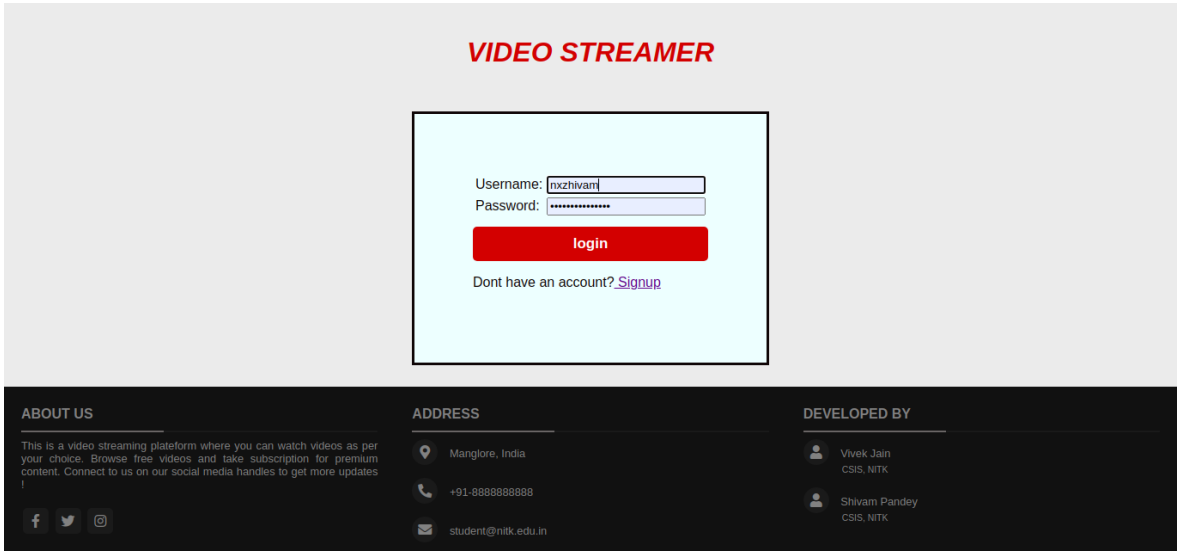

Figure 3.2: LogIn Page

3.4.3 Video Player

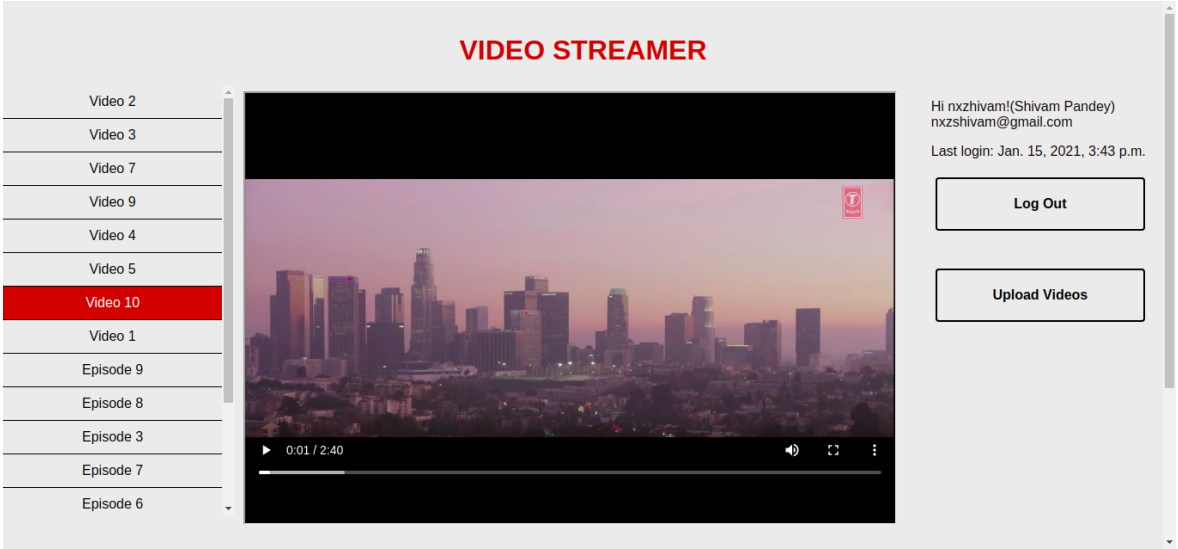

Figure 3.3: Video Player

3.4.4 Upload Page

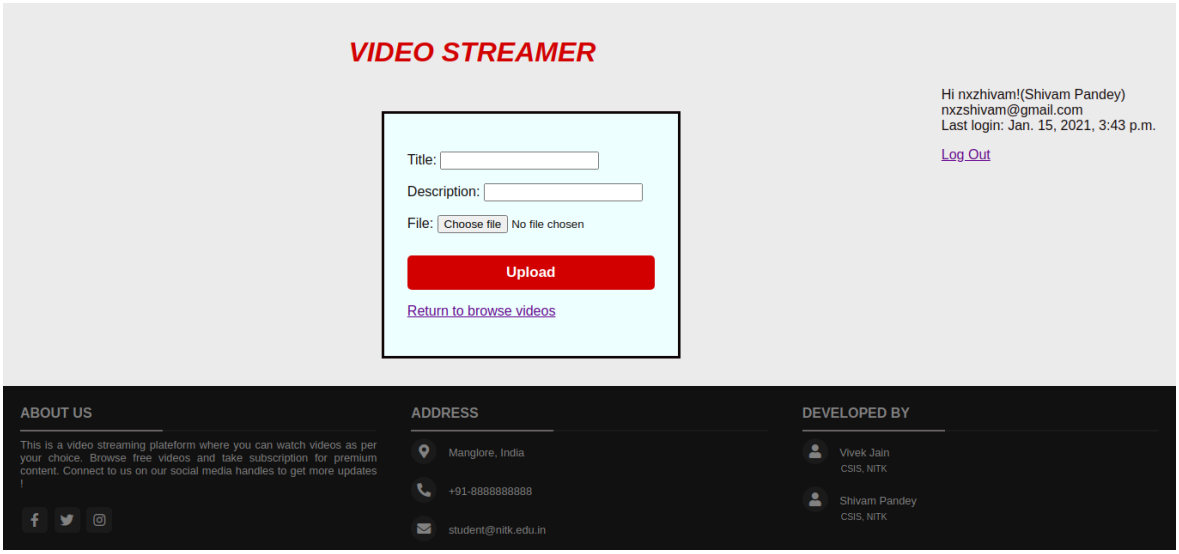

Figure 3.4: Upload Page

Chapter 4

CONCLUSION

Video Streaming is a technology that has completely changed the entertainment industry as well as consumption models among audience. A lot has changed since that very first Real Player transmission in 1995. Since then, technology has been constantly improving, making content delivery and access easier no matter the platform trying to access it. Netflix, Inc. is one of the best examples regarding commercial applications for Video Streaming. With millions subscribing to the service all over the world, the company has found a way to capitalize it services using it's title stock and outsourced infrastructure.

In our project we have developed a Video Streaming Application and implemented RBAC and ARBAC policies which helped us to understand that how these policies are implemented in real time. This project also gave us a clear understanding of how much these access control models are necessary for better management of an application.

Bibliography

[1] Ravi Sandhu, Venkata Bhamidipati, and Qamar Munawer. 1999. The ARBAC97 model for role-based administration of roles. ACM Trans. Inf. Syst. Secur. 2, 1 (Feb. 1999),105–135. DOI:https://doi.org/10.1145/300830.300839

[2] Ravi S. Sandhu, Edward J. Coyne, Hal L. Feinstein, and Charles E. Youman. 1996. Role-Based Access Control Models. Computer 29, 2 (February 1996), 38–47. DOI:https://doi.org/10.1109/2.485845

[3] Mahendra Pratap Singh, Shamik Sural, Jaideep Vaidya, and Vijayalakshmi Atluri. Managing attribute-based access control policies in a unified framework using data warehousing and in-memory database. Computers security 86 (2019): 183-205. DOI:https://doi.org/10.1016/j.cose.2019.06.001

[4] `https://docs.djangoproject.com/en/3.1/contents/`