

**END-OF-SEMESTER EXAM REPORT**  
**PROJECT LIBRARY INFORMATION SYSTEM:**  
**VOCATIONAL LIBRARY (VOKS-BRARY)**



**Compiled by the Group 4:**

- |                         |                   |
|-------------------------|-------------------|
| 1. Najwa Firdaus Akiyah | (233140700111063) |
| 2. Candra Wahyu Perdana | (233140701111043) |

**CLASS: T2A**

**MAJOR: INFORMATION TECHNOLOGY**  
**FACULTY OF VOCATIONAL**  
**BRAWIJAYA UNIVERSITY**

**Academic Year 2024**

**Even Semester 2023/2024**

## **PREFACE**

Praise be to Allah SWT for His mercy, guidance, and grace to all of us so that we can complete this report with the title 'Final Semester Exam Report (UAS) Library Information System Project: Vocational Library (VOKS-Brary)". This report was prepared as one of the requirements for the Final Semester project in the Web Programming course taught by Mr M. Sonhaji Akbar, S.Pd., M.Kom. In preparing this report, we maximise our abilities, so it is hoped that suggestions or input from readers so that better developments can be made and can be useful for various parties including us, students.

Malang, 7 June 2024

The Compiler

# TABLE OF CONTENTS

## COVER PAGE

PREFACE.....	i
--------------	---

TABLE OF CONTENTS.....	ii
------------------------	----

BAB I. INTRODUCTION.....	1
--------------------------	---

1.1 The Background.....	1
-------------------------	---

1.2 Problem Statement.....	2
----------------------------	---

1.3 The Purposes.....	2
-----------------------	---

1.4 The Benefits.....	2
-----------------------	---

BAB II. DISCUSSMENT.....	3
--------------------------	---

2.1 The Code.....	3
-------------------	---

2.2 Output: Website.....	12
--------------------------	----

BAB III. CLOSURE.....	16
-----------------------	----

3.1 The Conclusion.....	16
-------------------------	----

3.2 The Advice.....	16
---------------------	----

LITERATURE LIST.....	17
----------------------	----

# **BAB I**

## **INTRODUCTION**

### **1.1 The Background**

An information system is an integrated set of components to collect, store, and process data and provide information, knowledge, and digital products. In this report, a library information system is created at the Faculty of Vocational Studies, Universitas Brawijaya, called Vocational Library Information System (Voks-Brary). This information system is used to be accessed by users (students, lecturers, staff, etc.) to view book catalogues, borrow and return books.

In making this library information system using the required materials, namely HTML, CSS, PHP, bootstrap, and database using MySQL. HTML (Hypertext Markup Language) is a markup language for creating attractive and interconnected web page structures that can be accessed via the internet. The functions of using HTML include creating web pages that are usually paired with CSS and Javascript; as hyperlinks to direct moving to other pages that have conditions in the text; as the main foundation for making websites because several programming languages can be integrated in HTML; as text markers in website sections such as navigation, main, header, and footer sections.

CSS (Cascading Style Sheet) is a style sheet language used to determine the appearance and format of HTML-based web pages or other markup languages, such as setting font types, writing colours, and backgrounds. The functions of using CSS include offering many variations related to the appearance of the website, making the website look neat and optimal on various screen sizes (desktop, smartphone), making it easier to manage code because just edit the code in the CSS file then the changes will be applied to all pages of the website, and speeding up the process of loading website pages (loading) because writing one set of code for several website pages at once makes the amount of code can be minimised.

PHP (Hypertext Preprocessor) is an open-source scripting language widely used in web programming that runs on the server side. A scripting language is a language that automatically executes tasks in a special runtime environment including instructing static pages created with HTML and CSS. The function of PHP is to create dynamic web pages that can adjust the display of content depending on the situation such as storing data into a database, creating pages that change according to input,

processing forms, etc.

### **1.2 Problem Statement**

1. How to create a library information system connected to a database that can facilitate users?
2. How can the Library Information System: Vocational Library (VOKS-Brary) can facilitate users in accessing book-related information?

### **1.3 The Purposes**

1. To create a library information system connected to a database that can facilitate users.
2. To find out the functionality and usefulness of the Library Information System: Vocational Library (VOKS-Brary) that can facilitate users in accessing book-related information.

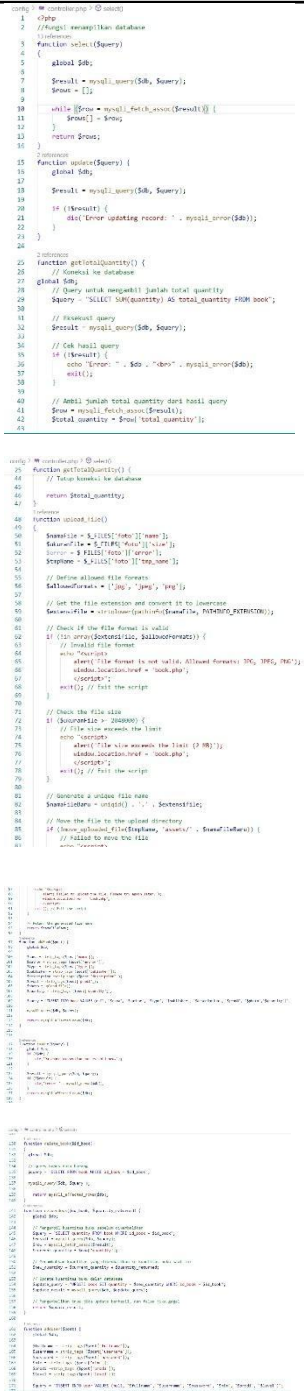
### **1.4 The Benefits**

1. Provide convenience for users in accessing book-related information in the catalogue feature.
2. Providing online services that can be accessed 24/7 anywhere, anytime, and by anyone.
3. Add insight into the process of creating a library information system that is connected to a database.
4. Providing online borrowing and returning services by utilising information technology.

## BAB II DISCUSSMENT

### 2.1 The Code

The following is the code created for the Vocational Library website:

No.	Image	File	Description
1	 <pre> 1 &lt;?php 2 //Koneksi ke database 3 function koneksi(\$server, \$username, \$password, \$host, \$dbname) { 4     \$koneksi = mysqli_connect(\$server, \$username, \$password, \$host, \$dbname); 5     if (!\$koneksi) { 6         die("Koneksi gagal: " . mysqli_error(\$koneksi)); 7     } 8     return \$koneksi; 9 } 10 11 // Fungsi untuk menampilkan data 12 function tampil_data(\$koneksi, \$table, \$id) { 13     \$query = "SELECT * FROM \$table WHERE id = \$id"; 14     \$result = mysqli_query(\$koneksi, \$query); 15     if (!\$result) { 16         die("Gagal menampilkan data: " . mysqli_error(\$koneksi)); 17     } 18     return \$result; 19 } 20 21 // Fungsi untuk menambahkan data 22 function tambah_data(\$koneksi, \$table, \$data) { 23     \$query = "INSERT INTO \$table (\$field1, \$field2, \$field3) VALUES (\$value1, \$value2, \$value3)"; 24     \$result = mysqli_query(\$koneksi, \$query); 25     if (!\$result) { 26         die("Gagal menambahkan data: " . mysqli_error(\$koneksi)); 27     } 28     return \$result; 29 } 30 31 // Fungsi untuk mengupdate data 32 function update_data(\$koneksi, \$table, \$id, \$data) { 33     \$query = "UPDATE \$table SET \$field1 = \$value1, \$field2 = \$value2 WHERE id = \$id"; 34     \$result = mysqli_query(\$koneksi, \$query); 35     if (!\$result) { 36         die("Gagal mengupdate data: " . mysqli_error(\$koneksi)); 37     } 38     return \$result; 39 } 40 41 // Fungsi untuk menghapus data 42 function hapus_data(\$koneksi, \$table, \$id) { 43     \$query = "DELETE FROM \$table WHERE id = \$id"; 44     \$result = mysqli_query(\$koneksi, \$query); 45     if (!\$result) { 46         die("Gagal menghapus data: " . mysqli_error(\$koneksi)); 47     } 48     return \$result; 49 } 50 51 // Fungsi untuk menampilkan daftar data 52 function daftar_data(\$koneksi, \$table) { 53     \$query = "SELECT * FROM \$table"; 54     \$result = mysqli_query(\$koneksi, \$query); 55     if (!\$result) { 56         die("Gagal menampilkan daftar data: " . mysqli_error(\$koneksi)); 57     } 58     return \$result; 59 } 60 61 // Fungsi untuk login 62 function login(\$koneksi, \$username, \$password) { 63     \$query = "SELECT * FROM users WHERE username = '\$username'"; 64     \$result = mysqli_query(\$koneksi, \$query); 65     if (!\$result) { 66         die("Gagal login: " . mysqli_error(\$koneksi)); 67     } 68     if (mysqli_num_rows(\$result) &gt; 0) { 69         \$user = mysqli_fetch_assoc(\$result); 70         if (\$password == \$user['password']) { 71             return \$user; 72         } 73     } 74     return false; 75 } 76 77 // Fungsi untuk logout 78 function logout(\$koneksi) { 79     \$query = "UPDATE users SET session_id = '' WHERE session_id = '\$session_id'"; 80     \$result = mysqli_query(\$koneksi, \$query); 81     if (!\$result) { 82         die("Gagal logout: " . mysqli_error(\$koneksi)); 83     } 84     return \$result; 85 } 86 87 // Fungsi untuk register 88 function register(\$koneksi, \$username, \$password) { 89     \$query = "INSERT INTO users (username, password) VALUES ('\$username', '\$password')"; 90     \$result = mysqli_query(\$koneksi, \$query); 91     if (!\$result) { 92         die("Gagal register: " . mysqli_error(\$koneksi)); 93     } 94     return \$result; 95 } 96 97 // Fungsi untuk reset password 98 function reset_password(\$koneksi, \$username, \$password) { 99     \$query = "UPDATE users SET password = '\$password' WHERE username = '\$username'"; 100     \$result = mysqli_query(\$koneksi, \$query); 101     if (!\$result) { 102         die("Gagal reset password: " . mysqli_error(\$koneksi)); 103     } 104     return \$result; 105 } 106 107 // Fungsi untuk forgot password 108 function forgot_password(\$koneksi, \$username) { 109     \$query = "SELECT * FROM users WHERE username = '\$username'"; 110     \$result = mysqli_query(\$koneksi, \$query); 111     if (!\$result) { 112         die("Gagal forgot password: " . mysqli_error(\$koneksi)); 113     } 114     if (mysqli_num_rows(\$result) &gt; 0) { 115         \$user = mysqli_fetch_assoc(\$result); 116         return \$user['password']; 117     } 118     return false; 119 } 120 121 // Fungsi untuk forgot password 122 function forgot_password(\$koneksi, \$username) { 123     \$query = "SELECT * FROM users WHERE username = '\$username'"; 124     \$result = mysqli_query(\$koneksi, \$query); 125     if (!\$result) { 126         die("Gagal forgot password: " . mysqli_error(\$koneksi)); 127     } 128     if (mysqli_num_rows(\$result) &gt; 0) { 129         \$user = mysqli_fetch_assoc(\$result); 130         return \$user['password']; 131     } 132     return false; 133 } 120 121 // Fungsi untuk forgot password 122 function forgot_password(\$koneksi, \$username) { 123     \$query = "SELECT * FROM users WHERE username = '\$username'"; 124     \$result = mysqli_query(\$koneksi, \$query); 125     if (!\$result) { 126         die("Gagal forgot password: " . mysqli_error(\$koneksi)); 127     } 128     if (mysqli_num_rows(\$result) &gt; 0) { 129         \$user = mysqli_fetch_assoc(\$result); 130         return \$user['password']; 131     } 132     return false; 133 } 120 121 // Fungsi untuk forgot password 122 function forgot_password(\$koneksi, \$username) { 123     \$query = "SELECT * FROM users WHERE username = '\$username'"; 124     \$result = mysqli_query(\$koneksi, \$query); 125     if (!\$result) { 126         die("Gagal forgot password: " . mysqli_error(\$koneksi)); 127     } 128     if (mysqli_num_rows(\$result) &gt; 0) { 129         \$user = mysqli_fetch_assoc(\$result); 130         return \$user['password']; 131     } 132     return false; 133 } 120 121 // Fungsi untuk forgot password 122 function forgot_password(\$koneksi, \$username) { 123     \$query = "SELECT * FROM users WHERE username = '\$username'"; 124     \$result = mysqli_query(\$koneksi, \$query); 125     if (!\$result) { 126         die("Gagal forgot password: " . mysqli_error(\$koneksi)); 127     } 128     if (mysqli_num_rows(\$result) &gt; 0) { 129         \$user = mysqli_fetch_assoc(\$result); 130         return \$user['password']; 131     } 132     return false; 133 } 120 121 // Fungsi untuk forgot password 122 function forgot_password(\$koneksi, \$username) { 123     \$query = "SELECT * FROM users WHERE username = '\$username'"; 124     \$result = mysqli_query(\$koneksi, \$query); 125     if (!\$result) { 126         die("Gagal forgot password: " . mysqli_error(\$koneksi)); 127     } 128     if (mysqli_num_rows(\$result) &gt; 0) { 129         \$user = mysqli_fetch_assoc(\$result); 130         return \$user['password']; 131     } 132     return false; 133 } 120 121 // Fungsi untuk forgot password 122 function forgot_password(\$koneksi, \$username) { 123     \$query = "SELECT * FROM users WHERE username = '\$username'"; 124     \$result = mysqli_query(\$koneksi, \$query); 125     if (!\$result) { 126         die("Gagal forgot password: " . mysqli_error(\$koneksi)); 127     } 128     if (mysqli_num_rows(\$result) &gt; 0) { 129         \$user = mysqli_fetch_assoc(\$result); 130         return \$user['password']; 131     } 132     return false; 133 } 120 121 // Fungsi untuk forgot password 122 function forgot_password(\$koneksi, \$username) { 123     \$query = "SELECT * FROM users WHERE username = '\$username'"; 124     \$result = mysqli_query(\$koneksi, \$query); 125     if (!\$result) { 126         die("Gagal forgot password: " . mysqli_error(\$koneksi)); 127     } 128     if (mysqli_num_rows(\$result) &gt; 0) { 129         \$user = mysqli_fetch_assoc(\$result); 130         return \$user['password']; 131     } 132     return false; 133 } 120 121 // Fungsi untuk forgot password 122 function forgot_password(\$koneksi, \$username) { 123     \$query = "SELECT * FROM users WHERE username = '\$username'"; 124     \$result = mysqli_query(\$koneksi, \$query); 125     if (!\$result) { 126         die("Gagal forgot password: " . mysqli_error(\$koneksi)); 127     } 128     if (mysqli_num_rows(\$result) &gt; 0) { 129         \$user = mysqli_fetch_assoc(\$result); 130         return \$user['password']; 131     } 132     return false; 133 } 120 121 // Fungsi untuk forgot password 122 function forgot_password(\$koneksi, \$username) { 123     \$query = "SELECT * FROM users WHERE username = '\$username'"; 124     \$result = mysqli_query(\$koneksi, \$query); 125     if (!\$result) { 126         die("Gagal forgot password: " . mysqli_error(\$koneksi)); 127     } 128     if (mysqli_num_rows(\$result) &gt; 0) { 129         \$user = mysqli_fetch_assoc(\$result); 130         return \$user['password']; 131     } 132     return false; 133 } 120 121 // Fungsi untuk forgot password 122 function forgot_password(\$koneksi, \$username) { 123     \$query = "SELECT * FROM users WHERE username = '\$username'"; 124     \$result = mysqli_query(\$koneksi, \$query); 125     if (!\$result) { 126         die("Gagal forgot password: " . mysqli_error(\$koneksi)); 127     } 128     if (mysqli_num_rows(\$result) &gt; 0) { 129         \$user = mysqli_fetch_assoc(\$result); 130         return \$user['password']; 131     } 132     return false; 133 } 120 121 // Fungsi untuk forgot password 122 function forgot_password(\$koneksi, \$username) { 123     \$query = "SELECT * FROM users WHERE username = '\$username'"; 124     \$result = mysqli_query(\$koneksi, \$query); 125     if (!\$result) { 126         die("Gagal forgot password: " . mysqli_error(\$koneksi)); 127     } 128     if (mysqli_num_rows(\$result) &gt; 0) { 129         \$user = mysqli_fetch_assoc(\$result); 130         return \$user['password']; 131     } 132     return false; 133 } 120 121 // Fungsi untuk forgot password 122 function forgot_password(\$koneksi, \$username) { 123     \$query = "SELECT * FROM users WHERE username = '\$username'"; 124     \$result = mysqli_query(\$koneksi, \$query); 125     if (!\$result) { 126         die("Gagal forgot password: " . mysqli_error(\$koneksi)); 127     } 128     if (mysqli_num_rows(\$result) &gt; 0) { 129         \$user = mysqli_fetch_assoc(\$result); 130         return \$user['password']; 131     } 132     return false; 133 } 120 121 // Fungsi untuk forgot password 122 function forgot_password(\$koneksi, \$username) { 123     \$query = "SELECT * FROM users WHERE username = '\$username'"; 124     \$result = mysqli_query(\$koneksi, \$query); 125     if (!\$result) { 126         die("Gagal forgot password: " . mysqli_error(\$koneksi)); 127     } 128     if (mysqli_num_rows(\$result) &gt; 0) { 129         \$user = mysqli_fetch_assoc(\$result); 130         return \$user['password']; 131     } 132     return false; 133 } 120 121 // Fungsi untuk forgot password 122 function forgot_password(\$koneksi, \$username) { 123     \$query = "SELECT * FROM users WHERE username = '\$username'"; 124     \$result = mysqli_query(\$koneksi, \$query); 125     if (!\$result) { 126         die("Gagal forgot password: " . mysqli_error(\$koneksi)); 127     } 128     if (mysqli_num_rows(\$result) &gt; 0) { 129         \$user = mysqli_fetch_assoc(\$result); 130         return \$user['password']; 131     } 132     return false; 133 } 120 121 // Fungsi untuk forgot password 122 function forgot_password(\$koneksi, \$username) { 123     \$query = "SELECT * FROM users WHERE username = '\$username'"; 124     \$result = mysqli_query(\$koneksi, \$query); 125     if (!\$result) { 126         die("Gagal forgot password: " . mysqli_error(\$koneksi)); 127     } 128     if (mysqli_num_rows(\$result) &gt; 0) { 129         \$user = mysqli_fetch_assoc(\$result); 130         return \$user['password']; 131     } 132     return false; 133 } 120 121 // Fungsi untuk forgot password 122 function forgot_password(\$koneksi, \$username) { 123     \$query = "SELECT * FROM users WHERE username = '\$username'"; 124     \$result = mysqli_query(\$koneksi, \$query); 125     if (!\$result) { 126         die("Gagal forgot password: " . mysqli_error(\$koneksi)); 133 </pre>	controller.php inside the config folder	<p>The code serves to display the database, connect to the database that has been created by displaying tables and modify data through the CRUD (Create, Read, Update, Delete) method.</p> <p>Additional operations performed on this code upload files through a form returned with the uploaded file name and book management (retrieve the total quantity of books, add books, input book data, delete books, return books) as well as adding and deleting users in the form of students and operators, admin cannot be deleted.</p>

<pre> 161 // koneksi ke database 162 function koneksi(\$host,\$user,\$pass,\$db) { 163     \$conn = mysqli_connect(\$host,\$user,\$pass,\$db); 164     if (!\$conn) { 165         die("Koneksi gagal: " . mysqli_connect_error()); 166     } 167     return \$conn; 168 } 169 170 // query hapus data barang 171 \$query = "DELETE FROM user WHERE id_user = \$id_user"; 172 173 mysqli_query(\$db, \$query); 174 175 return mysqli_affected_rows(\$db); 176 } 177 178 // koneksi ke database 179 function koneksi(\$host,\$user,\$pass,\$db) { 180     \$conn = mysqli_connect(\$host,\$user,\$pass,\$db); 181     if (!\$conn) { 182         die("Koneksi gagal: " . mysqli_connect_error()); 183     } 184     return \$conn; 185 } 186 187 // query hapus data barang 188 \$query = "DELETE FROM user WHERE id_user = \$id_user"; 189 190 mysqli_query(\$db, \$query); 191 192 return mysqli_affected_rows(\$db); 193 } </pre>		
<p>2</p> <pre> 1 // koneksi ke database 2 function koneksi(\$host,\$user,\$pass,\$db) { 3     \$conn = mysqli_connect(\$host,\$user,\$pass,\$db); 4     if (!\$conn) { 5         die("Koneksi gagal: " . mysqli_connect_error()); 6     } 7     return \$conn; 8 } 9 10 // query hapus data barang 11 \$query = "DELETE FROM user WHERE id_user = \$id_user"; 12 13 mysqli_query(\$db, \$query); 14 15 return mysqli_affected_rows(\$db); 16 } </pre>	<p>database.php</p>	<p>The code is created to function as a link or connection to the MySQL database that has been created called fixlibrary.sql.</p> <p>If it fails to connect, an error notification will appear stating that it has not connected to the MySQL database that has been created.</p>
<p>3</p> <pre> 1 // koneksi ke database 2 function koneksi(\$host,\$user,\$pass,\$db) { 3     \$conn = mysqli_connect(\$host,\$user,\$pass,\$db); 4     if (!\$conn) { 5         die("Koneksi gagal: " . mysqli_connect_error()); 6     } 7     return \$conn; 8 } 9 10 // query hapus data barang 11 \$query = "DELETE FROM user WHERE id_user = \$id_user"; 12 13 mysqli_query(\$db, \$query); 14 15 return mysqli_affected_rows(\$db); 16 } </pre>	<p>index.php</p>	<p>This code serves as the main page of the accessed website.</p> <p>Displaying a dynamic website related to library information, books, and book borrowing based on users: students, admins, and operators.</p> <p>The use of JavaScript for interactive features in the search books column by typing the book title or author. The down arrow leads to scrolling in the book list section.</p> <p>There is a session start at the very top that requires users to</p>





			<p>Session start at the very top for connection to the fixlibrary database.</p> <p>The login process with the post method that fills in the data according to the database table and if the login is successful it will be directed back to the index.php page.</p> <p>Display an error notification if the login fails.</p>
6	<pre> 1  &lt;?php 2  session_start(); 3 4  if (!isset(\$_SESSION["login"])) { 5      echo "&lt;script&gt; 6      alert('Login first please!'); 7      document.location.href = 'index.php'; 8      &lt;/script&gt;"; 9      exit; 10 } 11 //kosongkan \$_SESSION user login 12 \$_SESSION = []; 13 14 session_unset(); 15 session_destroy(); 16 header("location: index.php"); 17 ?&gt; </pre>	<p>logout.php</p> <p>nsid the loginperpus folder</p>	<p>The code functions to access the logout of the user account (student, admin, operator). Using session start and if not logged in, will be directed to log in first. If you have successfully logged in, use session unset and session destroy and return to the login page.</p>

```

1  *
2  *
3  *
4  *
5  *
6  *
7  *
8  *
9  *
10 *
11 *
12 *
13 *
14 *
15 *
16 *
17 *
18 *
19 *
20 *
21 *
22 *
23 *
24 *
25 *
26 *
27 *
28 *
29 *
30 *
31 *
32 *
33 *
34 *
35 *
36 *
37 *
38 *
39 *
40 *
41 *
42 *
43 *
44 *
45 *
46 *
47 *
48 *
49 *
50 *
51 *
52 *
53 *
54 *
55 *
56 *
57 *
58 *
59 *
60 *
61 *
62 *
63 *
64 *
65 *
66 *
67 *
68 *
69 *
70 *
71 *
72 *
73 *
74 *
75 *
76 *
77 *
78 *
79 *
80 *
81 *
82 *
83 *
84 *
85 *
86 *
87 *
88 *
89 *
90 *
91 *
92 *
93 *
94 *
95 *
96 *
97 *
98 *
99 *
100 *
101 *
102 *
103 *
104 *
105 *
106 *
107 *
108 *
109 *
110 *
111 *
112 *
113 *
114 *
115 *
116 *
117 *
118 *
119 *
120 *
121 *
122 *
123 *
124 *
125 *
126 *
127 *
128 *
129 *
130 *
131 *
132 *
133 *
134 *
135 *
136 *
137 *
138 *
139 *
140 *
141 *
142 *
143 *
144 *
145 *
146 *
147 *
148 *
149 *
150 *
151 *
152 *
153 *
154 *
155 *
156 *
157 *
158 *
159 *
160 *
161 *
162 *
163 *
164 *
165 *
166 *
167 *
168 *
169 *
170 *
171 *
172 *
173 *
174 *
175 *
176 *
177 *
178 *
179 *
180 *
181 *
182 *
183 *
184 *
185 *
186 *
187 *
188 *
189 *
190 *
191 *
192 *
193 *
194 *
195 *
196 *
197 *
198 *
199 *
200 *
201 *
202 *
203 *
204 *
205 *
206 *
207 *
208 *
209 *
210 *
211 *
212 *
213 *
214 *
215 *
216 *
217 *
218 *
219 *
220 *
221 *
222 *
223 *
224 *
225 *
226 *
227 *
228 *
229 *
230 *
231 *
232 *
233 *
234 *
235 *
236 *
237 *
238 *
239 *
240 *
241 *
242 *
243 *
244 *
245 *
246 *
247 *
248 *
249 *
250 *
251 *
252 *
253 *
254 *
255 *
256 *
257 *
258 *
259 *
260 *
261 *
262 *
263 *
264 *
265 *
266 *
267 *
268 *
269 *
270 *
271 *
272 *
273 *
274 *
275 *
276 *
277 *
278 *
279 *
280 *
281 *
282 *
283 *
284 *
285 *
286 *
287 *
288 *
289 *
290 *
291 *
292 *
293 *
294 *
295 *
296 *
297 *
298 *
299 *
300 *
301 *
302 *
303 *
304 *
305 *
306 *
307 *
308 *
309 *
310 *
311 *
312 *
313 *
314 *
315 *
316 *
317 *
318 *
319 *
320 *
321 *
322 *
323 *
324 *
325 *
326 *
327 *
328 *
329 *
330 *
331 *
332 *
333 *
334 *
335 *
336 *
337 *
338 *
339 *
340 *
341 *
342 *
343 *
344 *
345 *
346 *
347 *
348 *
349 *
350 *
351 *
352 *
353 *
354 *
355 *
356 *
357 *
358 *
359 *
360 *
361 *
362 *
363 *
364 *
365 *
366 *
367 *
368 *
369 *
370 *
371 *
372 *
373 *
374 *
375 *
376 *
377 *
378 *
379 *
380 *
381 *
382 *
383 *
384 *
385 *
386 *
387 *
388 *
389 *
390 *
391 *
392 *
393 *
394 *
395 *
396 *
397 *
398 *
399 *
400 *
401 *
402 *
403 *
404 *
405 *
406 *
407 *
408 *
409 *
410 *
411 *
412 *
413 *
414 *
415 *
416 *
417 *
418 *
419 *
420 *
421 *
422 *
423 *
424 *
425 *
426 *
427 *
428 *
429 *
430 *
431 *
432 *
433 *
434 *
435 *
436 *
437 *
438 *
439 *
440 *
441 *
442 *
443 *
444 *
445 *
446 *
447 *
448 *
449 *
450 *
451 *
452 *
453 *
454 *
455 *
456 *
457 *
458 *
459 *
460 *
461 *
462 *
463 *
464 *
465 *
466 *
467 *
468 *
469 *
470 *
471 *
472 *
473 *
474 *
475 *
476 *
477 *
478 *
479 *
480 *
481 *
482 *
483 *
484 *
485 *
486 *
487 *
488 *
489 *
490 *
491 *
492 *
493 *
494 *
495 *
496 *
497 *
498 *
499 *
500 *
501 *
502 *
503 *
504 *
505 *
506 *
507 *
508 *
509 *
510 *
511 *
512 *
513 *
514 *
515 *
516 *
517 *
518 *
519 *
520 *
521 *
522 *
523 *
524 *
525 *
526 *
527 *
528 *
529 *
530 *
531 *
532 *
533 *
534 *
535 *
536 *
537 *
538 *
539 *
540 *
541 *
542 *
543 *
544 *
545 *
546 *
547 *
548 *
549 *
550 *
551 *
552 *
553 *
554 *
555 *
556 *
557 *
558 *
559 *
560 *
561 *
562 *
563 *
564 *
565 *
566 *
567 *
568 *
569 *
570 *
571 *
572 *
573 *
574 *
575 *
576 *
577 *
578 *
579 *
580 *
581 *
582 *
583 *
584 *
585 *
586 *
587 *
588 *
589 *
590 *
591 *
592 *
593 *
594 *
595 *
596 *
597 *
598 *
599 *
600 *
601 *
602 *
603 *
604 *
605 *
606 *
607 *
608 *
609 *
610 *
611 *
612 *
613 *
614 *
615 *
616 *
617 *
618 *
619 *
620 *
621 *
622 *
623 *
624 *
625 *
626 *
627 *
628 *
629 *
630 *
631 *
632 *
633 *
634 *
635 *
636 *
637 *
638 *
639 *
640 *
641 *
642 *
643 *
644 *
645 *
646 *
647 *
648 *
649 *
650 *
651 *
652 *
653 *
654 *
655 *
656 *
657 *
658 *
659 *
660 *
661 *
662 *
663 *
664 *
665 *
666 *
667 *
668 *
669 *
670 *
671 *
672 *
673 *
674 *
675 *
676 *
677 *
678 *
679 *
680 *
681 *
682 *
683 *
684 *
685 *
686 *
687 *
688 *
689 *
690 *
691 *
692 *
693 *
694 *
695 *
696 *
697 *
698 *
699 *
700 *
701 *
702 *
703 *
704 *
705 *
706 *
707 *
708 *
709 *
710 *
711 *
712 *
713 *
714 *
715 *
716 *
717 *
718 *
719 *
720 *
721 *
722 *
723 *
724 *
725 *
726 *
727 *
728 *
729 *
730 *
731 *
732 *
733 *
734 *
735 *
736 *
737 *
738 *
739 *
740 *
741 *
742 *
743 *
744 *
745 *
746 *
747 *
748 *
749 *
750 *
751 *
752 *
753 *
754 *
755 *
756 *
757 *
758 *
759 *
760 *
761 *
762 *
763 *
764 *
765 *
766 *
767 *
768 *
769 *
770 *
771 *
772 *
773 *
774 *
775 *
776 *
777 *
778 *
779 *
780 *
781 *
782 *
783 *
784 *
785 *
786 *
787 *
788 *
789 *
790 *
791 *
792 *
793 *
794 *
795 *
796 *
797 *
798 *
799 *
800 *
801 *
802 *
803 *
804 *
805 *
806 *
807 *
808 *
809 *
810 *
811 *
812 *
813 *
814 *
815 *
816 *
817 *
818 *
819 *
820 *
821 *
822 *
823 *
824 *
825 *
826 *
827 *
828 *
829 *
830 *
831 *
832 *
833 *
834 *
835 *
836 *
837 *
838 *
839 *
840 *
841 *
842 *
843 *
844 *
845 *
846 *
847 *
848 *
849 *
850 *
851 *
852 *
853 *
854 *
855 *
856 *
857 *
858 *
859 *
860 *
861 *
862 *
863 *
864 *
865 *
866 *
867 *
868 *
869 *
870 *
871 *
872 *
873 *
874 *
875 *
876 *
877 *
878 *
879 *
880 *
881 *
882 *
883 *
884 *
885 *
886 *
887 *
888 *
889 *
890 *
891 *
892 *
893 *
894 *
895 *
896 *
897 *
898 *
899 *
900 *
901 *
902 *
903 *
904 *
905 *
906 *
907 *
908 *
909 *
910 *
911 *
912 *
913 *
914 *
915 *
916 *
917 *
918 *
919 *
920 *
921 *
922 *
923 *
924 *
925 *
926 *
927 *
928 *
929 *
930 *
931 *
932 *
933 *
934 *
935 *
936 *
937 *
938 *
939 *
940 *
941 *
942 *
943 *
944 *
945 *
946 *
947 *
948 *
949 *
950 *
951 *
952 *
953 *
954 *
955 *
956 *
957 *
958 *
959 *
960 *
961 *
962 *
963 *
964 *
965 *
966 *
967 *
968 *
969 *
970 *
971 *
972 *
973 *
974 *
975 *
976 *
977 *
978 *
979 *
980 *
981 *
982 *
983 *
984 *
985 *
986 *
987 *
988 *
989 *
990 *
991 *
992 *
993 *
994 *
995 *
996 *
997 *
998 *
999 *
1000 *

```

style.css inside  
the  
loginperpus  
folder

The code is to make the  
appearance of the website in  
the login section more  
attractive by selecting  
colours, fonts, spacing, etc.

[illegible]

9	<pre> 1 // style.css 2 3 /* Reset */ 4 * { 5     margin: 0; 6     padding: 0; 7 } 8 9 /* Body */ 10 body { 11     font-family: sans-serif; 12     font-size: 100%; 13     color: #333; 14 } 15 16 /* Header */ 17 header { 18     background-color: #f0f0f0; 19     padding: 10px 0; 20 } 21 22 /* Main Content */ 23 main { 24     padding: 20px 0; 25 } 26 27 /* Footer */ 28 footer { 29     background-color: #f0f0f0; 30     padding: 10px 0; 31 } 32 33 /* Links */ 34 a { 35     color: #007bff; 36     text-decoration: none; 37 } 38 39 /* Buttons */ 40 button { 41     background-color: #007bff; 42     color: white; 43     padding: 5px 10px; 44     border: none; 45     cursor: pointer; 46 } 47 48 /* Form Elements */ 49 input { 50     border: 1px solid #ccc; 51     padding: 5px; 52 } 53 54 /* Tables */ 55 table { 56     width: 100%; 57     border-collapse: collapse; 58 } 59 60 table tr { 61     background-color: #f2f2f2; 62 } 63 64 table td, table th { 65     padding: 10px; 66 } 67 68 /* Media Queries */ 69 @media (max-width: 768px) { 70     .container { 71         padding: 0 15px; 72     } 73 } 74 75 </pre>	style.css for index.php file	<p>The code is partially displayed on the side to make the web interface on the index.php page attractive according to the selection of colours, fonts, etc.</p> <p>The code is not displayed in full, can be seen more fully in the code file.</p>
10	<pre> 1 // style1.css 2 3 /* Reset */ 4 * { 5     margin: 0; 6     padding: 0; 7 } 8 9 /* Body */ 10 body { 11     font-family: sans-serif; 12     font-size: 100%; 13     color: #333; 14 } 15 16 /* Header */ 17 header { 18     background-color: #f0f0f0; 19     padding: 10px 0; 20 } 21 22 /* Main Content */ 23 main { 24     padding: 20px 0; 25 } 26 27 /* Footer */ 28 footer { 29     background-color: #f0f0f0; 30     padding: 10px 0; 31 } 32 33 /* Links */ 34 a { 35     color: #007bff; 36     text-decoration: none; 37 } 38 39 /* Buttons */ 40 button { 41     background-color: #007bff; 42     color: white; 43     padding: 5px 10px; 44     border: none; 45     cursor: pointer; 46 } 47 48 /* Form Elements */ 49 input { 50     border: 1px solid #ccc; 51     padding: 5px; 52 } 53 54 /* Tables */ 55 table { 56     width: 100%; 57     border-collapse: collapse; 58 } 59 60 table tr { 61     background-color: #f2f2f2; 62 } 63 64 table td, table th { 65     padding: 10px; 66 } 67 68 /* Media Queries */ 69 @media (max-width: 768px) { 70     .container { 71         padding: 0 15px; 72     } 73 } 74 75 </pre>	style1.css	<p>The code partially displayed at the top serves to make the display on the detailbuku.php web page more attractive with the selection of certain colours, fonts, etc..</p> <p>The code is not displayed in full, can be seen more fully in the code file.</p>

```

1  <script>
2  // =====
3  // @author: https://github.com/0x09cr4ck
4  // @version: 1.0.0
5  // @description: A simple script to demonstrate a basic security concept.
6  // @license: MIT
7  // =====
8
9  // Function to generate a random string
10 function getRandomString() {
11     let characters = 'abcdefghijklmnopqrstuvwxyz0123456789';
12     let randomString = '';
13     for (let i = 0; i < 10; i++) {
14         randomString += characters.charAt(Math.floor(Math.random() * characters.length));
15     }
16     return randomString;
17 }
18
19 // Function to generate a random number
20 function getRandomNumber() {
21     return Math.floor(Math.random() * 1000000000);
22 }
23
24 // Function to generate a random date
25 function getRandomDate() {
26     let date = new Date();
27     let year = date.getFullYear();
28     let month = date.getMonth();
29     let day = date.getDate();
30     return `${year}-${month}-${day}`;
31 }
32
33 // Function to generate a random token
34 function getRandomToken() {
35     let token = getRandomString() + '-' + getRandomNumber() + '-' + getRandomDate();
36     return token;
37 }
38
39 // Function to generate a random user ID
40 function getRandomUserID() {
41     let userID = getRandomString() + '-' + getRandomNumber();
42     return userID;
43 }
44
45 // Function to generate a random password
46 function getRandomPassword() {
47     let password = getRandomString() + '-' + getRandomNumber() + '-' + getRandomDate();
48     return password;
49 }
50
51 // Function to generate a random email
52 function getRandomEmail() {
53     let email = getRandomString() + '@example.com';
54     return email;
55 }
56
57 // Function to generate a random phone number
58 function getRandomPhoneNumber() {
59     let phoneNumber = getRandomString() + '-' + getRandomNumber();
60     return phoneNumber;
61 }
62
63 // Function to generate a random address
64 function getRandomAddress() {
65     let address = getRandomString() + '-' + getRandomNumber() + '-' + getRandomDate();
66     return address;
67 }
68
69 // Function to generate a random name
70 function getRandomName() {
71     let name = getRandomString() + '-' + getRandomNumber();
72     return name;
73 }
74
75 // Function to generate a random age
76 function getRandomAge() {
77     let age = Math.floor(Math.random() * 100) + 1;
78     return age;
79 }
80
81 // Function to generate a random gender
82 function getRandomGender() {
83     let gender = Math.floor(Math.random() * 2);
84     return gender === 0 ? 'Male' : 'Female';
85 }
86
87 // Function to generate a random occupation
88 function getRandomOccupation() {
89     let occupation = getRandomString() + '-' + getRandomNumber();
90     return occupation;
91 }
92
93 // Function to generate a random hobby
94 function getRandomHobby() {
95     let hobby = getRandomString() + '-' + getRandomNumber();
96     return hobby;
97 }
98
99 // Function to generate a random pet name
100 function getRandomPetName() {
101     let petName = getRandomString() + '-' + getRandomNumber();
102     return petName;
103 }
104
105 // Function to generate a random pet type
106 function getRandomPetType() {
107     let petType = Math.floor(Math.random() * 3);
108     return petType === 0 ? 'Dog' : petType === 1 ? 'Cat' : 'Bird';
109 }
110
111 // Function to generate a random pet age
112 function getRandomPetAge() {
113     let petAge = Math.floor(Math.random() * 10) + 1;
114     return petAge;
115 }
116
117 // Function to generate a random pet color
118 function getRandomPetColor() {
119     let petColor = getRandomString() + '-' + getRandomNumber();
120     return petColor;
121 }
122
123 // Function to generate a random pet breed
124 function getRandomPetBreed() {
125     let petBreed = getRandomString() + '-' + getRandomNumber();
126     return petBreed;
127 }
128
129 // Function to generate a random pet owner name
130 function getRandomPetOwnerName() {
131     let petOwnerName = getRandomString() + '-' + getRandomNumber();
132     return petOwnerName;
133 }
134
135 // Function to generate a random pet owner address
136 function getRandomPetOwnerAddress() {
137     let petOwnerAddress = getRandomString() + '-' + getRandomNumber() + '-' + getRandomDate();
138     return petOwnerAddress;
139 }
140
141 // Function to generate a random pet owner phone number
142 function getRandomPetOwnerPhoneNumber() {
143     let petOwnerPhoneNumber = getRandomString() + '-' + getRandomNumber();
144     return petOwnerPhoneNumber;
145 }
146
147 // Function to generate a random pet owner email
148 function getRandomPetOwnerEmail() {
149     let petOwnerEmail = getRandomString() + '@example.com';
150     return petOwnerEmail;
151 }
152
153 // Function to generate a random pet owner name
154 function getRandomPetOwnerName() {
155     let petOwnerName = getRandomString() + '-' + getRandomNumber();
156     return petOwnerName;
157 }
158
159 // Function to generate a random pet owner address
160 function getRandomPetOwnerAddress() {
161     let petOwnerAddress = getRandomString() + '-' + getRandomNumber() + '-' + getRandomDate();
162     return petOwnerAddress;
163 }
164
165 // Function to generate a random pet owner phone number
166 function getRandomPetOwnerPhoneNumber() {
167     let petOwnerPhoneNumber = getRandomString() + '-' + getRandomNumber();
168     return petOwnerPhoneNumber;
169 }
170
171 // Function to generate a random pet owner email
172 function getRandomPetOwnerEmail() {
173     let petOwnerEmail = getRandomString() + '@example.com';
174     return petOwnerEmail;
175 }
176
177 // Function to generate a random pet owner name
178 function getRandomPetOwnerName() {
179     let petOwnerName = getRandomString() + '-' + getRandomNumber();
180     return petOwnerName;
181 }
182
183 // Function to generate a random pet owner address
184 function getRandomPetOwnerAddress() {
185     let petOwnerAddress = getRandomString() + '-' + getRandomNumber() + '-' + getRandomDate();
186     return petOwnerAddress;
187 }
188
189 // Function to generate a random pet owner phone number
190 function getRandomPetOwnerPhoneNumber() {
191     let petOwnerPhoneNumber = getRandomString() + '-' + getRandomNumber();
192     return petOwnerPhoneNumber;
193 }
194
195 // Function to generate a random pet owner email
196 function getRandomPetOwnerEmail() {
197     let petOwnerEmail = getRandomString() + '@example.com';
198     return petOwnerEmail;
199 }
200
201 // Function to generate a random pet owner name
202 function getRandomPetOwnerName() {
203     let petOwnerName = getRandomString() + '-' + getRandomNumber();
204     return petOwnerName;
205 }
206
207 // Function to generate a random pet owner address
208 function getRandomPetOwnerAddress() {
209     let petOwnerAddress = getRandomString() + '-' + getRandomNumber() + '-' + getRandomDate();
210     return petOwnerAddress;
211 }
212
213 // Function to generate a random pet owner phone number
214 function getRandomPetOwnerPhoneNumber() {
215     let petOwnerPhoneNumber = getRandomString() + '-' + getRandomNumber();
216     return petOwnerPhoneNumber;
217 }
218
219 // Function to generate a random pet owner email
220 function getRandomPetOwnerEmail() {
221     let petOwnerEmail = getRandomString() + '@example.com';
222     return petOwnerEmail;
223 }
224
225 // Function to generate a random pet owner name
226 function getRandomPetOwnerName() {
227     let petOwnerName = getRandomString() + '-' + getRandomNumber();
228     return petOwnerName;
229 }
230
231 // Function to generate a random pet owner address
232 function getRandomPetOwnerAddress() {
233     let petOwnerAddress = getRandomString() + '-' + getRandomNumber() + '-' + getRandomDate();
234     return petOwnerAddress;
235 }
236
237 // Function to generate a random pet owner phone number
238 function getRandomPetOwnerPhoneNumber() {
239     let petOwnerPhoneNumber = getRandomString() + '-' + getRandomNumber();
240     return petOwnerPhoneNumber;
241 }
242
243 // Function to generate a random pet owner email
244 function getRandomPetOwnerEmail() {
245     let petOwnerEmail = getRandomString() + '@example.com';
246     return petOwnerEmail;
247 }
248
249 // Function to generate a random pet owner name
250 function getRandomPetOwnerName() {
251     let petOwnerName = getRandomString() + '-' + getRandomNumber();
252     return petOwnerName;
253 }
254
255 // Function to generate a random pet owner address
256 function getRandomPetOwnerAddress() {
257     let petOwnerAddress = getRandomString() + '-' + getRandomNumber() + '-' + getRandomDate();
258     return petOwnerAddress;
259 }
260
261 // Function to generate a random pet owner phone number
262 function getRandomPetOwnerPhoneNumber() {
263     let petOwnerPhoneNumber = getRandomString() + '-' + getRandomNumber();
264     return petOwnerPhoneNumber;
265 }
266
267 // Function to generate a random pet owner email
268 function getRandomPetOwnerEmail() {
269     let petOwnerEmail = getRandomString() + '@example.com';
270     return petOwnerEmail;
271 }
272
273 // Function to generate a random pet owner name
274 function getRandomPetOwnerName() {
275     let petOwnerName = getRandomString() + '-' + getRandomNumber();
276     return petOwnerName;
277 }
278
279 // Function to generate a random pet owner address
280 function getRandomPetOwnerAddress() {
281     let petOwnerAddress = getRandomString() + '-' + getRandomNumber() + '-' + getRandomDate();
282     return petOwnerAddress;
283 }
284
285 // Function to generate a random pet owner phone number
286 function getRandomPetOwnerPhoneNumber() {
287     let petOwnerPhoneNumber = getRandomString() + '-' + getRandomNumber();
288     return petOwnerPhoneNumber;
289 }
290
291 // Function to generate a random pet owner email
292 function getRandomPetOwnerEmail() {
293     let petOwnerEmail = getRandomString() + '@example.com';
294     return petOwnerEmail;
295 }
296
297 // Function to generate a random pet owner name
298 function getRandomPetOwnerName() {
299     let petOwnerName = getRandomString() + '-' + getRandomNumber();
300     return petOwnerName;
301 }
302
303 // Function to generate a random pet owner address
304 function getRandomPetOwnerAddress() {
305     let petOwnerAddress = getRandomString() + '-' + getRandomNumber() + '-' + getRandomDate();
306     return petOwnerAddress;
307 }
308
309 // Function to generate a random pet owner phone number
310 function getRandomPetOwnerPhoneNumber() {
311     let petOwnerPhoneNumber = getRandomString() + '-' + getRandomNumber();
312     return petOwnerPhoneNumber;
313 }
314
315 // Function to generate a random pet owner email
316 function getRandomPetOwnerEmail() {
317     let petOwnerEmail = getRandomString() + '@example.com';
318     return petOwnerEmail;
319 }
320
321 // Function to generate a random pet owner name
322 function getRandomPetOwnerName() {
323     let petOwnerName = getRandomString() + '-' + getRandomNumber();
324     return petOwnerName;
325 }
326
327 // Function to generate a random pet owner address
328 function getRandomPetOwnerAddress() {
329     let petOwnerAddress = getRandomString() + '-' + getRandomNumber() + '-' + getRandomDate();
330     return petOwnerAddress;
331 }
332
333 // Function to generate a random pet owner phone number
334 function getRandomPetOwnerPhoneNumber() {
335     let petOwnerPhoneNumber = getRandomString() + '-' + getRandomNumber();
336     return petOwnerPhoneNumber;
337 }
338
339 // Function to generate a random pet owner email
340 function getRandomPetOwnerEmail() {
341     let petOwnerEmail = getRandomString() + '@example.com';
342     return petOwnerEmail;
343 }
344
345 // Function to generate a random pet owner name
346 function getRandomPetOwnerName() {
347     let petOwnerName = getRandomString() + '-' + getRandomNumber();
348     return petOwnerName;
349 }
350
351 // Function to generate a random pet owner address
352 function getRandomPetOwnerAddress() {
353     let petOwnerAddress = getRandomString() + '-' + getRandomNumber() + '-' + getRandomDate();
354     return petOwnerAddress;
355 }
356
357 // Function to generate a random pet owner phone number
358 function getRandomPetOwnerPhoneNumber() {
359     let petOwnerPhoneNumber = getRandomString() + '-' + getRandomNumber();
360     return petOwnerPhoneNumber;
361 }
362
363 // Function to generate a random pet owner email
364 function getRandomPetOwnerEmail() {
365     let petOwnerEmail = getRandomString() + '@example.com';
366     return petOwnerEmail;
367 }
368
369 // Function to generate a random pet owner name
370 function getRandomPetOwnerName() {
371     let petOwnerName = getRandomString() + '-' + getRandomNumber();
372     return petOwnerName;
373 }
374
375 // Function to generate a random pet owner address
376 function getRandomPetOwnerAddress() {
377     let petOwnerAddress = getRandomString() + '-' + getRandomNumber() + '-' + getRandomDate();
378     return petOwnerAddress;
379 }
380
381 // Function to generate a random pet owner phone number
382 function getRandomPetOwnerPhoneNumber() {
383     let petOwnerPhoneNumber = getRandomString() + '-' + getRandomNumber();
384     return petOwnerPhoneNumber;
385 }
386
387 // Function to generate a random pet owner email
388 function getRandomPetOwnerEmail() {
389     let petOwnerEmail = getRandomString() + '@example.com';
390     return petOwnerEmail;
391 }
392
393 // Function to generate a random pet owner name
394 function getRandomPetOwnerName() {
395     let petOwnerName = getRandomString() + '-' + getRandomNumber();
396     return petOwnerName;
397 }
398
399 // Function to generate a random pet owner address
400 function getRandomPetOwnerAddress() {
401     let petOwnerAddress = getRandomString() + '-' + getRandomNumber() +
```

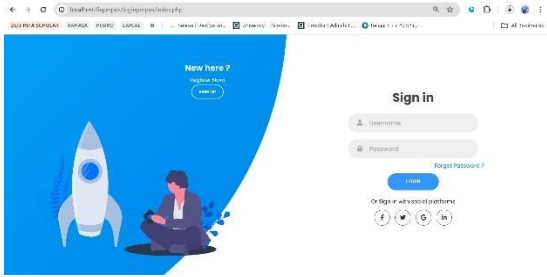
dashboard.php

This code is created for admins and operators to process book loans using the connected fixlibrary database.



15	<pre> 1 // delete.php 2 include 'config/app.php'; 3 \$id_book = \$_GET['id_book']; // ID buku yang akan dihapus 4 5 if (\$id_book &lt;= 0) { 6     echo "error"; 7     header('location: index.php'); 8     exit(); 9 } 10 // Validasi apakah buku tersebut ada di database 11 \$sql = "SELECT * FROM buku WHERE id_book = \$id_book"; 12 \$result = mysqli_query(\$conn, \$sql); 13 if (mysqli_num_rows(\$result) &lt; 1) { 14     echo "error"; 15     header('location: index.php'); 16     exit(); 17 } 18 // Hapus buku dari database 19 \$sql = "DELETE FROM buku WHERE id_book = \$id_book"; 20 \$result = mysqli_query(\$conn, \$sql); 21 if (mysqli_affected_rows(\$conn) &gt; 0) { 22     echo "success"; 23     header('location: index.php'); 24     exit(); 25 } 26 echo "error"; 27 header('location: index.php'); 28 exit(); </pre>	deletebook.php	The code to enable deletion of book data.
16	<pre> 1 // deleteuser.php 2 include 'config/app.php'; 3 \$id_user = \$_GET['id_user']; // ID user yang akan dihapus 4 5 if (\$id_user &lt;= 0) { 6     echo "error"; 7     header('location: index.php'); 8     exit(); 9 } 10 // Validasi apakah user tersebut ada di database 11 \$sql = "SELECT * FROM user WHERE id_user = \$id_user"; 12 \$result = mysqli_query(\$conn, \$sql); 13 if (mysqli_num_rows(\$result) &lt; 1) { 14     echo "error"; 15     header('location: index.php'); 16     exit(); 17 } 18 // Hapus user dari database 19 \$sql = "DELETE FROM user WHERE id_user = \$id_user"; 20 \$result = mysqli_query(\$conn, \$sql); 21 if (mysqli_affected_rows(\$conn) &gt; 0) { 22     echo "success"; 23     header('location: index.php'); 24     exit(); 25 } 26 echo "error"; 27 header('location: index.php'); 28 exit(); </pre>	deleteuser.php	The code to enable deletion of user account data.
17	<pre> 1 &lt;header class="header"&gt; 2 &lt;div class="logo"&gt; 3     &lt;a href="#"&gt;Vocational Library&lt;/a&gt; 4 &lt;/div&gt; 5 &lt;div class="search-box"&gt; 6     &lt;input type="text" placeholder="Search"&gt; 7     &lt;i class="fa-sharp fa-solid fa-magnifying-glass"&gt;&lt;/i&gt; 8 &lt;/div&gt; 9 &lt;div class="header-icons"&gt; 10     &lt;i class="fas fa-bell"&gt;&lt;/i&gt; 11     &lt;div class="account"&gt; 12         &lt;img src="pic/img.jpg" alt=""&gt; 13         &lt;div&gt;Admin&lt;/div&gt; 14     &lt;/div&gt; 15 &lt;/div&gt; 16 &lt;/header&gt; </pre>	header.php	Code at the top of the page.
18	<pre> 1 // return_book.php 2 // return_book.php 3 include 'config/app.php'; 4 // Ambil ID pemesanan dari request 5 \$loan_id = \$_POST['loan_id']; 6 7 // Lakukan operasi pengembalian buku di database 8 // Misalnya, mengurangi kuantitas buku yang dipinjam 9 10 // Simpan kuantitas buku yang baru ke dalam variabel \$new_quantity 11 \$new_quantity = 1; // Ganti dengan kuantitas buku yang sesuai 12 13 // Kirim kuantitas baru kembali ke halaman JavaScript 14 echo \$new_quantity; 15 16 </pre>	returnbook.php	Code to enable book return.
19	<pre> 1 // sidebar.php 2 // sidebar.php 3 // sidebar.php 4 // sidebar.php 5 // sidebar.php 6 // sidebar.php 7 // sidebar.php 8 // sidebar.php 9 // sidebar.php 10 // sidebar.php 11 // sidebar.php 12 // sidebar.php 13 // sidebar.php 14 // sidebar.php 15 // sidebar.php 16 // sidebar.php 17 // sidebar.php 18 // sidebar.php 19 // sidebar.php 20 // sidebar.php 21 // sidebar.php 22 // sidebar.php 23 // sidebar.php 24 // sidebar.php 25 // sidebar.php 26 // sidebar.php 27 // sidebar.php 28 // sidebar.php 29 // sidebar.php 30 // sidebar.php 31 // sidebar.php 32 // sidebar.php 33 // sidebar.php 34 // sidebar.php 35 // sidebar.php 36 // sidebar.php 37 // sidebar.php 38 // sidebar.php 39 // sidebar.php 40 // sidebar.php 41 // sidebar.php 42 // sidebar.php 43 // sidebar.php 44 // sidebar.php 45 // sidebar.php 46 // sidebar.php 47 // sidebar.php 48 // sidebar.php 49 // sidebar.php 50 // sidebar.php 51 // sidebar.php 52 // sidebar.php 53 // sidebar.php 54 // sidebar.php 55 // sidebar.php 56 // sidebar.php 57 // sidebar.php 58 // sidebar.php 59 // sidebar.php 60 // sidebar.php 61 // sidebar.php 62 // sidebar.php 63 // sidebar.php 64 // sidebar.php 65 // sidebar.php 66 // sidebar.php 67 // sidebar.php 68 // sidebar.php 69 // sidebar.php 70 // sidebar.php 71 // sidebar.php 72 // sidebar.php 73 // sidebar.php 74 // sidebar.php 75 // sidebar.php 76 // sidebar.php 77 // sidebar.php 78 // sidebar.php 79 // sidebar.php 80 // sidebar.php 81 // sidebar.php 82 // sidebar.php 83 // sidebar.php 84 // sidebar.php 85 // sidebar.php 86 // sidebar.php 87 // sidebar.php 88 // sidebar.php 89 // sidebar.php 90 // sidebar.php 91 // sidebar.php 92 // sidebar.php 93 // sidebar.php 94 // sidebar.php 95 // sidebar.php 96 // sidebar.php 97 // sidebar.php 98 // sidebar.php 99 // sidebar.php 100 // sidebar.php </pre>	sidebar.php	Code to enable the sidebar on the admin and operator dashboard menus.
20	<pre> 1 // user.php 2 // user.php 3 // user.php 4 // user.php 5 // user.php 6 // user.php 7 // user.php 8 // user.php 9 // user.php 10 // user.php 11 // user.php 12 // user.php 13 // user.php 14 // user.php 15 // user.php 16 // user.php 17 // user.php 18 // user.php 19 // user.php 20 // user.php 21 // user.php 22 // user.php 23 // user.php 24 // user.php 25 // user.php 26 // user.php 27 // user.php 28 // user.php 29 // user.php 30 // user.php 31 // user.php 32 // user.php 33 // user.php 34 // user.php 35 // user.php 36 // user.php 37 // user.php 38 // user.php 39 // user.php 40 // user.php 41 // user.php 42 // user.php 43 // user.php 44 // user.php 45 // user.php 46 // user.php 47 // user.php 48 // user.php 49 // user.php 50 // user.php 51 // user.php 52 // user.php 53 // user.php 54 // user.php 55 // user.php 56 // user.php 57 // user.php 58 // user.php 59 // user.php 60 // user.php 61 // user.php 62 // user.php 63 // user.php 64 // user.php 65 // user.php 66 // user.php 67 // user.php 68 // user.php 69 // user.php 70 // user.php 71 // user.php 72 // user.php 73 // user.php 74 // user.php 75 // user.php 76 // user.php 77 // user.php 78 // user.php 79 // user.php 80 // user.php 81 // user.php 82 // user.php 83 // user.php 84 // user.php 85 // user.php 86 // user.php 87 // user.php 88 // user.php 89 // user.php 90 // user.php 91 // user.php 92 // user.php 93 // user.php 94 // user.php 95 // user.php 96 // user.php 97 // user.php 98 // user.php 99 // user.php 100 // user.php </pre>	user.php	The code is used for user login (student, admin, operator).

## 2.2 Output: Website

No.	Image	Description
1		<p>Sign in page for users (students, admins, operators) to access the main page of the VOKS-Brary information system.</p> <p>Enter the username and password according to the data from the user table in the fixlibrary database.</p>



2

**Vocational Library**

# Unlock a World of Knowledge

What if you're a student, a researcher, or a curious mind that wants a vast collection of books, digital resources, and expert staff and here to support your journey.

**Search Books**

## Why Choose Us

Choose us and experience a library that is committed to unlocking your potential and fueling your passion for knowledge.

- Fast & Free Shipping**  
Get your favorite books with our fast & free shipping service without delay.
- Extensive Collections**  
Our library boasts a diverse and impressive collection of books, journals, digital resources, and multimedia resources catering to all ages and interests.
- 24/7 Support**  
Experience peace of mind knowing that our dedicated team is available round the clock.
- Digital Access**  
Our online resources and digital library are available 24/7, providing you with access to e-books, audiobooks, journals, and databases from the comfort of your home. Stay connected to learning anytime, anywhere.

## Be Smart

Unlock a world of knowledge today!

## Choose your favorite books

Get it Now!

**Atomic Habits** James Clear

**English Nutsch** Adam Saputra

**RICH DAD POOR DAD** Robert Kiyosaki

**Testimonials**

Ever since I joined this library, my reading habits have changed. The collection is extensive, covering diverse topics, and the borrowing process is seamless. It's truly a haven for book lovers like me.

**Candra**  
Entrepreneur

**FOLLOW US**  
Follow

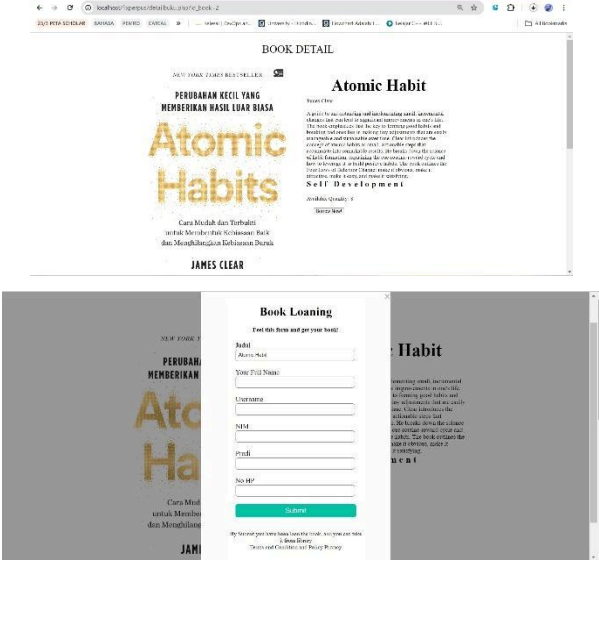
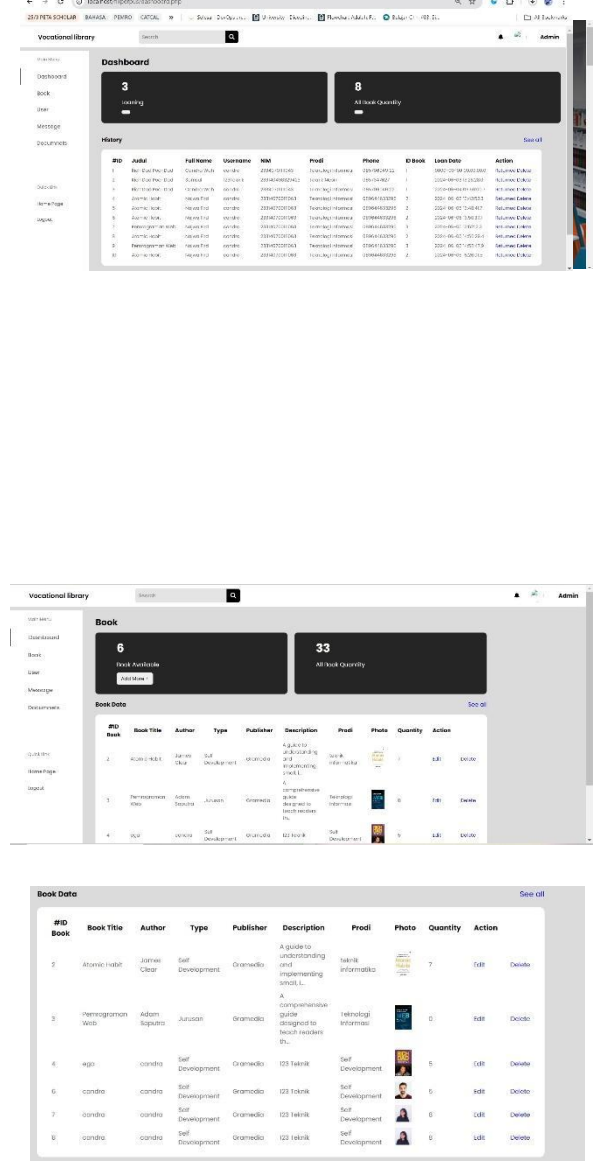
**Vocational Library**  
Copyright © 2023 Vocational Library of Intelligence University. All rights reserved.

**localhost says**  
Are you sure to logout?

**OK** **Cancel**

The main page of the VOKS-Brary information system that can be scrolled down.

Notification when logging out of the user account.

<p>3</p>		<p>Book details after one of the books is clicked on the add icon.</p> <p>The Borrow Now button will open a pop-up book borrowing form that must be filled in with the right data and click submit if it has been filled in.</p>
<p>4</p>		<p>The main page for admins and operators. Overall the content is still the same. The only difference is the navigation bar on the far right shows 'Admin'.</p> <p>Dashboard page for admins and operators which has a sidebar menu containing dashboard, book, user. At the bottom is the homepage and logout.</p>

Vocational Library

Search

Admin

User

7

All User

Admin

Administ

Administ

Admin

Student

See all

ID	Full Name	Username	Password	MM	Prodi	Action
1	CSA B...	csa...	123456789	2024-01-01	Manajemen Perhotelan	edit delete
2	CSA B...	csa...	123456789	2024-01-01	Manajemen Perhotelan	edit delete
3	CSA B...	csa...	123456789	2024-01-01	Manajemen Perhotelan	edit delete
4	CSA B...	csa...	123456789	2024-01-01	Manajemen Perhotelan	edit delete
5	CSA B...	csa...	123456789	2024-01-01	Manajemen Perhotelan	edit delete
6	CSA B...	csa...	123456789	2024-01-01	Manajemen Perhotelan	edit delete
7	CSA B...	csa...	123456789	2024-01-01	Manajemen Perhotelan	edit delete
8	CSA B...	csa...	123456789	2024-01-01	Manajemen Perhotelan	edit delete

Operator

Username	Action
admin	edit delete

BOOK TITLE

Seni Bersikap Bodo Amat

Author

Mark Manson

Type

Self Development

Publisher

Gramedia

Description

Isi sendiri

Prodi

Manajemen Perhotelan

Photo

Choose File new\_release\_g-preview.png

Quantity

1

Add

localhost says

User Data added successfully

OK

The add book form can be accessed and filled in by the operator and there is a notification if the book data has been successfully added.

## **BAB III CLOSURE**

### **3.1 The Conclusion**

The VOKS-Brary library information system above can be used by users (students, admins, operators) whose accounts have been registered and recorded in the database. The system above is dynamic which is connected to the database that has been created. Features that can be used by this system are book borrowing by students and confirmation regarding book availability, book retrieval, and book borrowing by operators. The rest, users can see the book catalogue available on the system to borrow if needed.

### **3.2 The Advice**

From the design of the information system above, there are several parts that are still unresolved, inconsistencies in some parts, features that are still not accessible, and other technical obstacles. Hopefully, this system can be completed to the final level and can be used and accessed by those who need it at the Faculty of Vocational Studies, Universitas Brawijaya and implemented with experts in their fields.

## LITERATURE LIST

- IDCloudHost, C. (2022, April 19). *Pengertian Sistem Informasi, Fungsi, Komponen dan Contohnya*. IDCloudHost. <https://idcloudhost.com/blog/sistem-informasi-adalah/>
- Patria, R. (2023, August 28). *Pengertian HTML, Fungsi dan Struktur dan Contohnya - DomaiNesia*. DomaiNesia. <https://www.domainesia.com/berita/html-adalah/>
- Nayoan, A. (2022, December 6). *Apa Itu CSS? Pengertian, Fungsi, dan Contohnya*. Niagahoster Blog. <https://www.niagahoster.co.id/blog/pengertian-css/>
- A, F., & A, F. (2023, December 4). *Apa Itu PHP? Pengertian PHP untuk Pemula*. Hostinger Tutorial. <https://www.hostinger.co.id/tutorial/apa-itu-php/>
- Andre. (2019, Juli 22). *Tutorial Belajar PHP Part 1: Pengertian dan Fungsi PHP dalam Pemrograman Web*. Duniaikom. <https://www.duniaikom.com/pengertian-dan-fungsi-php-dalam-pemograman-web/>