THE ACADEMY
OF APPLIED TECHNICAL
AND PRESCHOOL STUDIES

Application for classroom scheduling

Student:

Miloš Petrović ITS 07/23

THE ACADEMY OF APPLIED TECHNICAL AND PRESCHOOL STUDIES

Table of contents

THE ACADEMY OF APPLIED TECHNICAL AND PRESCHOOL STUDIES

1. Introduction

I worked on developing a web application for classroom reservation at the faculty, using React for the front end and PHP for the back end. The goal of the application is to enable professors to check classroom availability and book time slots for lectures or other activities.

2. Project Description

The classroom scheduling application has been developed as a practical solution for managing reservations and usage schedules for classrooms in educational institutions. The main goal of this project is to simplify the process of reserving classrooms, reduce scheduling conflicts, and ensure transparency and efficiency in the use of space.

2.1 Project Goals

The primary objectives of the project are:

- To provide users (professors, students, administrative staff) with an easy way to check classroom availability.
- To offer a straightforward and intuitive method for reserving classrooms.
- To automatically avoid scheduling conflicts and notify users of any overlaps.
- To ensure transparency in resource usage through a clear display of all reservations.

2.2 Features

The classroom scheduling application includes the following key functionalities:

- Login and Authentication: Users can log in through a simple interface. The authentication system ensures that only authorized users can make reservations.
- Availability Overview: An interactive calendar allows users to view classroom availability in real-time.
- Booking Appointments: Users can reserve classrooms for specific time slots.
- Administrative Overview and Management: Administrators have special access for reviewing and managing reservations.

3. Technologies

Frontend: React, CSS

Backend: PHP, MySQL

4. Project Description through Back-End Technologies

The back-end consists of several key components:

4.1 PHP part:

- intervals.php: This file is responsible for manipulating data related to occupied and free time slots for classrooms.

```php
$workingHoursStart = 8;
$workingHoursEnd = 21;

$freeIntervals = array();
$lastEndTime = $workingHoursStart;

foreach ($busyIntervals as $interval) {
    $startTime = (int)$interval['startTime'];
    $endTime = (int)$interval['endTime'];

    if ($lastEndTime < $startTime) {
        $freeIntervals[] = array(
            'startTime' => $lastEndTime,
            'endTime' => $startTime
        );
    }
    $lastEndTime = max($lastEndTime, $endTime);
}

if ($lastEndTime < $workingHoursEnd) {
    $freeIntervals[] = array(
        'startTime' => $lastEndTime,
        'endTime' => $workingHoursEnd
    );
}
```

Image 1: PHP code for checking occupied/free intervals

- scheduled_appointments.php: This file allows for retrieving scheduled appointments for a specific professor based on the session.

```php
if (isset($_SESSION['name'])) {
    $professor_name = $_SESSION['name'];

    $query = "SELECT * FROM busy_classrooms WHERE professor = '$professor_name'";
    $result = query($query);

    confirm($result);

    $appointments = fetch_all($result);

    echo json_encode($appointments);
} else {
    echo json_encode(["success" => false, "message" => "Niste ulogovani."]);
}
```

Image 2: Retrieving occupied time slots

- busy_insert.php: This file allows for adding new booked time slots for classrooms to the database.

```php
$sql = "INSERT INTO busy_classrooms (calendar, startTime, endTime, amphitheater, professor)
        VALUES (:calendar, :startTime, :endTime, :amphitheater, :professor)";
```

Image 3: SQL query for inserting a booking

- check_classroms.php: This file updates the occupancy status of classrooms and deletes past entries from the database based on the current time and date.

```php
$sql = "SELECT amphitheater, startTime, endTime, professor
        FROM busy_classrooms
        WHERE calendar = '$currentDate'
        AND startTime <= '$currentTime'
        AND endTime > '$currentTime'";
$result = $conn->query($sql);
```

Image 4: SQL query for displaying/updating occupied rooms

- add_professor.php: This file allows for adding new professors to the database.

```php
$name = isset($data['name']) ? $data['name'] : null;
$surname = isset($data['surname']) ? $data['surname'] : null;
$email = isset($data['email']) ? $data['email'] : null;
$password = isset($data['password']) ? password_hash($data['password'], PASSWORD_BCRYPT) : null;

if ($name && $surname && $email && $password) {
    $query = "INSERT INTO professors (name, surname, email, password) VALUES (?, ?, ?, ?)";
    $stmt = $con->prepare($query);
    $stmt->bind_param("ssss", $name, $surname, $email, $password);

    if ($stmt->execute()) {
        echo json_encode(["success" => true, "message" => "Profesor uspešno dodat"]);
    } else {
        echo json_encode(["success" => false, "message" => "Greška prilikom dodavanja profesora"]);
    }

    $stmt->close();
} else {
    echo json_encode(["success" => false, "message" => "Nevažeći unos"]);
}
```

Image 5: PHP code for adding a professor

- delete_professor.php: This file allows for deleting a professor from the database based on the received professor ID.

```php
if ($_SERVER['REQUEST_METHOD'] === 'DELETE') {
    if (isset($_GET['id'])) {
        $id = $_GET['id'];

        $sql = "DELETE FROM professors WHERE id = ?";
        if ($stmt = $con->prepare($sql)) {
            $stmt->bind_param("i", $id);
            if ($stmt->execute()) {
                echo json_encode(['success' => true, 'message' => 'Profesor uspešno obrisan']);
            } else {
                echo json_encode(['success' => false, 'message' => 'Greška pri brisanju profesora']);
            }
```

Image 6: PHP code for deleting a professor

- update_professor.php: This file allows for updating professor data in the database.

```php
$stmt = $con->prepare("UPDATE professors SET name = ?, surname = ?, email = ? WHERE id = ?");
$stmt->bind_param("sssi", $name, $surname, $email, $id);
$stmt->execute();
```

Image 7: PHP code for updating professor credentials

4.2 MySQL – Database Section:

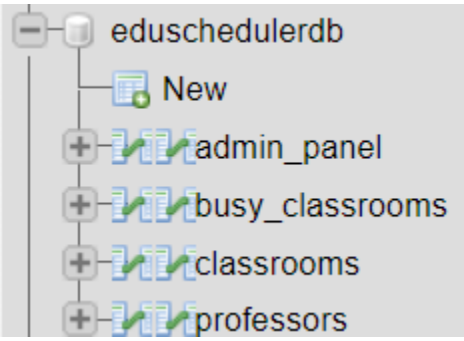

Image 8: Database schema with tables

Display of a portion of the database structure named eduschedulerdb. This database contains several tables used for managing schedules.

1. **admin_panel**:
   - This table contains data related to administrators in the application.

id	username	password
1	admin	$2y$10$1Ac9XfEVSNhDYO/ONwHX2eUVSUlExphChdYKy04t.jh...

Image 9 Contents of the admin_panel table

2. **busy_classrooms**:
   - This table contains information about occupied classrooms.

amphitheater	startTime	endTime	professor	reason	calendar
A1	14	16	Aleksandra		2024-07-15
U7	8	10	Aleksandra		2024-07-17
A3	16	18	Dušan		2024-07-23
A2	12	14	Dušan		2024-07-29
U106	13	15	Slađana		2024-07-30
U207	8	12	Slađana		2024-08-20

Image 10: Contents of the busy_classrooms table

3. **classrooms**:

- This table contains basic information about all classrooms.

id	classroom	occupied	features	floor
1	U1	0	Projector	1
2	U2	0	Projector, Computer Lab	1
3	U3	0	Lab Equipment, Projector	1
4	U4	0	Projector, Whiteboard	1
5	U5	0	Seminar Room, Projector	1
6	U6	0	Computer Lab, Projector	1
7	U7	0	Whiteboard, Seminar Room	1
8	U8	0	Projector, Whiteboard	1
9	U9	0	Projector, Computer Lab	1
10	A1	0	Lab Equipment, Projector	1
11	A2	0	Projector, Whiteboard	2
12	A3	0	Projector, Computer Lab	2

Image 11: Contents of classrooms table

4. **professors**:

- Ova tabela sadrži informacije o profesorima.

id	name	surname	email	password
1	Aleksandra	Boričić	aleksandra.boricic@akademijanis.edu.rs	$2y$10$4jbgA54LMhS8vAvZMaN3tOu46RPKIIo8oLHTlfqU6Kp...
2	Aleksandra	Marinković	aleksandra.marinkovic@akademijanis.edu.rs	$2y$10$17nd9.HbEiShBAuVfNuRuORohLjvm4bSp6nQJmzwJFu...
3	Anica	Milošević	anica.milosevic@akademijanis.edu.rs	$2y$10$7l0gNWkS.tTASh9T4kBD8.9LNj6OaTNay55.YSgJ.DH...
4	Biljana	Milutinović	biljana.milutinovic@akademijanis.edu.rs	$2y$10$5EadOwKtFfSs1l3ULAUYkuFRiQKFCyvgLC0VHpYjHw6...
5	Boban	Cvetanović	boban.cvetanovic@akademijanis.edu.rs	$2y$10$03sjhFmQ3tvX6lRnEKsxcOaxim0QcNNgiewLmIh.VU5...
6	Danijela	Zlatković	danijela.zlatkovic@akademijanis.edu.rs	$2y$10$dRJg9XqLnKN4sh/g1FylQ.EGfhn7iF5qlmgTAvXBIT7...
7	Dejan	Blagojević	dejan.blagojevic@akademijanis.edu.rs	$2y$10$Y6hTxKUDRcFm/OFONtkzjuEJWfiMXUpPeQ5sU99lhje...
8	Dejan	Bogićević	dejan.bogicevic@akademijanis.edu.rs	$2y$10$mm.GcF/UHW.sNx/ZaRL1HedQuklN5ElZwc.348RwCJw...
9	Dušan	Radosavljević	dusan.radosavljevic@akademijanis.edu.rs	$2y$10$oe7DPQ4BslB.rk1q5J3uKes2iV/2sCQXFMbbmh2qk7c...
10	Dušan	Stefanović	dusan.stefanovic@akademijanis.edu.rs	$2y$10$tLJAR1nKkPpq2Xs.4xxlheer5LTyj1IoP.53oHegtX4...

Image 12: Contents of the professors table

5. User Interface

Special emphasis was placed on an intuitive and user-friendly interface. The user experience is optimized to allow easy navigation and quick classroom booking. An interactive calendar, clearly marked available and occupied classrooms, and a simple login and authentication system make this application efficient and easy to use.

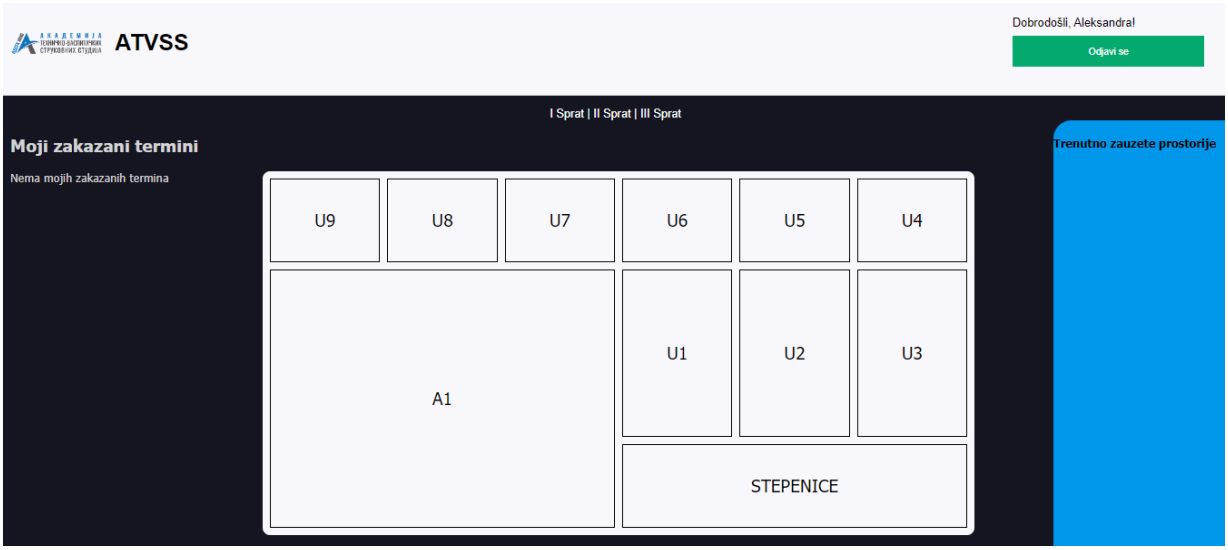

Image 13: User interface

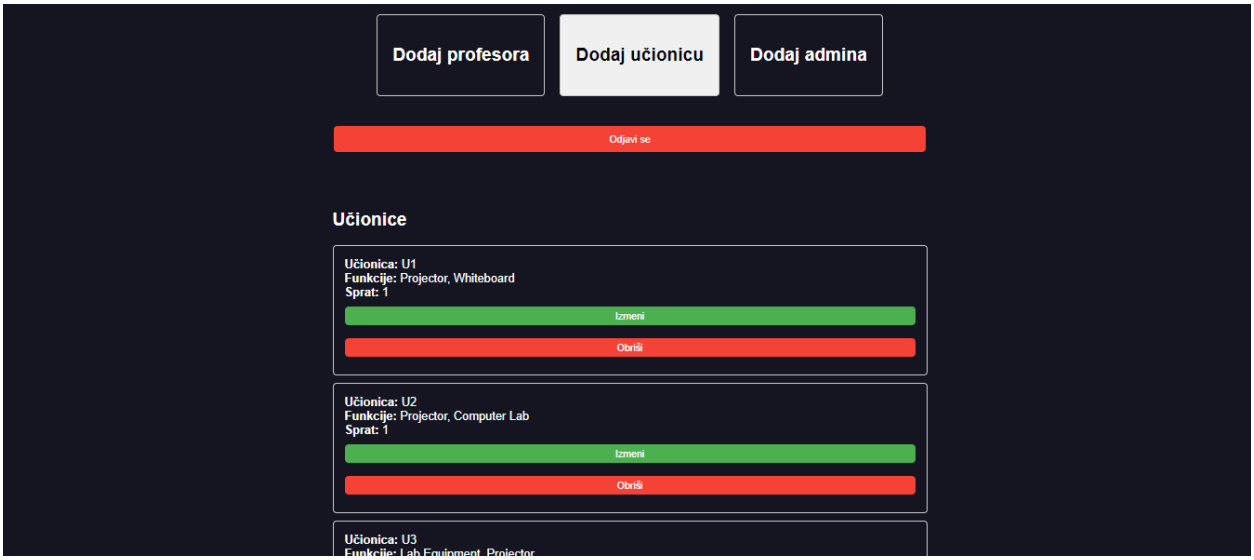

Image 14: admin user interface

6. Benefits

The implementation of this application brings numerous benefits:

- Increased Efficiency: Faster and simpler classroom booking reduces administrative costs and enhances efficiency.
- Conflict Reduction: Automatic scheduling conflict control minimizes the chance of overlaps.
- Transparency: Clear records of all reservations enable better organization and planning.

7. Challenges

One of the new challenges we encountered is enabling communication between two different ports. As previously mentioned, this application was developed using the React.js library as a front-end tool running on port 3000, while the back-end operates on port 80. To allow smooth communication between these two ports, it is necessary to add a CORS policy that permits specific methods for interaction between these two ports.

7.1 What is CORS?

CORS (Cross-Origin Resource Sharing) is a security mechanism implemented in web browsers that allows or restricts resources shared between different domains. In the context of web applications, this means that an application running on one domain (or port) can request resources (such as API data) from another domain. Without a properly configured CORS policy, browsers will block these requests for security reasons. A CORS policy defines which domains and HTTP methods are allowed to access resources on the server, ensuring secure data exchange between different sources. [Image 15]

```php
if (isset($_SERVER['HTTP_ORIGIN'])) {
    header("Access-Control-Allow-Origin: {$_SERVER['HTTP_ORIGIN']}");
    header('Access-Control-Allow-Credentials: true');
    header('Access-Control-Max-Age: 86400'); // cache for 1 day
}

if ($_SERVER['REQUEST_METHOD'] == 'OPTIONS') {
    if (isset($_SERVER['HTTP_ACCESS_CONTROL_REQUEST_METHOD']))
        header("Access-Control-Allow-Methods: GET, POST, OPTIONS");

    if (isset($_SERVER['HTTP_ACCESS_CONTROL_REQUEST_HEADERS']))
        header("Access-Control-Allow-Headers: {$_SERVER['HTTP_ACCESS_CONTROL_REQUEST_HEADERS']}");

    exit(0);
}
```

Image 15: Code for CORS policy

The image shows PHP code implementing CORS (Cross-Origin Resource Sharing) headers for HTTP requests.

1. The first if condition is the check for the HTTP_ORIGIN header:

• Ako je postavljeno HTTP_ORIGIN zaglavlje (koje označava domen sa kojeg potiče zahtev), postavlja se Access-Control-Allow-Origin zaglavlje da dozvoli taj domen.

- HTTP_ORIGIN zaglavlje je deo HTTP zahteva koji sadrži informaciju o originalnom izvoru (domeni) sa kojeg je zahtev poslat. Ovaj header je ključan za CORS (Cross-Origin Resource Sharing) mehanizam, koji omogućava web aplikacijama na jednom domenu da pristupaju resursima na drugom domenu.

• Access-Control-Allow-Credentials: true omogućava da se šalju kolačići i HTTP autentifikacija sa zahtevima.

• Access-Control-Max-Age: 86400 postavlja keširanje CORS odgovora na jedan dan (86400 sekundi).

2. The second if condition is the check for the HTTP method::

- If the HTTP request method is OPTIONS, it checks for the presence of the HTTP_ACCESS_CONTROL_REQUEST_METHOD header and sets Access-Control-Allow-Methods to: GET, POST, OPTIONS to allow the specified HTTP methods.

- If the HTTP_ACCESS_CONTROL_REQUEST_HEADERS header is set, Access-Control-Allow-Headers is configured with the values of the headers requested by the client.

- exit(0) stops further execution of the code after sending the response to the OPTIONS request.

8. Conclusion

The classroom scheduling application project represents a practical and modern solution for managing educational resources. Its implementation can significantly contribute to efficiency, organization, and user satisfaction in educational institutions.

This project allowed me to gain practical experience in web application development using React and PHP. I learned how to effectively manage application state, work with backend services, and create interactive and user-friendly interfaces. This project was challenging but extremely beneficial for my further professional development.