

Screenify

AN INTERNSHIP PROJECT REPORT

Submitted by

Parmar Dixit

[190130107076]

In partial fulfillment for the award of the degree Of

BACHELOR OF ENGINEERING

in

Computer Enginnering

Government Engineering College, Gandhinagar



GTU, Ahmedabad

April,2023



Government Engineering College

Sector 28 GIDC, Gandhinagar

Gujarat - 382028, India

CERTIFICATE

This is to certify that the internship report submitted along with the internship entitled **Screenify** has been carried out by **Parmar Dixit NareshBhai** under my guidance in partial fulfillment for the degree of Bachelor of Engineering in Computer Engineering, 8th Semester of GTU, Ahmedabad during the academic year 2023-24

Prof. Hemani Shah

Internal Guide

Prof. Jitendra Kumar Dhobi

Head of the Department



Government Engineering College, Gandhinagar

Sector 28 GIDC, Sector 28, Gandhinagar

382028, Gujarat, India

DECLARATION

We hereby declare that the Internship report submitted along with the Internship entitled **Screenify** submitted in partial fulfillment for the degree of Bachelor of Engineering in Information Technology to GTU, Ahmedabad, is a bonafide record of original project work carried out by me at **La Net Team Software Solutions Pvt. Ltd.** Under the supervision **Mr. Hitesh Pandya** and that no part of this report has been directly copied from any students report or taken from any other source, without providing due reference.

Name of the Student

Sign of Student

Parmar Dixit

ACKNOWLEDGEMENT

I would like to express my sincere gratitude and appreciation to all those who have supported me during my internship at La Net Team Software Solutions Pvt. Ltd. I am thankful to Mr. Hitesh Pandya, who provided me with constant guidance, encouragement, and support throughout my internship. Their vast knowledge, valuable insights, and expertise in the field of Web Development that helped me develop new skills and broaden my understanding of the industry. I am also grateful to the entire team at La Net Team Software Solutions Pvt. Ltd for providing me with a conducive work environment and giving me the opportunity to work on various projects that have helped me improve my technical skills. I would also like to thank my colleagues for their support and valuable feedback that helped me learn and grow during my internship.

Finally, I am grateful to everyone who contributed to my learning experience at La Net Team Software Solutions Pvt. Ltd. It was an invaluable experience that I will take with me throughout my career.

Parmar Dixit

ABSTRACT

Screenify is a web application built using React JS and Node.js with MongoDB as the backend database. The website provides a sleek and user-friendly interface where users can browse and stream movies and TV shows. The project leverages the power of MongoDB's document-based structure to store and retrieve movie and TV show information, including availability and recommended content. Users can create and manage their accounts, save their favorite movies and TV shows, and get personalized recommendations based on their viewing history. The site also has an admin portal where administrators can manage content, view user history, and update recommendations. Overall, Screenify is designed to provide a seamless and enjoyable streaming experience for users while providing easy management for administrators.

LIST OF FIGURES

Figure No	Figure Description	Page No
Figure 4.1.1	Class Diagram	10
Figure 4.1.2	Activity Diagram	11
Figure 4.2.1	MongoDB Connection	12
Figure 4.2.2	Schema Design(User)	13
Figure 4.2.3	Schema Design(Movie)	14
Figure 4.2.4	Schema Design(List)	15
Figure 4.2.5	MongoDB Atlas	16
Figure 4.2.6	PostMan API	17

TABLE OF CONTENTS

Acknowledgement	i
Abstract	ii
List of Tables	iv
Table of Contents	vi
Chapter: 1	Organization
	1.1. History of Organization
	1.2. Scope of Work
	1.3. Organization Chart
	1.4. Capacity of Organization
Chapter: 2	Introduction to Project
	2.1 Introduction
	2.2 Purpose
	2.3 Objective
	2.4 Scope
	2.5 Technologies used
Chapter: 3	Learning Phase
	● Learning The Fundamentals
	● Learning about various Applications
	● JavaScript
	● NodeJs
	● ExpressJs
	● NestJs

Chapter: 4	System Design	9
	4.1 Detail Design	10
	4.1.1 Use-case Diagram	10
	4.1.2 Activity Diagram	11
	4.2 Database Design	12
	4.3 Screens	0
Chapter: 5	Implementation Planning	20
	5.1 Implementation Environment	21
	5.2 Program/Modules Specification	21
	5.3 Coding Standards	21
	5.4 Coding Scenario	22
Chapter: 6	Testing	23
	6.1 Testing Plan/ Strategy	24
	6.2 Testing Methods	25
	6.2.1 Test Cases	26
Chapter: 7	Limitations and Future Enhancement	30
	7.1 Limitations	31
	7.2 Future Enhancement	31
Chapter: 8	Conclusions	33
References		34

Training Certificate

This is to certify that **Mr. Parmar Dixitkumar Nareshkumar**, is working at **La Net Team Software Solutions Pvt Ltd** as a **Software Developer Trainee** from **23rd January 2023** to **22nd April 2023**.

During the training, he was found sincere and hardworking.

We wish him all the best in his future endeavors.



Palak Sharma
(HR Executive)

WWW.LANETTEAM.COM



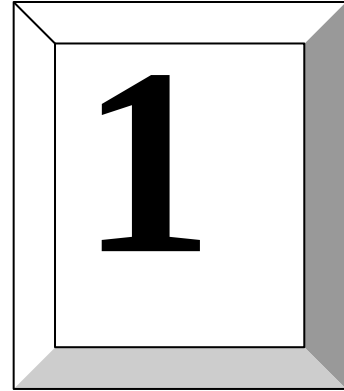
406 Luxuria Business Hub, Near VR mall, Surat-Dumas Rd, Surat, Gujarat-395007



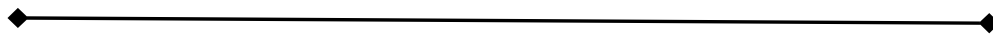
+91 6353235503



hr@lanetteam.com



Chapter 1: Organization



- 1.1. History of Organization**
- 1.2. Scope of Work**
- 1.3. Organization Chart**
- 1.4. Capacity of Organization**

1.1. History of Organization:

La Net Team is an India based software outsourcing company that offers high quality and cost effective software development service to its clients. We strive for on time delivery of the projects and adhere to stringent quality standards.

We offer flexible and cutting edge solutions that help our clients to operate more efficiently and gain an edge over their competitors. We offer high quality offshore software engineering and programming talent that can be leveraged to gain competitive advantage. We believe in delivering smart business solutions through smart usage of technology and continuously focus on quality of deliverables to our clients.

Our wide range of services include website designing, e-commerce services, Android and IOS Applications, Graphic and Logo design, Website Templates, Website Marketing, Product Development, Software Development etc.

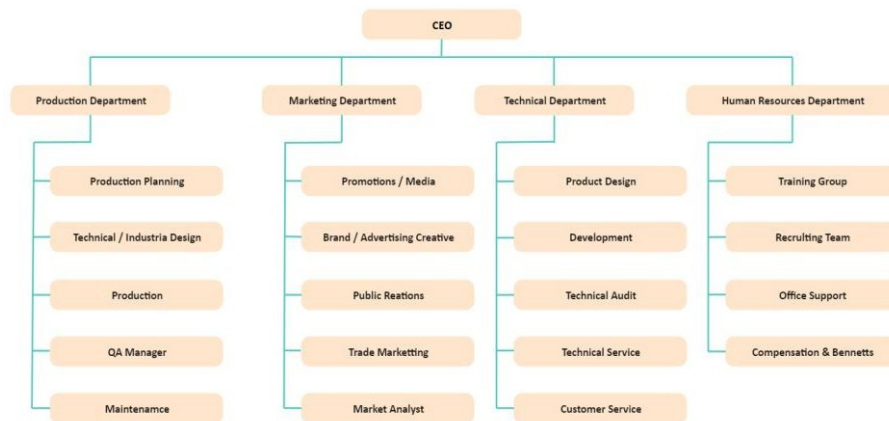
Our local presence guarantees cost advantages specially for the key locations that we are situated in. We strive to upgrade our domain expertise to help us provide not only IT services but valuable business services to our clients. Our primary goal is to help the client to focus on his business and leave the onus upon us to deliver what they need to run their business and make it more competitive through technology.

To ensure that our customers get exactly what they need, we offer a variety of solutions that can be customized, combined, or deployed right out of the box and integrated with existing Enterprise applications.

1.2. Scope Of Work

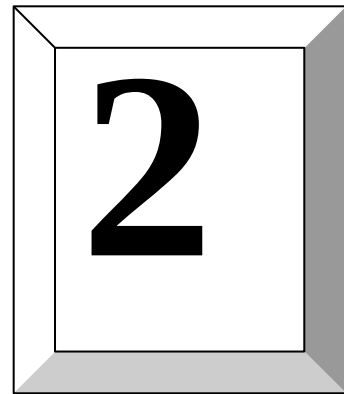
We put resources into development to enable our customers to release new potential over their associations. What makes La Net Team one of a kind is our pool of talented developers, industry skill, and a genuine comprehension of what to do so as to succeed. With an industry experience that traverses quite a few years, we offer a plenty of customer driven services by empowering undertakings to accomplish upper hand through adaptable and cutting edge worldwide delivery models. Our wide range of services include website designing, e-commerce services, Android and IOS Applications, Graphic and Logo design, Website Templates, Website Marketing, Product Development, Software Development etc.

1.3. Organization Chart



1.4. Capacity of Organization

Our company strength is about 250 employees and our company has two headquarters one in Surat.



Chapter 2: Introduction to Project

- 2.1 Introduction**
- 2.2 Purpose**
- 2.3 Scope**
- 2.4 Objective**
- 2.5 Technologies Used**

2.1 Introduction:

The Screenify project is a cutting-edge solution designed to provide a seamless and efficient streaming experience for users. The project is built using ReactJs and nodeJs, a powerful platform that allows developers to create dynamic and interactive user interfaces with ease. The project's database is implemented using MongoDB, a flexible and scalable NoSQL database that offers efficient data storage and retrieval. By combining NodeJs and MongoDB, the project provides fast and reliable performance, even with heavy traffic.

The website's user-friendly interface allows users to easily browse and stream movies and TV shows, create and manage their profiles, and add titles to their watchlist. In addition, administrators can use the admin portal to manage content, view user history, and track analytics.

Overall, this Screenify project is a robust and scalable solution that provides a seamless platform for users to stream their favorite movies and TV shows, and for businesses to reach out to their audience and streamline their operations.

2.2 Purpose:

The purpose of this Screenify is to provide a user-friendly platform for streaming movies, TV shows, and other video content online. By offering a wide range of content options, users can browse and watch their favorite shows and movies from the comfort of their own homes. The website is designed to be intuitive and easy to use, making it accessible to a wide range of users.

In addition, the Screenify allows users to personalize their viewing experience by creating profiles and curating a list of their favorite shows and movies. This feature enables users to easily find their preferred content and receive recommendations for new content based on their viewing history. The platform also provides users with access to exclusive content that may not be available on other streaming services.

The Screenify also offers businesses the opportunity to reach a wider audience through online streaming. By going online, businesses can tap into a global market and offer their content to users from all over the world. This increases the potential customer base and opens up new revenue streams for the company. Additionally, the platform allows businesses to track and analyze user behavior, such as viewing habits and preferences, to make informed decisions about content offerings and marketing strategies.

2.3 Scope:

2.3.1 User account creation is done by the user through the app or website.

2.3.2 User can login through email and password or through social media accounts.

2.3.3 User searches for TV shows or movies by typing in keywords or browsing through categories.

2.3.4 User filters search results by criteria such as genre, release year, rating, etc.

2.3.5 User adds TV shows or movies to their watchlist.

2.3.6 User selects a TV show or movie to watch and clicks the "Play" button.

2.3.7 The system verifies the user's subscription status and checks if the selected TV show or movie is available in the user's region.

2.3.8 If the user's subscription is invalid or if the selected TV show or movie is not available in the user's region, the system shows an error message and asks the user to subscribe or select a different TV show or movie.

2.3.9 If the subscription is valid and the selected TV show or movie is available, the system plays the selected TV show or movie.

2.4 Objective:

Screenify is an online streaming platform that allows users to watch their favorite movies and TV shows. The objectives of Screenify are as follows:

1. Provide a user-friendly platform: The main goal of Screenify is to provide a user-friendly platform that allows customers to easily search and watch their favorite movies and TV shows online. The website should be easy to navigate and provide a seamless streaming experience.

2. Improve brand visibility: Screenify should help movie and TV show producers increase their brand visibility by showcasing their content online and promoting their brand through digital marketing strategies. The site should include features such as user ratings and reviews, personalized recommendations, and social media integration to help producers build their brand identity.

3. Increase user engagement: Screenify should be designed to increase user engagement by providing users with efficient tools to manage their watchlist, provide feedback and ratings, and interact with other users through forums and comments sections. The site should also include features such as binge-watching mode and auto-play to encourage users to stay engaged.

4. Collect user data: Screenify should help producers collect user data such as viewing history, preferences, and search queries. This data can be used to make informed business decisions about content offerings, pricing, and marketing strategies.

5. Ensure data security: Screenify should ensure the security of user data and content by implementing appropriate security measures such as encryption, access control, and backups. The site should also comply with industry standards and regulations related to data privacy and security.

6. Optimize site performance: Screenify should optimize site performance by implementing caching, load balancing, and other techniques to ensure fast and reliable streaming. The site should also be optimized for different devices and internet speeds to ensure a consistent user experience across all platforms.

7. Streamline content management: Screenify should provide efficient tools for content management, such as easy uploading, metadata management, and content categorization. This will enable producers to manage their content efficiently and keep it organized for users to easily find and watch.

2.5 Technologies Used:

2.5.1 Design:

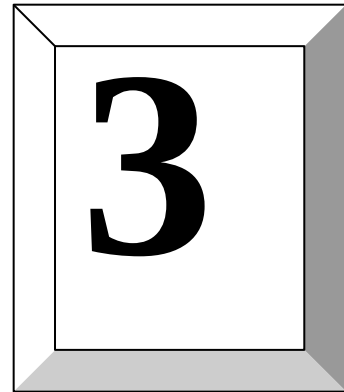
- JavaScript
- ReactJs

2.5.2 Logic:

- NodeJs
- React Redux

2.5.3 Database:

- MongoDb
- Firebase



Chapter 3: Learning Phase

- 3.1. Learning The Fundamentals
- 3.2. Learning about various Applications
- 3.3. JavaScript
- 3.4. NodeJs
- 3.5. ExpressJs
- 3.6. NestJs

3.1 Learning The Fundamentals

I began my internship having an orientation of the company. I went through the process of basic knowledge transfer on a continuous basis throughout my internship. I cleared many concepts in various programming language including JavaScript etc...

Right from setting up the environment to learning the fundamentals of any programming language & also working through databases etc..

3.2 Learning about various Applications

Web development involves the use of a variety of applications, software tools, and frameworks to design, develop, test, and deploy websites and web applications. Some of the most common applications used in web development include:

1. Code editors: Code editors, such as Visual Studio Code, Sublime Text, and Atom, are used to write and edit code in HTML, CSS, JavaScript, and other programming languages.
2. Integrated Development Environments (IDEs): IDEs, such as Eclipse, NetBeans, and JetBrains IntelliJ IDEA, are used to create and manage web development projects, as well as to write, debug, and deploy code.
3. Version control systems: Version control systems, such as Git, SVN, and Mercurial, are used to manage and track changes to code, collaborate with other developers, and revert changes if necessary.
4. Task runners: Task runners, such as Grunt, Gulp, and npm, automate repetitive tasks in the web development process, such as compiling code, optimizing images, and running tests.
5. Package managers: Package managers, such as npm, Yarn, and Bower, manage dependencies and libraries required for web development projects, making it easier to install and update software.
6. Design tools: Design tools, such as Adobe Photoshop, Sketch, and Figma, are used to create wireframes, mockups, and designs for websites and web applications.
7. Content Management Systems (CMS): CMS, such as WordPress, Drupal, and Joomla, provide an interface to manage the content and functionality of a website or web application.
8. Testing tools: Testing tools, such as Selenium, Jest, and Mocha, automate testing processes, ensuring that websites and web applications are bug-free and function as intended.
9. Web servers: Web servers, such as Apache, Nginx, and IIS, are used to host websites and web applications and deliver content to users.

10. Frameworks: Frameworks, such as React, Angular, Vue, Ruby on Rails, Django, and Laravel, provide pre-built tools and libraries to help developers create more efficient and scalable websites and web applications.

These applications and tools are essential for web developers to create modern, dynamic, and functional websites and web applications that meet the needs of businesses and individuals in the digital age.

3.3 JavaScript

1. Variables: Variables are used to store data values in JavaScript. They can be declared using the ``var``, ``let``, or ``const`` keywords.
2. Data types: JavaScript supports several data types, including strings, numbers, booleans, arrays, and objects.
3. Functions: Functions are reusable blocks of code that perform specific tasks. They can be defined using the ``function`` keyword and can be passed arguments and return values.
4. Conditionals: Conditionals, such as ``if`` statements and ``switch`` statements, are used to execute different blocks of code depending on whether a certain condition is true or false.
5. Loops: Loops, such as ``for`` loops and ``while`` loops, are used to repeat blocks of code multiple times.
6. Arrays: Arrays are used to store collections of data values in JavaScript. They can be accessed using index numbers and can be manipulated using various methods.
7. Objects: Objects are used to store collections of properties and values in JavaScript. They can be accessed using dot notation or bracket notation and can be manipulated using various methods.
8. Events: Events, such as mouse clicks and key presses, are used to trigger JavaScript code to execute in response to user actions.
9. DOM manipulation: The Document Object Model (DOM) is a programming interface that allows JavaScript code to interact with HTML and CSS elements on a webpage. JavaScript can be used to add, remove, and modify HTML elements dynamically.
10. Asynchronous programming: Asynchronous programming is used in JavaScript to execute code that does not block the main thread of execution. This is achieved using techniques such as callbacks, promises, and `async/await`.

These concepts are fundamental to JavaScript and are used extensively in web development to create dynamic and interactive websites and web applications.

3.4 NodeJs

Node.js is a server-side runtime environment for executing JavaScript code. It is built on the V8 JavaScript engine used by the Google Chrome browser and provides an event-driven, non-blocking I/O model that allows applications to handle a large number of connections simultaneously without blocking.

The key concepts of Node.js include:

1. **Asynchronous programming:** Node.js is designed to handle asynchronous I/O operations, meaning it can execute multiple tasks simultaneously without blocking other tasks. This allows Node.js applications to be highly scalable and performant.
2. **Modules:** Node.js has a built-in module system that allows developers to organize their code into reusable modules. Modules can be shared across applications, making it easy to develop and maintain complex applications.
3. **Package manager:** Node.js has a package manager called npm that provides access to a vast ecosystem of open-source modules and packages. Developers can easily install and manage dependencies for their projects using npm.
4. **Server-side programming:** Node.js is designed for server-side programming, allowing developers to build scalable and efficient server-side applications using JavaScript.
5. **Event-driven architecture:** Node.js uses an event-driven architecture that allows developers to respond to incoming requests and events in a non-blocking way. This allows Node.js applications to be highly responsive and efficient.

Overall, Node.js provides a powerful and flexible platform for building scalable and performant server-side applications using JavaScript. Its event-driven, non-blocking I/O model and vast ecosystem of modules and packages make it a popular choice for web developers around the world.

3.5 ExpressJs

Express is a popular open-source framework for building web applications using Node.js. It provides a simple and flexible API for building web servers and handling HTTP requests and responses.

Some key features of Express include:

1. **Routing:** Express provides a simple and flexible API for defining routes and handling HTTP requests. Developers can define routes based on HTTP methods (GET, POST, PUT, DELETE, etc.) and URL patterns.
2. **Middleware:** Express allows developers to use middleware functions to handle HTTP requests and responses. Middleware functions can be used to perform tasks such as authentication, logging, and error handling.
3. **Template engines:** Express provides support for various template engines, such as EJS and Handlebars, which allow developers to generate dynamic HTML pages.
4. **Static file serving:** Express allows developers to serve static files, such as images and CSS files, using the built-in static middleware.
5. **Error handling:** Express provides a robust error handling mechanism that allows developers to handle errors in a centralized way.
6. **Built-in support for HTTP cookies:** Express provides built-in support for handling HTTP cookies, making it easy to manage user sessions and authentication.

Overall, Express is a powerful and flexible framework for building web applications using Node.js. Its routing system, middleware support, template engines, static file serving, error handling, and built-in support for HTTP cookies make it a popular choice for building scalable and efficient web servers.

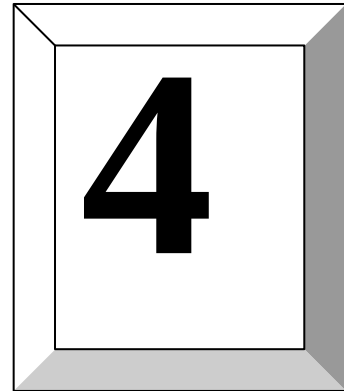
3.6 NestJs

Nest.js is a powerful, progressive Node.js framework for building scalable and efficient server-side applications. It is built on top of the popular Express.js framework and provides a robust set of features and architectural patterns to help developers build complex applications.

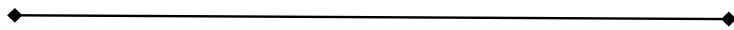
Some key features of Nest.js include:

1. **Modular architecture:** Nest.js uses a modular architecture that allows developers to organize their code into reusable modules. Modules can be shared across applications, making it easy to develop and maintain complex applications.
2. **Dependency injection:** Nest.js supports dependency injection, a design pattern that allows developers to write loosely coupled code. This makes it easier to test and maintain code and improves overall code quality.
3. **Built-in support for WebSockets:** Nest.js provides built-in support for WebSockets, making it easy to build real-time applications.
4. **Type safety:** Nest.js is written in TypeScript, a superset of JavaScript that adds type safety to the language. This makes it easier to write bug-free code and catch errors before they occur.
5. **Built-in support for microservices:** Nest.js provides built-in support for microservices, a popular architectural pattern for building distributed systems.

Overall, Nest.js is a powerful and flexible framework for building scalable and efficient server-side applications using Node.js. Its modular architecture, dependency injection, and built-in support for WebSockets and microservices make it a popular choice for developers around the world.



Chapter 4: System Design



4.1 Detail Design

4.1.1 Class Diagram

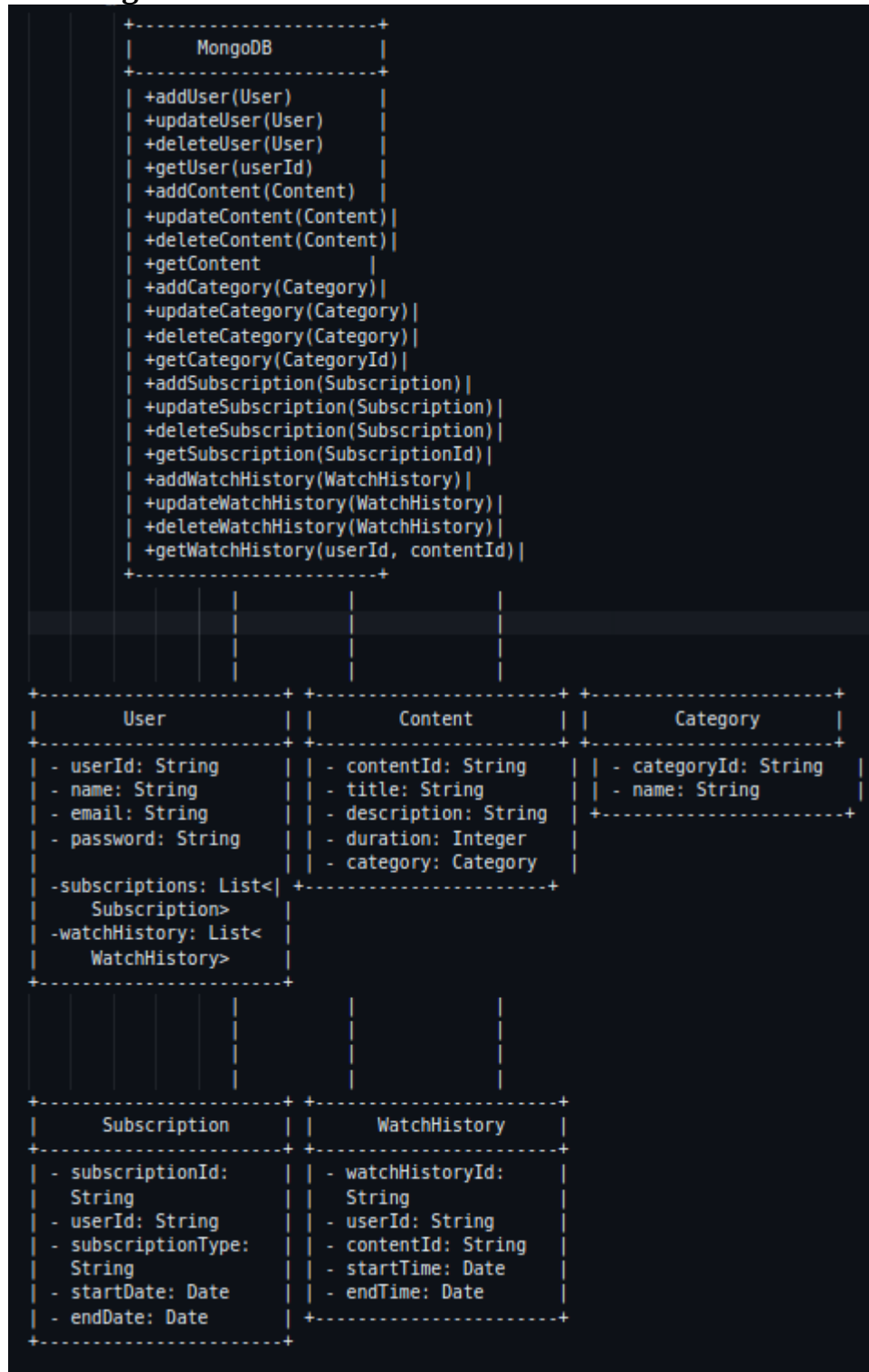
4.1.2 Activity Diagram

4.2 Database Design

4.3 Screens

4.1 Detail Design

4.1.1 Class Diagram



[Fig.4.1.1: Class Diagram]

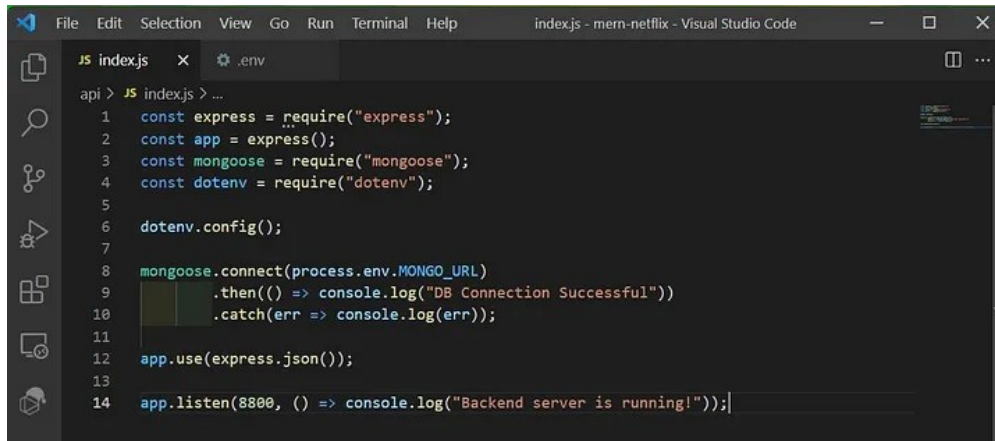
4.1.2 Activity Diagram



[Fig.4.1.2: Activity Diagram]

4.2 Database Design

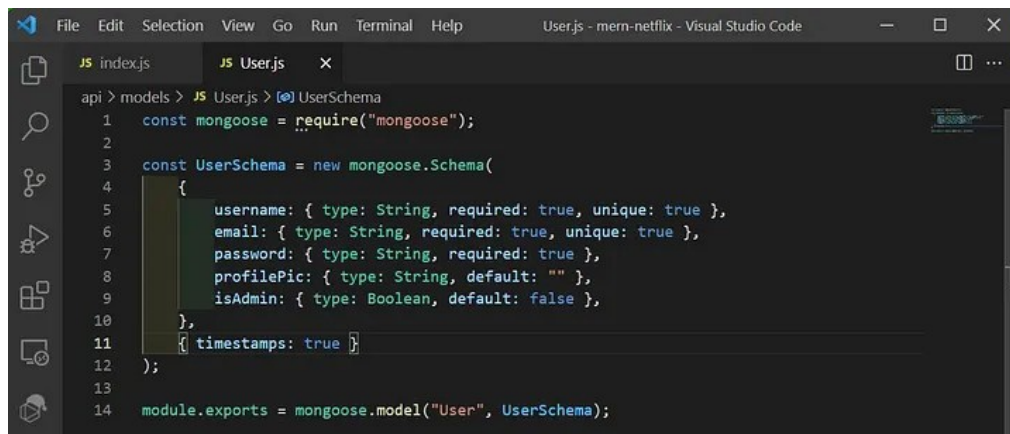
MongoDB Database



```

index.js - mern-netflix - Visual Studio Code
JS index.js x .env
api > JS index.js > ...
1  const express = require("express");
2  const app = express();
3  const mongoose = require("mongoose");
4  const dotenv = require("dotenv");
5
6  dotenv.config();
7
8  mongoose.connect(process.env.MONGO_URL)
9    .then(() => console.log("DB Connection Successful"))
10   .catch(err => console.log(err));
11
12 app.use(express.json());
13
14 app.listen(8800, () => console.log("Backend server is running!"));
  
```

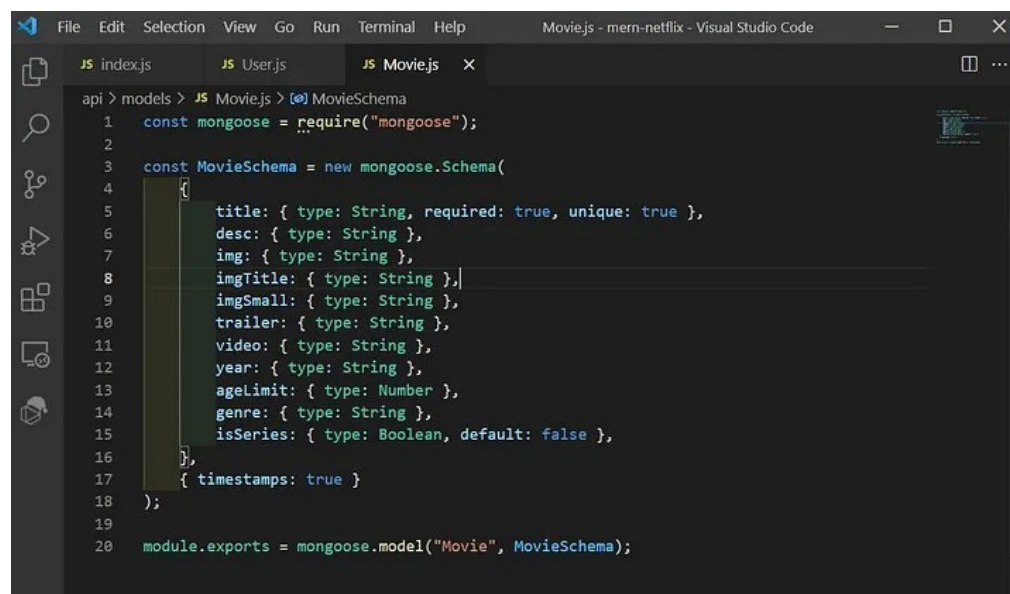
[Fig.4.2.1:MongoDB Connection]



```

User.js - mern-netflix - Visual Studio Code
JS index.js JS User.js x
api > models > JS User.js > UserSchema
1  const mongoose = require("mongoose");
2
3  const UserSchema = new mongoose.Schema(
4    {
5      username: { type: String, required: true, unique: true },
6      email: { type: String, required: true, unique: true },
7      password: { type: String, required: true },
8      profilePic: { type: String, default: "" },
9      isAdmin: { type: Boolean, default: false },
10     },
11     { timestamps: true }
12   );
13
14 module.exports = mongoose.model("User", UserSchema);
  
```

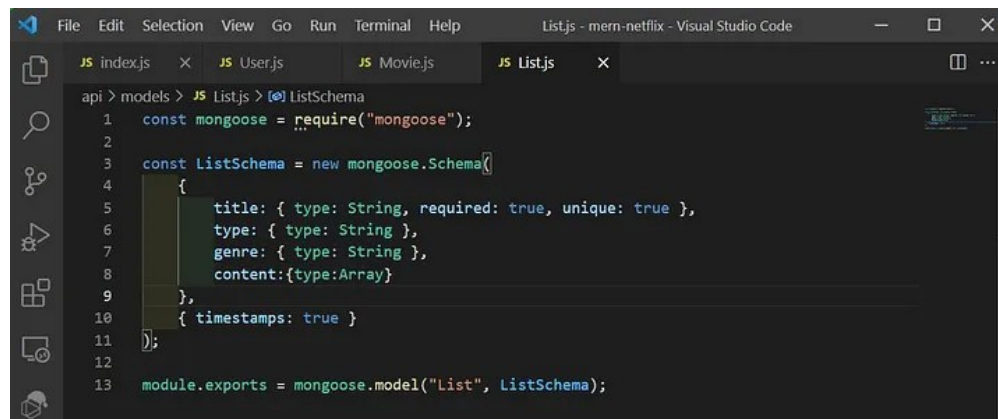
[Fig.4.2.2:Schema Design(User)]



```

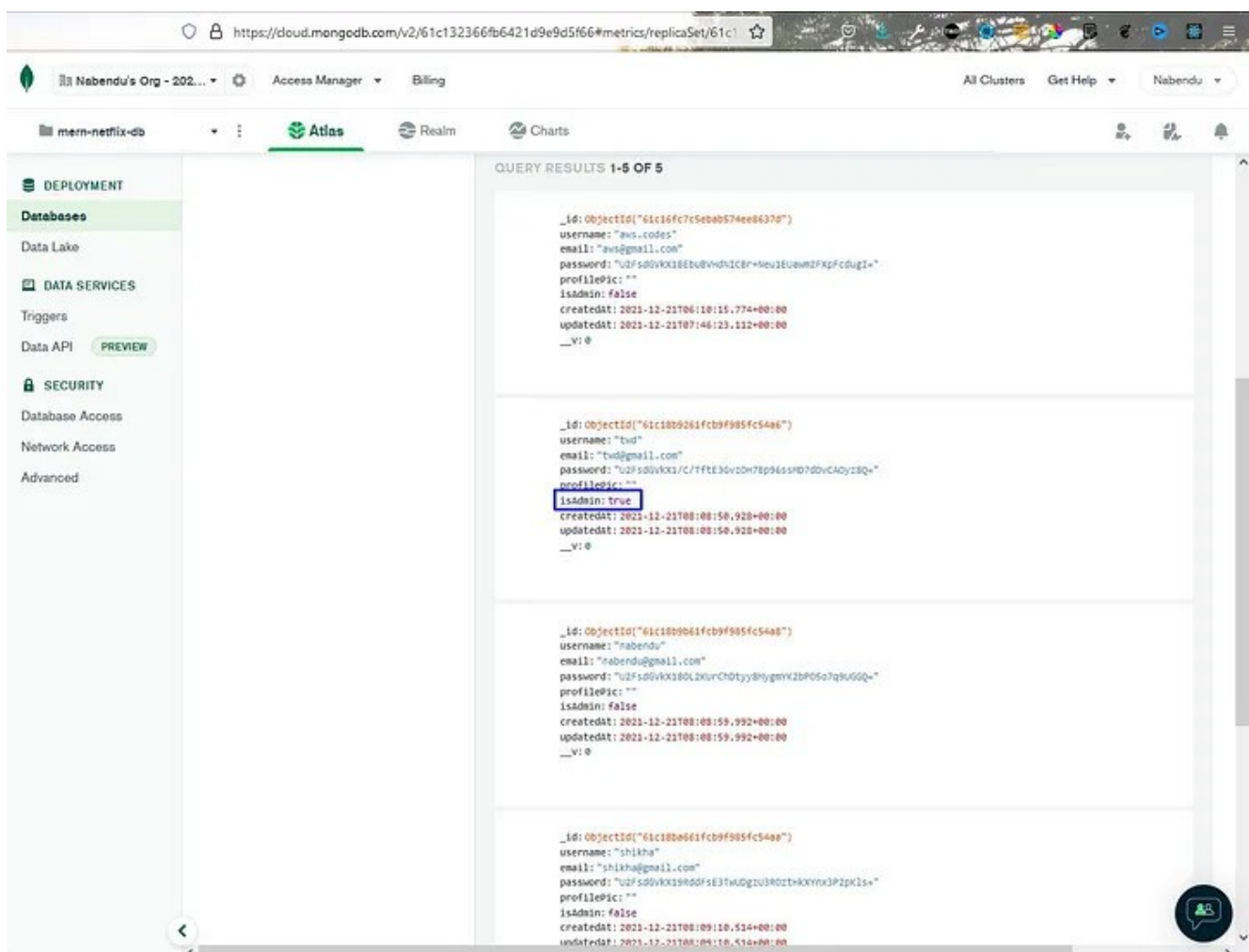
Movie.js - mern-netflix - Visual Studio Code
JS index.js JS User.js JS Movie.js x
api > models > JS Movie.js > MovieSchema
1  const mongoose = require("mongoose");
2
3  const MovieSchema = new mongoose.Schema(
4    {
5      title: { type: String, required: true, unique: true },
6      desc: { type: String },
7      img: { type: String },
8      imgTitle: { type: String },
9      imgSmall: { type: String },
10     trailer: { type: String },
11     video: { type: String },
12     year: { type: String },
13     ageLimit: { type: Number },
14     genre: { type: String },
15     isSeries: { type: Boolean, default: false },
16     },
17     { timestamps: true }
18   );
19
20 module.exports = mongoose.model("Movie", MovieSchema);
  
```

[Fig.4.2.3:Schema Design(Movie)]



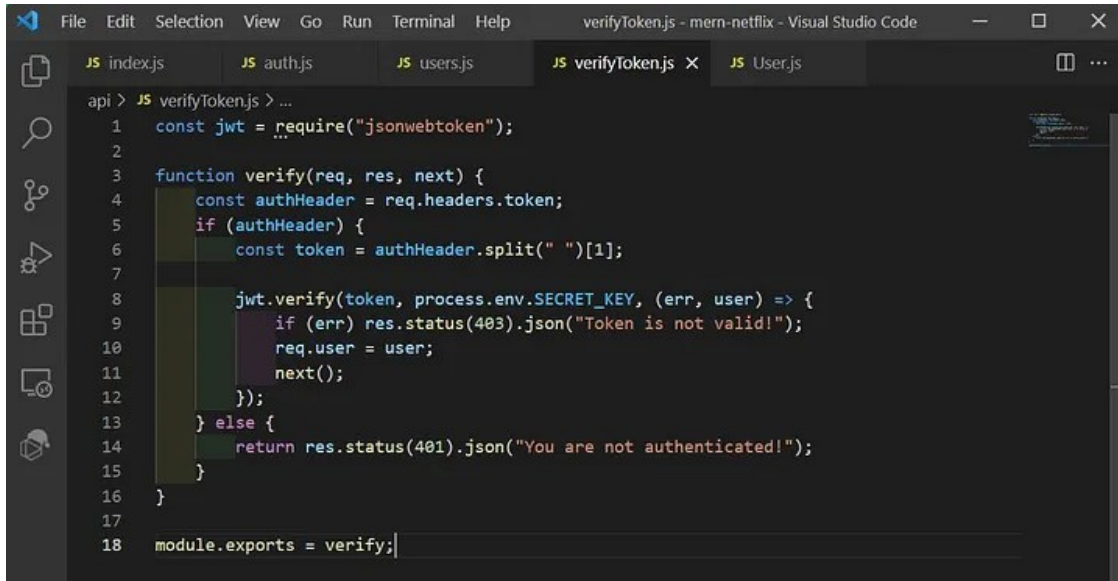
```
api > models > JS Listjs > ListSchema
1  const mongoose = require("mongoose");
2
3
4  const ListSchema = new mongoose.Schema({
5    {
6      title: { type: String, required: true, unique: true },
7      type: { type: String },
8      genre: { type: String },
9      content:{type:Array}
10   },
11   { timestamps: true }
12 });
13 module.exports = mongoose.model("List", ListSchema);
```

[Fig.4.2.4:Schema Design(List)]



[Fig.4.2.5:MongoDB Atlas]

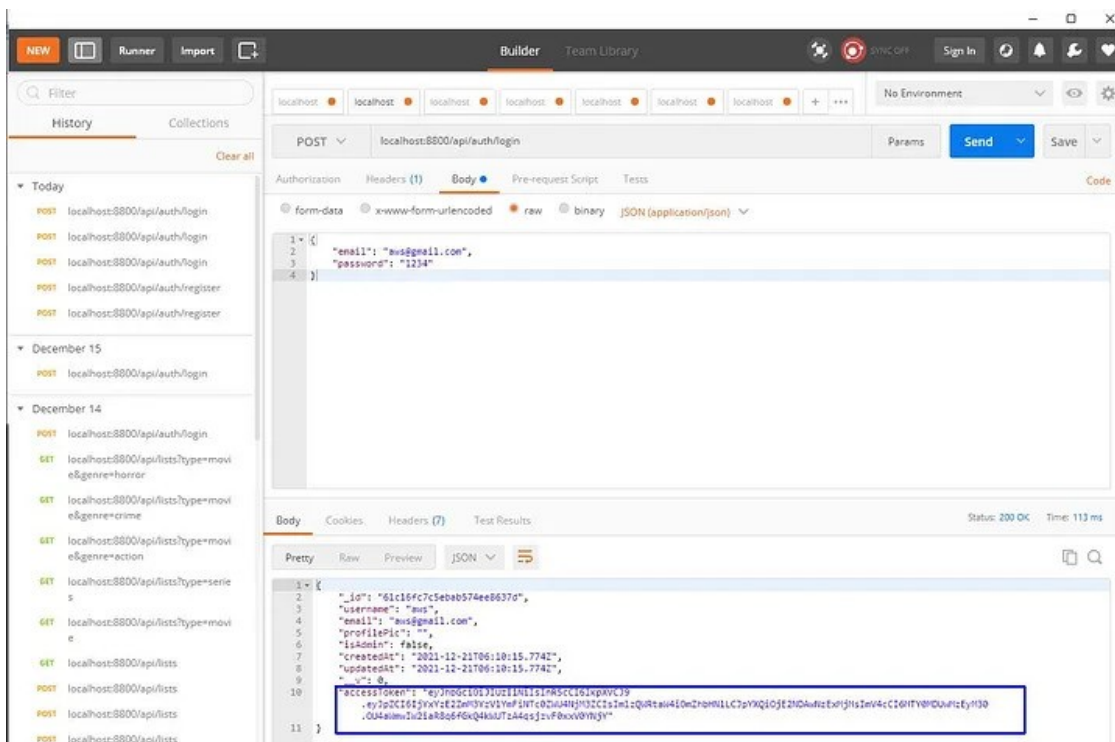
4.3 Authentication



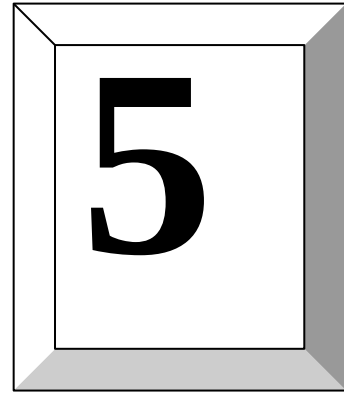
```

api > JS verifyToken.js > ...
1  const jwt = require("jsonwebtoken");
2
3  function verify(req, res, next) {
4    const authHeader = req.headers.token;
5    if (authHeader) {
6      const token = authHeader.split(" ")[1];
7
8      jwt.verify(token, process.env.SECRET_KEY, (err, user) => {
9        if (err) res.status(403).json("Token is not valid!");
10       req.user = user;
11       next();
12     });
13   } else {
14     return res.status(401).json("You are not authenticated!");
15   }
16 }
17
18 module.exports = verify;

```



[Fig.4.3.2: PostMan API]



Chapter 5: Implementation Planning

- 5.1 Implementation Environment
- 5.2 Program/Modules Specification
- 5.3 Coding Standards
- 5.4 Coding Scenario

5.1 Implementation Environment

The application is a single-server, multi-client application. Multiple users can log in to use the system.

Multi-user vs. single-user

A single-user web application is designed to serve only one user at a time, while a multi-user web application is designed to be used by multiple users simultaneously. In a single-user web application, the user has exclusive access to the system and can perform any operation in the application without worrying about data conflicts. In a multi-user web application, on the other hand, multiple users can access and use the system simultaneously. The application must handle concurrent access and manage any data conflicts that may arise. Multi-user web applications typically require more complex security and user management features to ensure that the system remains secure and that users can access only the data they are authorized to use.

GUI vs. non-GUI

GUI (Graphical User Interface) and non-GUI are two different approaches to building web applications. GUI-based web applications have a user interface that uses visual elements such as buttons, menus, forms, and images to allow users to interact with the application. Non-GUI web applications, also known as command-line applications, have no graphical interface and rely on text-based input and output.

GUI-based web applications are more user-friendly and intuitive, making it easier for users to navigate and interact with the application. Non-GUI web applications are generally simpler and more streamlined, making them easier to develop and maintain. However, they may not be as accessible to users who are unfamiliar with command-line interfaces.

5.2 Program/Modules Specification

- Web app development made with NodeJs
- User Module: - User .
- Admin Module:- Admin can track, change status, view and block user.

5.3 Coding Standards

Coding techniques incorporate many facts about software development. Although they usually have no impact on the functionality of the application; they contribute to an improved comprehension of source code. All forms of source code are considered here, including programming, scripting markup, and query languages.

Purpose of Coding Standards and Best Practices

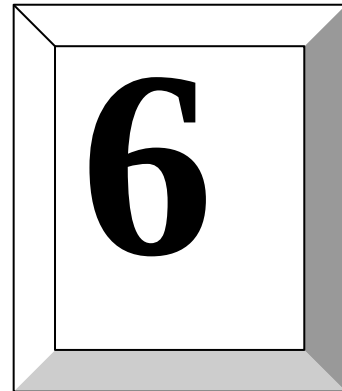
To develop reliable and maintainable applications, you must follow coding standards and best practices. The naming conventions, coding standards and best practices described in this document are compiled from our own experience and by referring to various guidelines. There are several standards that exist in the programming industry. None of them are wrong or bad and you may follow any of them. What is more important is, selecting one standard approach and ensuring that everyone is following it.

In this phase of software development, the design is related to a system converted into a machine-readable code that can be compiled and executed. Although the coding phase does not affect the structure of the system, it has a great impact on the internal structure of the module, which affects the testability, under the stability of the system.

5.4 Coding Scenario

I used NodeJS in this web app. This way I made each component reusable and reduced the footprint so that each functionality can be reused when needed.

Also, using Firebase helped make the website secure by storing and authenticating each individual user when they log in and make a purchase.



Chapter 6: Testing



6.1 Testing Plan

6.2 Test Strategy

6.3 Testing Methods

6.3.1 Test Cases

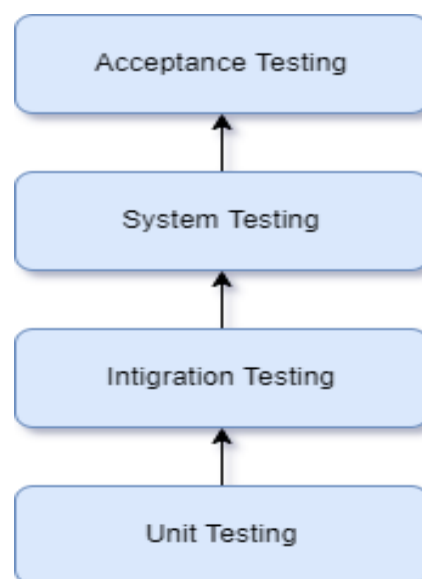
6.1 Testing Plan

The objective of the system testing is to ensure that all individual programs are working as expected, that the programs link together to meet the requirements specified and ensure that the computer system and the associated clerical and other procedures work together. Systems are not designed as entire systems but they are tested as single systems. The analyst must perform both unit and system testing.

Different types of testing methods are available. We have tested our system for different aspects like Does the system meet the goals for which it has been designed? This was a very important question that stood before us as the system was designed to be implemented on such a large network.

To fulfill its goal of being able to run on different systems we went through a series of tests at different places where this is supposed to be used the most. As we need to make our system efficient enough, we need to test it thoroughly.

Finally, we tested the system with real-time data, for which it is actually designed. We are successful in satisfying our needs as it was designed according to client's requirements. But it is very necessary to maintain this system and so our work is not still over.



6.2 Testing Strategy

Once source code has been generated, the software must be tested to uncover as many errors as possible before delivery to the customer. Our goal is to design a series of test cases that have a high likelihood of finding errors. Software testing techniques provide systematic guidance for designing tests that (1) Exercise the internal logic of software components (2) Exercise the inputs and outputs domains of the program to uncover errors in program function, behaviour and performance.

During the early stages of testing, a software engineer performs all tests. However, as the testing process progresses, testing specialists may become involved. Reviews and other activities can and do uncover errors, but they are not sufficient. Every time the program is executed, the customer tests it! Therefore, you have to execute the program before it gets to the customer with the specific intent of finding and removing all errors. In order to find the highest possible number of errors, tests must be conducted systematically and test cases must be designed using disciplined techniques.

6.2.1 Testing Objectives

Testing is a process of executing a program with the intention of finding an error.

A good test case is one that has a high probability of finding an as-yet undiscovered error.

A successful test is one that uncovers an as-yet undiscovered error.

6.2.2 Unit Testing

Unit testing is a software development process in which the smallest testable part of an application, called units, are individually scrutinized for proper operation. Unit testing is often automated but it can also be done manually. This testing mode is a component of Extreme Programming (XP), a pragmatic method of software development that takes a meticulous approach to building a product by means of continual testing and revision.

Unit testing involves only those characteristics that are vital to the performance of the unit under test.

This encourages developers to modify the source code without immediate concerns about how such changes might affect the functioning of the units or the program as a whole. Once all of the units in a program have been found to be working in the most efficient and error free manner possible, larger components of the program can be evaluated by means of integration testing.

6.2.3 System Testing

Now, it's time for whole System testing. We have found some cosmetic bugs and minor bugs. We have fixed it and tested it again. We worked on each error and exception that we got while testing and most of them are resolved or handled programmatically.

6.2.4 Recovery Testing

It is a system test that forces the software to fail in a variety of ways and verifies that recovery is properly performed.

6.2.5 Performance Testing

It is designed to test the run-time performance of software within the context of an integrated system performance testing occurs throughout all steps in the testing process.

6.3 Testing Methods

Acceptance Testing

Acceptance testing can be connected by the end user, customer, or client to validate whether or not to accept the product. Acceptance testing may be performed as part of the hand-off process between any two phases of development. The acceptance test suite is run against the supplied input data or using an acceptance test script to direct the tester. Then the results obtained are compared with the expected results. If there is a correct match for every case, the test suite is said to pass.

Alpha & beta testing

The alpha test is conducted at the developer's site by a customer. The software is used in a natural setting with the developer

“looking over shoulder” of the user and recording errors and usage problems.

Alpha test is conducted in a controlled environment. The beta testing is conducted at one or more customer sites by the end-user of the software. Unlike alpha testing, the developer is generally not present. Therefore, the beta test is a “live” application of the software in an environment that cannot be controlled by the developer.

Black-box testing

Also known as functional testing. Software testing techniques where by the internal working of the item being tested are not known by the tester. For example, in a black box test on software design the tester only knows the inputs and what the expected outcomes should be and not how the program arrives at those outputs. The tester does not ever examine the programming code and does not need any further knowledge of the program other than its specification.

The advantages of this type of testing include:

- The test is unbiased as the designer and the tester are independent of each other.
- The tester does not need knowledge of any specific programming languages.
- The test is done from the point of view of the user, not the designer. Test cases can be designed as soon as the specifications are complete.

The disadvantages of this type of testing include:

- The test can be redundant if the software designer has already run a test case.
- The test cases are difficult to design. Testing every possible input stream is unrealistic because it would take an inordinate amount of time: hence many program paths will go untested.

White Box Testing

Also known as glass box, structural, clear box and open box testing. A software testing technique where by explicit knowledge of the internal workings of the item being tested are used to select the test data. Unlike black box testing, white box testing uses specific knowledge of programming code to examine outputs. The test is accurate only if the tester knows what the program is supposed to do.

6.4 Test Cases

To minimize the number of errors in software, a rich variety of test design methods have evolved for software. These methods provide the developer with a systematic approach to testing. More importantly, methods provide a mechanism that can help to ensure the completeness of the test and provide the highest likelihood for uncovering errors in software.

An engineering product can be tested in one of the two ways:

Knowing the specified function that product has been designed to perform, tests can be conducted that demonstrate each function is fully operational while at the same time searching for errors in each function.

Knowing the internal workings of a product, tests can be conducted to ensure that “all gear mesh “, that is, internal oppression are performed according to specifications and all internal components have been adequately exercised. Here are the test cases that we had made for our application.

Test Case For Login

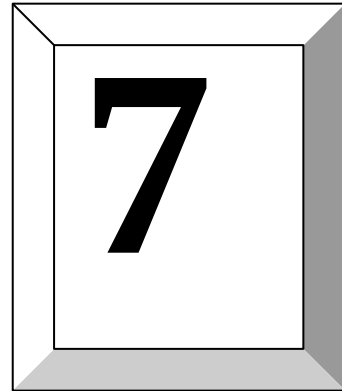
- Check the Email
- Check Password

Test Case For Sign Up

- Input username and email check if they are duplicate in database
- Input atleast 6 digit password.
- All this field are required.

Test Case Purchase

- Check if logged in and if product is available in inventory.
- Input address, state, city, pin code and mobile number correctly.
- Enter correct card number if purchasing through online payment.



Chapter 7: Limitations and Future Enhancement

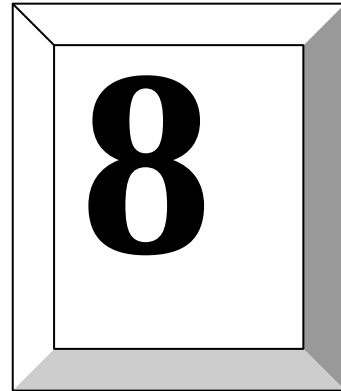
- ◆ —◆
- **Limitations**
- **Future Enhancement**

Limitations

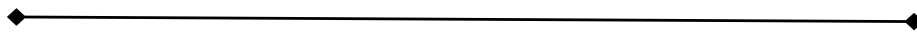
- **Subscription Required:** To access the content on the Netflix clone, users must subscribe to the service and pay a monthly or annual fee. The subscription can be managed through the app or website, but an internet connection is required to process payments.
- **Offline Downloads Available:** To overcome the issue of internet connectivity, the Netflix clone allows users to download movies and TV shows for offline viewing. However, the downloads can only be accessed through the app or website, and an internet connection is required to initiate the download process.
- **HD Quality Depends on Internet Speed:** The quality of the video on the Netflix clone depends on the speed and stability of the user's internet connection. Users with a slower internet connection may experience lower video quality or buffering while streaming high-definition content.

Future Enhancement

- **Virtual Reality Integration:** Adding virtual reality (VR) technology to the Netflix clone would allow users to experience movies and TV shows in a more immersive way. This would require users to have VR headsets, but it could provide a unique viewing experience and attract a new audience to the platform.
- **Interactive Content:** Interactive content, such as choose-your-own-adventure style shows, can provide users with a more personalized and engaging experience. By allowing users to make decisions and influence the outcome of the story, this type of content can increase user engagement and create a more memorable experience. Netflix has already experimented with interactive content in shows like "Black Mirror: Bandersnatch," and this trend could continue to grow in the future.



Chapter 8: Conclusion



8.1 Conclusion

In this app admin is main part of this application. in this app I learn lots of thing in this loan application. Finlay I complete this project and client are very satisfied for this loan application.

REFERENCES

- [1] Quora. (2009) For Solution of Question
[Online] [Accessed from February to April 2022]
<https://www.quora.com/>
- [2] Stack Overflow. (2008) Coding Related Question Solution
[Online] [Accessed from February to April 2022]
<https://stackoverflow.com/>
- [3] GitHub. (2007) GitHub Support Community
[Online] [Accessed from February to April
2022] <https://github.community>
- [4] Google Drive API. [~2012] To read, write, and sync files in Google
Drive. [Online] [Accessed from February to April 2022]
<https://developers.google.com/drive/api/v3/about-sdk>