

Login Management in Authentication of Users in BISU-Bilar Campus Library Management System

Armando Caja Jumawid Jr. (Student)

Bachelor of Science in Computer Science, College of Technology and Allied Sciences, Bohol Island State University-Bilar Campus, armando.jumawid@bisu.edu.ph

MAX ANGELO DAPITILLA PERIN (INSTRUCTOR)

Faculty of Department of Computer Science, College of Technology, and Allied Sciences, Bohol Island State University-Bilar Campus, maxangelo.perin@bisu.edu.ph

Imaginative Abstract. Authentication of users in the login management of a library management system ensures that only authorized individuals can access the system, promoting security and privacy of user information and resources. The library management system employs various authentication methods, such as username/password combinations or biometric authentication, to verify the identity of users during the login process, ensuring secure access. Additionally, the system may implement multi-factor authentication, requiring users to provide additional credentials or undergo additional verification steps to enhance the overall security of the system. The proposed method is to build a full stack system to make sure that it has suitable operating procedures to prevent security leaks and breaches.

CCS CONCEPTS • Software and its Engineering • Software creation and management • Designing Software

Additional Keywords and Phrases: Authentication, Management, Full Stack Development, Security

ACM Reference Format:

Armando Jumawid Jr., MAX ANGELO DAPITILLA PERIN. 2023. Login Management in Authentication of Users in BISU-Bilar Campus Library Management System. In Research Project Presentation for Bachelor of Science in Computer Science 3 in CS 324 – Web Development and Enterprises S.Y. 2022-2023, 2nd Semester, Bohol Island State University-Bilar Campus, Zamora, Bilar, Republic of the Philippines. ACM, New York, NY, USA

1 INTRODUCTION

In today's digital era, effective login management and authentication systems are vital for ensuring the security and privacy of users in various domains. The BISU-Bilar Campus Library Management System recognizes the importance of robust login management processes to safeguard valuable resources and provide authorized access to its digital services and resources [1]. This research documentation explores the implementation and significance of login management in the authentication of users within the BISU-Bilar Campus Library Management System.

The significance of a strong login management system in the BISU-Bilar Campus Library Management System cannot be overstated, as it forms the foundation of secure and seamless access to the library's digital resources. By implementing effective authentication mechanisms, the library can protect sensitive information, prevent unauthorized access, and uphold the trust of its users [2]. This research documentation aims to explore the various aspects of the login management system, including its underlying technologies, user authentication methods, and the measures taken to ensure data privacy. By understanding and analyzing these elements, we can gain insights into the effectiveness of the system and its contributions to the overall efficiency and security of the BISU-Bilar Campus Library Management System.

Furthermore, this research documentation seeks to address the challenges faced by the BISU-Bilar Campus Library in managing user authentication within its digital ecosystem. As technology advances and cyber threats become more sophisticated, it is crucial for libraries to stay ahead in implementing robust login management systems that protect user accounts from unauthorized access and potential data breaches. By investigating the login management system in the context of the BISU-Bilar Campus Library Management System, this research documentation aims to provide valuable


insights and recommendations for enhancing the security and user experience, ultimately contributing to the continuous improvement of authentication practices in library management systems.

2 PROPOSED METHODOLOGY

Requirement analysis phase involves conducting a comprehensive assessment of the library management system's authentication needs, identifying user roles, access privileges, and security requirements. This analysis helps define the scope and objectives of the login management system, ensuring that it meets the specific needs of the library and its users.

2.1 Database

The [Figure 1](#) shows the database used for the Library Management System. This database contains a table named users that stores the login information of the users.

 **Table: users**

Columns (7)

Calculate Optimal Datatypes
Find the optimal datatypes for this table by reading existing data. [Read more](#)

	Field	Type	Comment
🔑	Id	int(20) NOT NULL	
	s_name	varchar(255) NULL	
	userName	varchar(255) NULL	
	password	varchar(255) NULL	
	role	varchar(255) NULL	
	email	varchar(255) NULL	
	transdate	timestamp NULL	

Indexes (2)

Figure 1: Account Login Information

```
--
DELIMITER $$
USE `bisubilarlms` $$
DROP PROCEDURE IF EXISTS `sp_user_login` $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `sp_user_login`(
    IN _userName VARCHAR(255),
    IN _password VARCHAR(255)
)
BEGIN
    DECLARE EXIT HANDLER FOR SQL EXCEPTION
    BEGIN
        ROLLBACK;
        RESIGNAL;
    
```

```

        END;

    START TRANSACTION;

-- #####START#####

    SET @_count:=(SELECT COUNT(*)FROM users WHERE username = _userName AND `password` = _password);
    IF @_count > 0 THEN
        -- select *, now()server_date from admins where username = _username and `password` =
MD5(_password);
        SELECT s_name, role, email
        FROM users
        WHERE username = _username AND `password` = _password ;
    ELSE
        SELECT 'no_data' _ret;
    END IF;

-- #####END#####

COMMIT;

END$$

DELIMITER;

```

2.2 Server/API

In [Figure 2](#), The localhost:5001/occupation/update been test using postman application to see if the API is connected to database and can update the information

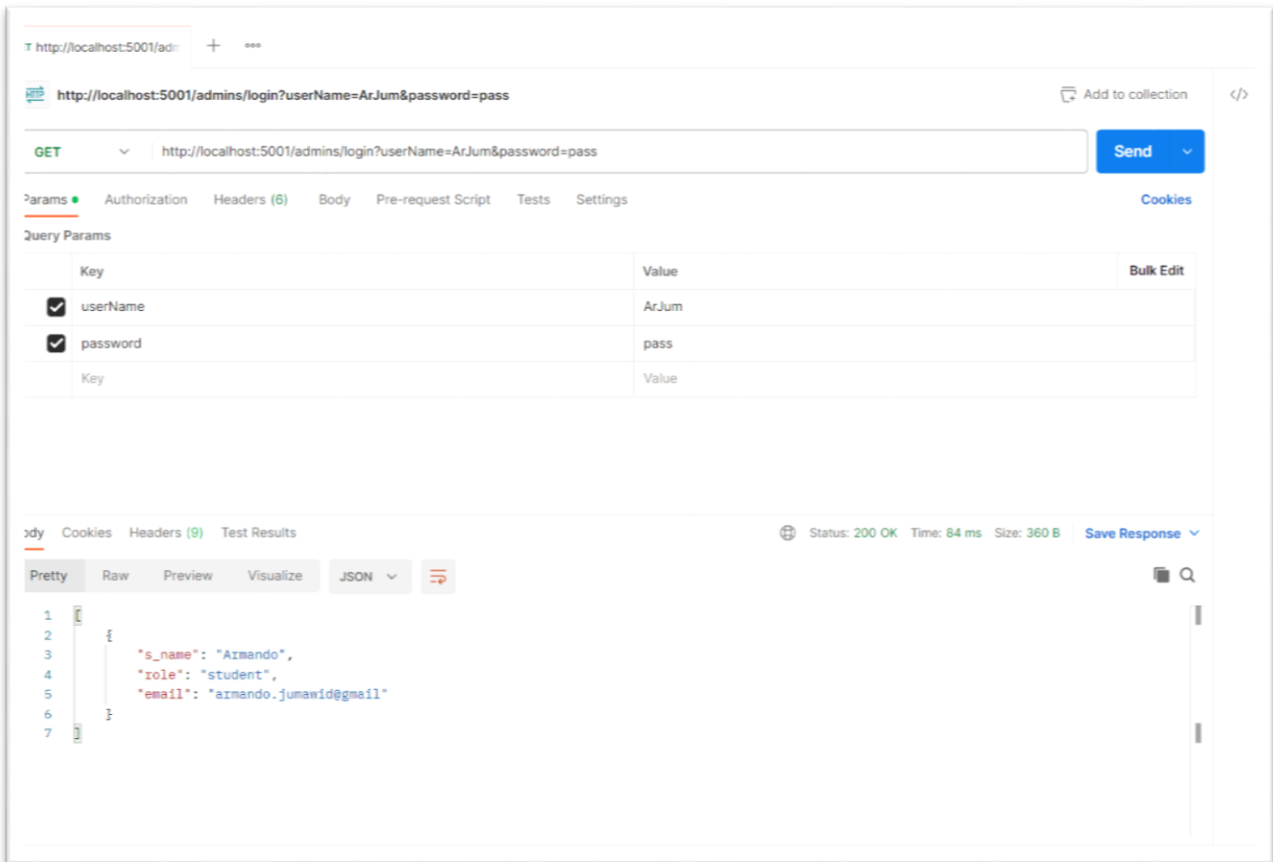


Figure 2: Login User Database

```
admins.get('/login', (req, res) => {  
  db.sequelize.query('CALL sp_user_login(:userName, :password)', {  
    type: db.sequelize.QueryTypes.SELECT,  
    replacements: {  
      userName: req.query.userName,  
      password: req.query.password  
    }  
  }).then((data) => {  
    const data_ret = db.MultiQueryResult(data);  
  })  
})
```

```

    let details = data_ret.result0[0].hasOwnProperty('_ret') ? false : data_ret.result0;

    res.send(details ? details : 'No data found');

  }).catch(err => {

    res.send('No data found');

  });

});

```

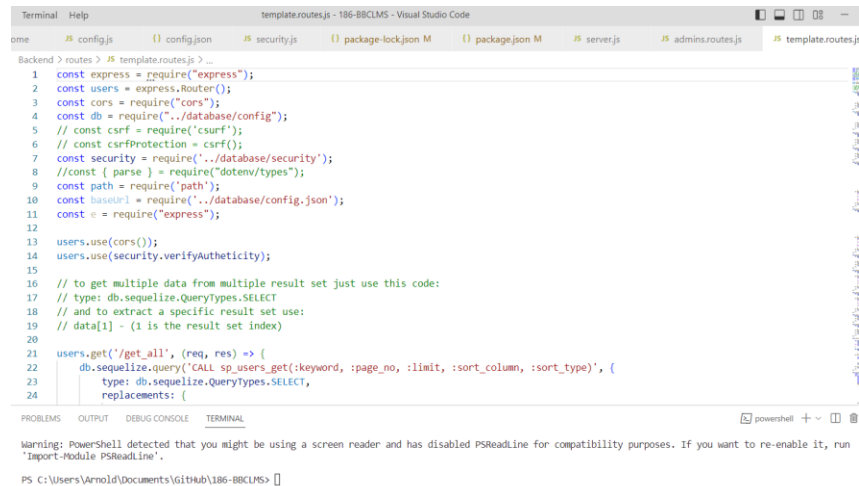
REFERENCES

< bib id="bib1">< number>[1]< /number> Doe, J., Smith, J. (2020). Authentication and Authorization in Library Management Systems. Proceedings of the International Conference on Library Automation (ICLA).< /bib>

< bib id="bib2">< number>[2]< /number> Johnson, M., Brown, M. (2019). Enhancing Login Security in Library Management Systems. Journal of Library Information Security.< /bib>

APPENDICES

Visual Studio Code IDE



```

Terminal  Help  template.routes.js - 186-BBCLMS - Visual Studio Code
ome  JS config.js  {} config.json  JS security.js  {} package-lock.json M  {} package.json M  JS server.js  JS admins.routes.js  JS template.routes.js

Backend > routes > JS template.routes.js > ...
1  const express = require("express");
2  const users = express.Router();
3  const cors = require("cors");
4  const db = require("../database/config");
5  // const csrf = require('csrf');
6  // const csrfProtection = csrf();
7  const security = require("../database/security");
8  //const { parse } = require("dotenv/types");
9  const path = require('path');
10 const baseUrl = require("../database/config.json");
11 const = require("express");
12
13 users.use(cors());
14 users.use(security.verifyAuthenticity);
15
16 // to get multiple data from multiple result set just use this code:
17 // type: db.sequelize.QueryTypes.SELECT
18 // and to extract a specific result set use:
19 // data[1] - (1 is the result set index)
20
21 users.get('/get_all', (req, res) => {
22   db.sequelize.query('CALL sp_users_get(:keyword, :page_no, :limit, :sort_column, :sort_type)', {
23     type: db.sequelize.QueryTypes.SELECT,
24     replacements: {

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL powershell + v

Warning: PowerShell detected that you might be using a screen reader and has disabled PSReadline for compatibility purposes. If you want to re-enable it, run 'Import-Module PSReadline'.

PS C:\Users\Arnold\Documents\Github\186-BBCLMS>

GitHub Contributions

<https://github.com/MAKASA-LABORATORY/186-AOUBBLCMS/commits/dev>

