Proceedings of the Informatics Conference Vol. 11 No. 22 (2025): Requirements Engineering Methodologies ISSN: 2477-5894

Web Based Midwife Reservation System with Laravel PHP and MySQL Database

Teddy Lioner, *Student, Universitas Budiluhur*
Rinaldi Febryatna, *Student, Universitas Budiluhur*
Andika Hasbigumdi Sudewo, *Student, Universitas Budiluhur*
Yusup Maolani, *Student, Universitas Budiluhur*
Setyawan Widyarto, *Lecturer, Universiti Selangor*

***Abstract*— In recent years, web applications have increasingly streamlined administrative processes in various domains; however, specialized solutions for midwives remain limited. This study addresses this gap by developing a web-based midwife reservation system using the Laravel PHP framework and MySQL database. The system is designed to support efficient scheduling, enhance patient engagement, and provide open-source accessibility to promote collaboration and customization. Key features include automated appointment management, timely notifications, and user-friendly interfaces tailored to the unique requirements of midwives and patients. The development process adopts agile and kanban methodologies to ensure flexibility, iterative improvements, and alignment with user needs, while black-box testing is employed to validate functionality and reliability. The proposed system demonstrates the potential of web-based technologies to improve healthcare services for midwives, optimize administrative efficiency, and strengthen patient-provider interactions.**

***Index Terms*—Tocology, Web Application, Efficient Scheduling, Patient Engagement, Open Source, PHP, MySQL**

I. Introduction

In recent years, web applications have revolutionized how professionals manage their clients. From appointment scheduling to data storage, these applications simplify administrative tasks and enhance overall efficiency. However, existing solutions often lack specialization, leaving certain professional domains underserved. One such area is tocology—the field of gynecology focused on pregnancy and childbirth.

While numerous studies have explored web applications for restaurant reservations [1][2], tourism bookings [3], and general healthcare management [4], there is a noticeable gap when it comes to specialized solutions for tocologists or midwives. Existing literature rarely delves into the unique requirements of this niche group. As a result, tocologists or midwives often resort to manual scheduling methods, leading to inefficiencies and missed opportunities for patient engagement.

To address this research gap, our study includes interviews with practicing tocologists or midwives. By directly involving these experts, we gain insights into their specific needs and pain points. Our target audience includes solo practitioners and small clinics, where personalized patient care is paramount. The main feature of our web application is appointment scheduling, tailored to the busy schedules of tocologists or midwives.

Our primary objective is to create a user-friendly web application that tocologists or midwives can readily deploy. We focus on the following goals.

Efficient Scheduling: Our system streamlines scheduling by automating manual processes. Whether it's appointment bookings, resource allocation, or event coordination, our solution ensures efficiency and accuracy.

Patient Engagement: Through timely reminders and personalized communication, we enhance patient interactions. Patients receive relevant information, follow-up messages, and notifications, leading to better engagement and adherence to appointments or treatments.

Open Source: We believe in openness and collaboration. Our source code is freely available online, encouraging developers to explore, customize, and contribute. By fostering an open-source community, we aim to improve healthcare technology for everyone.

We follow established software engineering practices throughout the development process. Our approach combines elements of agile and kanban methodologies, allowing flexibility while maintaining project discipline. Additionally, we employ black-box testing strategies to ensure the application's reliability and security.

In summary, our research aims to bridge the gap in specialized web applications for tocologists or midwives, offering a practical solution that combines ease of use, scalability, and patient-centric features.

Proceedings of the Informatics Conference Vol. 11 No. 22 (2025): Requirements Engineering Methodologies ISSN: 2477-5894

II. Methodology

A. Agile

Agile, traditionally a software development framework, has been increasingly adopted in research [2][5]. This study explores the application of Agile principles to the research process, specifically focusing on the concept of sprints with scrum [5]. A sprint is a time-bound period, typically two weeks as recommended in [6] for our case, dedicated to completing a specific set of research objectives.

During sprint planning, the engineering team identifies and prioritizes research questions based on their significance and feasibility. For instance, in our study of patient engagement with a healthcare application, we defined specific research questions about patient interactions and feedback. This focused approach ensures that the most critical aspects of the research are addressed first.

Data collection and analysis occur iteratively throughout the sprint. By employing methods such as interviews with a tocologist, we gather data and analyze it promptly to gain initial insights. After that, in our research on reservation management features, we conducted short user interviews, patients, in each sprint to understand user preferences and incorporate their feedback into the design process.

Regular interaction with stakeholders is essential in Agile research. Throughout the project, we shared findings, insights, and progress with advisors, peers, and potential users. Their feedback was invaluable in guiding the research direction and making necessary adjustments. For instance, presenting prototypes of the reservation management system to stakeholders allowed us to incorporate their suggestions into the design process.

The two-week sprint structure imposed a sense of urgency and efficiency on the research process. By compressing the research cycle into focused bursts, we were able to achieve tangible results at the end of each sprint. These results included refined research questions, developed research instruments, and initial data analysis.

B. PHP: A Server-Side Scripting Language

PHP, an acronym for Hypertext Preprocessor, is an open-source programming language primarily used for web development. Embedded within HTML, PHP code is executed on the server to generate dynamic web content [7].

When a user requests a PHP file, the web server processes the script. PHP code is interpreted and executed, generating HTML output that is then sent to the client's browser. This dynamic process allows for interactive web applications, unlike static HTML pages which provide fixed content.

Essentially, PHP acts as a bridge between the server and the client, handling data processing and presentation logic.

Currently, PHP is supported by vast majority of hosting providers, which makes it almost main language with which you can develop any Web project, therefore, there is lot of work in this area [8].

The results of the analysis allow us to conclude that there are several advantages to PHP technology in terms of performance characteristics. The main benefits of PHP, as we see, include practicality, efficiency, performance, and flexibility [9]

C. MySQL

MySQL is a widely-used, open-source database management system (DBMS) that stores data in a structured format [7]. Composed of one or more tables, it efficiently manages and organizes information. MySQL's popularity stems from its reliability, performance, and flexibility.

MySQL is used as a relational database for managing asset data. The use of SQL allows systematic and efficient recording, updating, and deletion of asset data [10]. Within MySQL, data is organized into tables, and each piece of information is stored in a field. These fields have specific data types, such as text, numbers, or dates, to ensure data integrity and efficiency.

III. Result and Discussions

A. Software Requirements

This section focuses on the functional requirements of the system, as detailed in the use case, activity, and sequence diagrams presented in the Appendix. These diagrams provide a visual representation of the system's behavior and interactions.

The system encompasses several core functionalities as shown in Table 1.

TABLE I
USE CASES

Proceedings of the Informatics Conference Vol. 11 No. 22 (2025): Requirements Engineering Methodologies ISSN: 2477-5894

No	Functional	Description	Actor
1	Login	The process of entering the user dashboard by inputting username and password	All Users
2	Midwife Data Management, Patient Data Management, Admin Data Management, Practice Hours Management, Practice Data Management, Practice Schedule History	Adding, editing, deleting midwife data, patient data, admin data, practice hours data, practice data, viewing practice schedule history	Admin
3	Practice Data Management, Practice Schedule History	Adding, editing, deleting practice schedule data, viewing practice schedule history	Midwife
4	Reservation History, Appointment Reservation, Reschedule Appointment	Viewing reservation history, making reservations, rescheduling appointments	User/Patient

Activity diagrams provide a detailed breakdown of the system's processes, such as appointment booking and data management. Sequence diagrams visualize the interactions between system components and users during specific use cases. These diagrams enhance understanding of information and control flow within the system.

Black box testing will be employed to verify the system's adherence to the identified functional requirements. Test cases, derived from the use cases, will assess the system's performance under various conditions.

By meticulously defining and testing the functional requirements, the SIREBI system aims to effectively cater to the needs of midwives, patients, and administrators in managing appointments and associated data.

B. Software Architecture

The website application is an application built with Laravel programming, Laravel is a PHP framework that is often used to build web applications and is included in the Model-View-Controller (MVC) architecture, Software application architecture is a process for defining the structure of an application that can fulfill all criteria from a technical and operational perspective, with quality considerations such as performance, security and manageability.

The website used Laravel follows the Model-View-Controller (MVC) architectural pattern , which separates the application into three key components: Models, Views, and Controllers [11]. This separation enhances code organization, promotes reusability, and makes maintenance a breeze. Models handle data logic, Views manage presentation, and Controllers handle user interaction and orchestration of data flow.

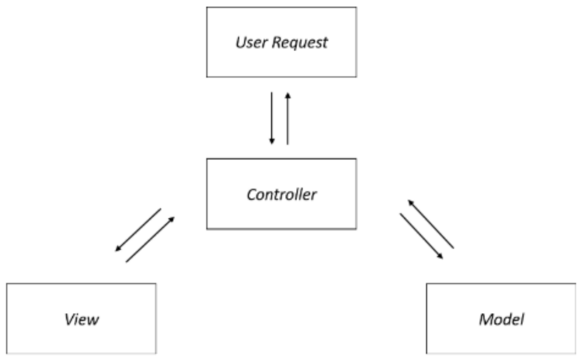


Fig. 1. MVC Design Pattern

Each layer of the application is separate by using the API method, but what is still interconnected is memory handling, because each communication layer will carry/send data so that memory allocation will occur and ultimately requires memory handling.

There are 4 parts of layered architecture where each layer has a relationship between the components in it from top to bottom, namely

1. The presentation layer: all parts related to the presentation layer.
2. The business layer: deals with business logic.
3. The persistence layer: useful for managing all functions related to relational objects
4. The database layer: Where all layer data is stored.

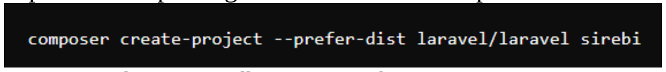
C. Software Design and Development

The application was developed using the PHP programming language and the Laravel framework. Laravel is an open-source PHP framework that makes it easy to develop web applications with the MVC (Model-View-Controller) architectural pattern. In the MVC concept users interact with the system through a user interface (UI), called a View, their actions such as filling in a search form and pressing the "Search" button trigger an HTTP request to the server. This request then passes through the routing system, which is tasked with determining where the request should be forwarded based on the URL and HTTP method used, mapping the request to the appropriate Controller. The controller is responsible for controlling the flow of data between the Model and View. After receiving a request from the router, the Controller will process it and interact with the Model to get the required data. The model manages application data by accessing the database and performing the business logic necessary to fulfill those requests, then returning the data to the Controller.

Proceedings of the Informatics Conference Vol. 11 No. 22 (2025): Requirements Engineering Methodologies ISSN: 2477-5894

The Controller then processes the data received from the Model and sends it to the View. View is tasked with displaying this data to users in an easy-to-understand format. As a result, the information sought by the user is displayed on the screen. With this structure, Laravel helps developers in building structured and efficient web applications. A clear separation of responsibilities between Model, View, and Controller, as well as routing that determines the flow of requests, ensures that application code remains well organized, simplifying the process of developing, maintaining, and testing applications.

The first stage carried out in developing the application is installing the Laravel project using the composer command at the command prompt. Composer is a package manager for the PHP programming language used to manage dependencies and libraries required by applications. In the context of Laravel, Composer is used to install the Laravel framework and all the dependencies it requires. Additionally, Composer allows developers to manage their application dependencies via the composer.json file, which will automatically download and install the necessary libraries. Composer also provides an autoloading feature that makes it easy to load PHP classes without the need for manual require or include, and simplifies the process of updating libraries with one simple command.



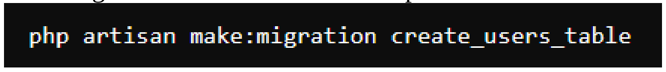
```
composer create-project --prefer-dist laravel/laravel sirebi
```

Fig. 2. Laravel project installation command

The Laravel project installation process automatically generates several main folders and files by default when the application is first run. These folders and files include:

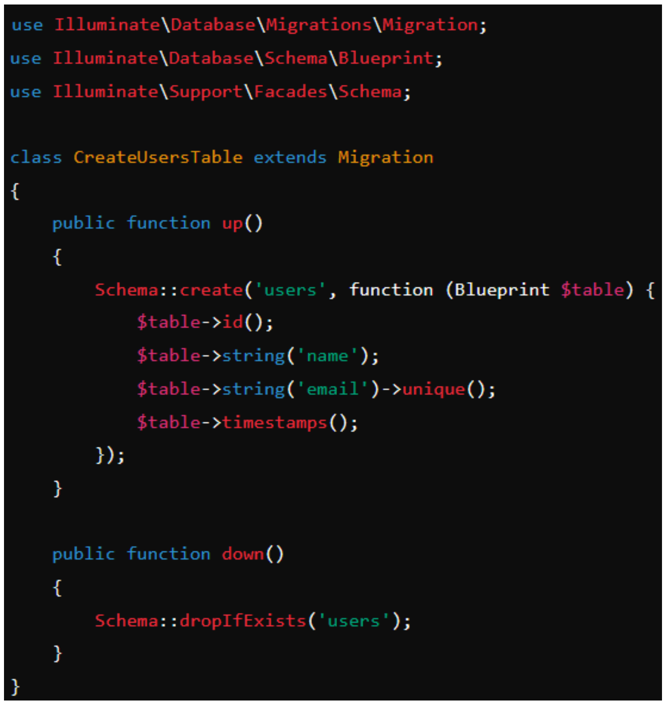
1. **app/** : Contains application code, including controllers, middleware, models, and providers.
   - **Http/Controllers/** : Location of application controllers.
   - **Http/Middleware/** : Places application middleware.
   - **Models/** : Location of Eloquent models.
2. **bootstrap/** : Contains files for bootstrap and the framework cache.
   - **cache/** : Saves the application cache.
3. **config/** : Contains configuration files for various services and application settings.
4. **database/** : Contains files related to the database, such as migrations, seeders, and factories.
   - **migrations/** : Saves database migration files.
   - **seeders/** : Saves database seeder files.
5. **public/** : The root directory of the web server that stores public files such as index.php.
   - **index.php** : Entry point for all HTTP requests.
6. **resources/** : Contains view assets and language resources.
   - **views/** : View Blade for the user interface.
7. **routes/** : Contains application route definitions.
   - **web.php** : Routes for web applications.
   - **api.php** : Routes for the API.
8. **storage/** : Stores files generated by Laravel, such as logs, cache, and uploaded files.
   - **logs/** : Saves application logs.
9. **tests/** : Contains unit tests and feature tests.
   - **Feature/** : Place to test application features.
   - **Unit/** : Place of application unit tests.
10. **vendor/** : Stores all third-party dependencies installed by Composer.
11. **.env** : Environment configuration file that stores sensitive settings such as database connections.
12. **artisan** : Laravel CLI used to run various Artisan commands.
13. **composer.json** : Stores a list of Composer dependencies and settings.

After the installation process is complete, the next stage is to create a database design using the migration command. Migration is a feature in Laravel that allows developers to manage and organize database schemas in an easy and structured manner. With migration, developers can create, change, and delete tables and columns in the database using PHP code, without having to write SQL manually. Migration functions as a versioning system for databases, allowing developers to track database schema changes over time and facilitating collaboration within development teams.



```
php artisan make:migration create_users_table
```

Fig. 3. User table migration command



```
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateUsersTable extends Migration
{
    public function up()
    {
        Schema::create('users', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->string('email')->unique();
            $table->timestamps();
        });
    }

    public function down()
    {
        Schema::dropIfExists('users');
    }
}
```

Fig. 4. User table design

After the table design process is complete, the migration process is carried out from the Laravel project script to the database. In the application, the database system used is MySQL. The table design that has been generalized into a database can be seen in Fig. 5.

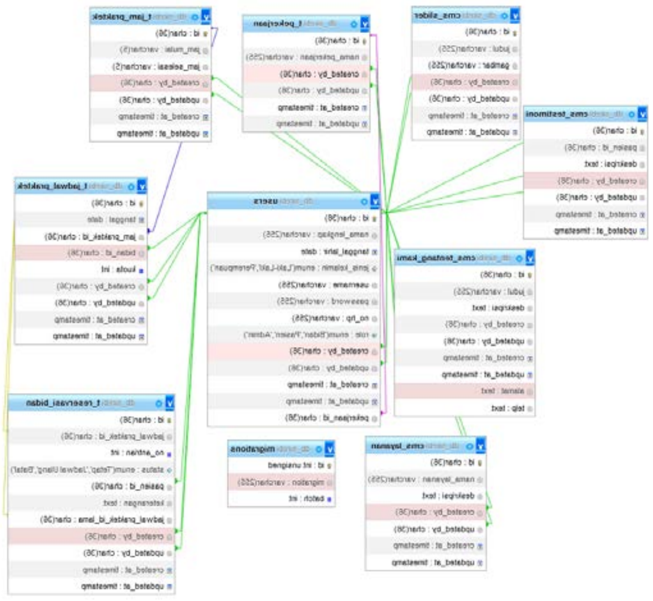
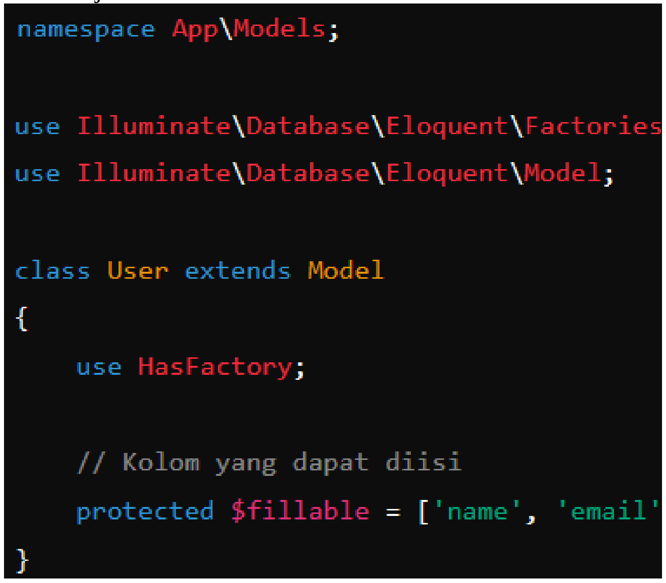


Fig. 5. Database Diagram

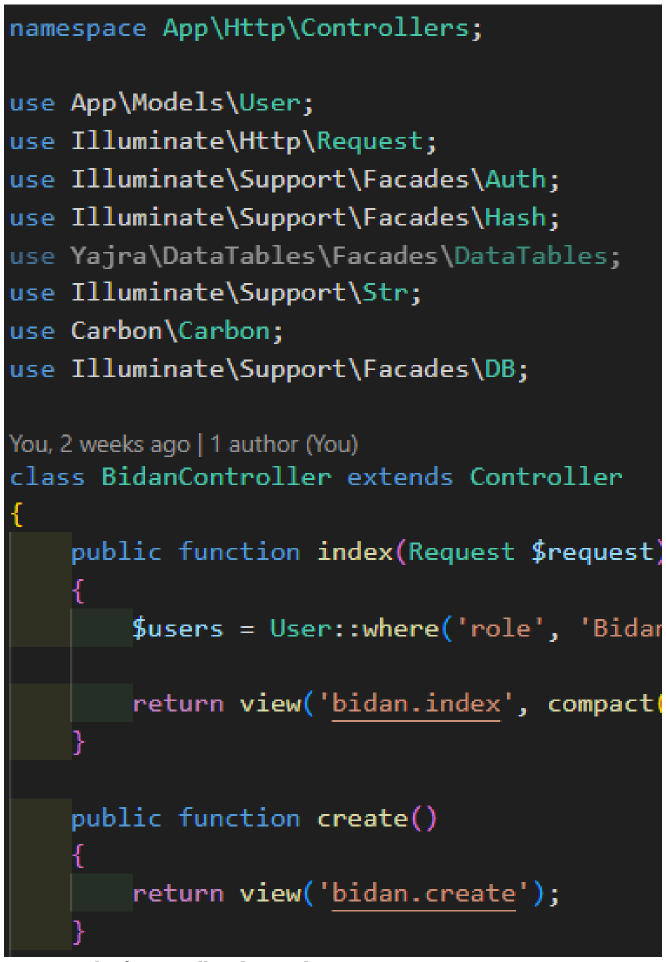
After all the necessary tables have been created, the next step is to create a model. Models in Laravel are representations of tables in the database. Models are used to interact with tables, retrieve data, insert data, as well as perform various other database-related operations. The model in Laravel follows the concept of ORM (Object-Relational Mapping) through Eloquent ORM, which allows developers to interact with the database using PHP syntax that is more intuitive than straight SQL. In the application the number of model classes created will be adjusted to the number of tables in the database.



```
namespace App\Models;

use Illuminate\Database\Eloquent\Factories
use Illuminate\Database\Eloquent\Model;

class User extends Model
{
    use HasFactory;

    // Kolom yang dapat diisi
    protected $fillable = ['name', 'email'
}
```

Fig. 6. User model class code

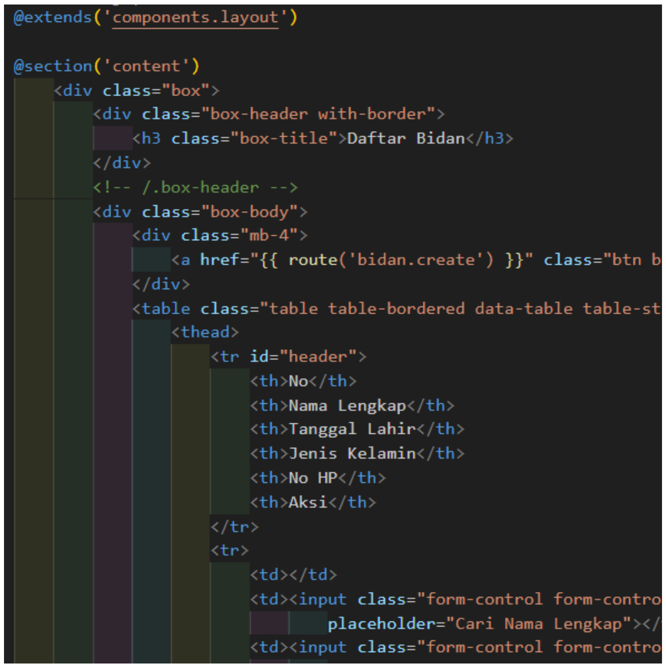
The next step is to create a controller class. The controller in Laravel is a component that handles application logic and acts as a bridge between the model (data) and the view (display). Controllers receive input from users via HTTP requests, process them using models, and then return appropriate responses in the form of views or other data. With a controller, developers can separate business logic from views and data, so the code becomes more structured and easy to manage. In the controller class, business logic functions such as CRUD (Create, Read, Update, Delete) are created. In the development of the application, the functions created follow the number of modules and forms which will be continued with the screen design on the interface display.



```
namespace App\Http\Controllers;

use App\Models\User;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Hash;
use Yajra\DataTables\Facades\DataTables;
use Illuminate\Support\Str;
use Carbon\Carbon;
use Illuminate\Support\Facades\DB;

You, 2 weeks ago | 1 author (You)
class BidanController extends Controller
{
    public function index(Request $request)
    {
        $users = User::where('role', 'Bidar

        return view('bidan.index', compact(
    }

    public function create()
    {
        return view('bidan.create');
    }
```

Fig. 7. Midwife controller class code

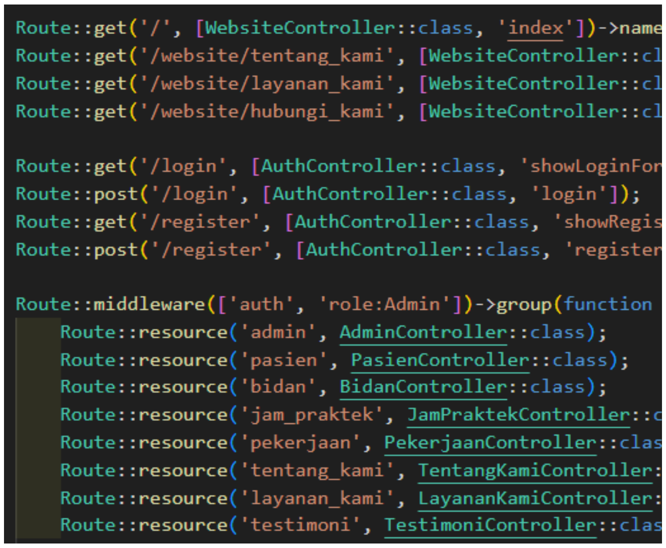
So that the application can be seen by users, an interface is needed to facilitate interaction between the system and users. For the interface creation stage, a view is required. Views in Laravel are used to display data to users in HTML form. View is responsible for rendering the view produced by the controller, which processes data from the model. Views typically contain HTML markup, along with Blade syntax, which is Laravel's template engine that allows writing PHP code in view files in a neater and easier to maintain manner.



```
@extends('components.layout')

@section('content')
    <div class="box">
        <div class="box-header with-border">
            <h3 class="box-title">Daftar Bidan</h3>
        </div>
        <!-- /.box-header -->
        <div class="box-body">
            <div class="mb-4">
                <a href="{{ route('bidan.create') }}" class="btn b
            </div>
            <table class="table table-bordered data-table table-st
                <thead>
                    <tr id="header">
                        <th>No</th>
                        <th>Nama Lengkap</th>
                        <th>Tanggal Lahir</th>
                        <th>Jenis Kelamin</th>
                        <th>No HP</th>
                        <th>Aksi</th>
                    </tr>
                    <tr>
                        <td></td>
                        <td><input class="form-control form-contro
                                placeholder="Cari Nama Lengkap"></
                        <td><input class="form-control form-contro
```

Fig. 8. View class code

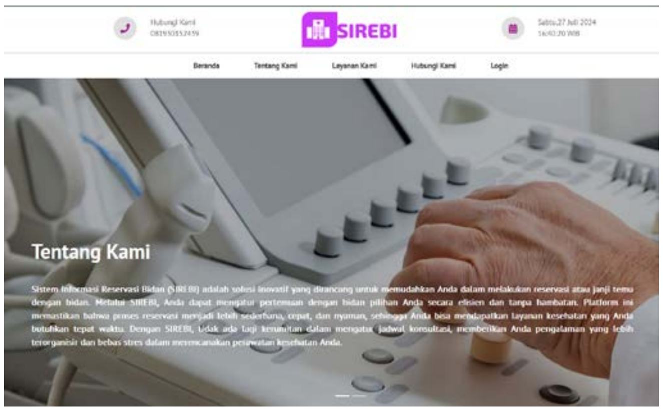
After the model, controller and view classes have been created, the final stage is creating routing. Routing in Laravel is the process of defining how an application responds to various incoming HTTP requests. Routes define URLs that users can access and route those requests to the appropriate controller or function to handle the application logic. Laravel provides a flexible and powerful routing system, allowing developers to easily manage URLs and application responses. In the application, routing is divided into public and private. The public route is intended for users when accessing the website page in the form of general information on the landing page. Meanwhile, private routes are intended for users who have registered in the system to access special menus according to the types of users that have been set in the system.



```
Route::get('/', [WebsiteController::class, 'index'])->name
Route::get('/website/tentang_kami', [WebsiteController::cl
Route::get('/website/layanan_kami', [WebsiteController::cl
Route::get('/website/hubungi_kami', [WebsiteController::cl

Route::get('/login', [AuthController::class, 'showLoginFor
Route::post('/login', [AuthController::class, 'login']);
Route::get('/register', [AuthController::class, 'showRegis
Route::post('/register', [AuthController::class, 'register

Route::middleware(['auth', 'role:Admin'])->group(function
    Route::resource('admin', AdminController::class);
    Route::resource('pasien', PasienController::class);
    Route::resource('bidan', BidanController::class);
    Route::resource('jam_praktek', JamPraktekController::c
    Route::resource('pekerjaan', PekerjaanController::clas
    Route::resource('tentang_kami', TentangKamiController:
    Route::resource('layanan_kami', LayananKamiController:
    Route::resource('testimoni', TestimoniController::clas
```

Fig. 9. Routing code

After all stages have been completed, the next step is to review the application that has been created via the browser. The first page that appears is the landing page.



Tentang Kami

Sistem Informasi Reservasi Bidan (SIREBI) adalah solusi inovatif yang dirancang untuk memudahkan Anda dalam melakukan reservasi atau janji temu dengan bidan. Melalui SIREBI, Anda dapat mengatur pertemuan dengan bidan pilihan Anda secara efisien dan tanpa hambatan. Platform ini memastikan bahwa proses reservasi menjadi lebih sederhana, cepat, dan nyaman, sehingga Anda bisa mendapatkan layanan kesehatan yang Anda butuhkan tepat waktu. Dengan SIREBI, tidak ada lagi kerumitan dalam mengatur jadwal konsultasi, memberikan Anda pengalaman yang lebih terorganisir dan bebas stres dalam merencanakan perawatan kesehatan Anda.

Fig. 10. landing page

Then to access the special page, the user is required to log in. After the login process is complete, the user will be directed to modules that have been adjusted to each user's access rights.

Proceedings of the Informatics Conference Vol. 11 No. 22 (2025): Requirements Engineering Methodologies ISSN: 2477-5894

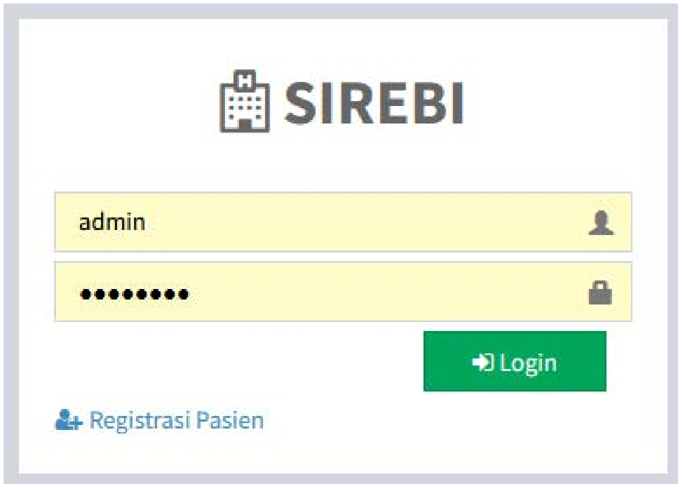


Fig. 11. Login page

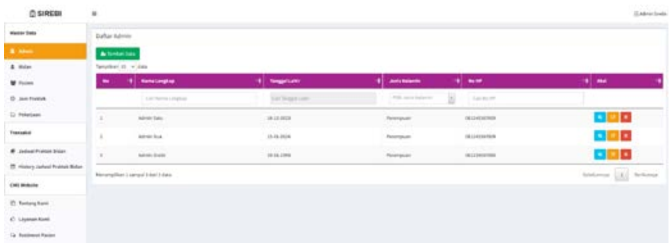


Fig. 12. Admin data management page

Especially for patients, a special menu is provided for registration. This allows patients to easily register patient data without having to come to the midwife first.

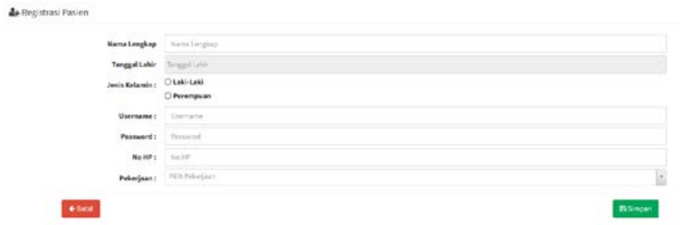


Fig. 13. Patient registration page

D.Software Testing

Black-box testing is a software testing method that focuses exclusively on the application's external behavior, particularly its input and output, regardless of its internal structure. This testing phase is a critical component of the software development lifecycle. By employing black-box testing, researchers can proactively identify and rectify errors or deficiencies within the application.

There are the 15 main test cases made and tested. Mid-developing, these test cases failed. But the result is that no defect or bug was detected for the final result. The test cases can be found in the appendix section.

IV. Conclusion

Our web application significantly enhances the workflow of tocologists and midwives. By automating manual scheduling processes, we optimize time management, enabling healthcare providers to efficiently handle appointments, resource allocation, and event coordination. This streamlined approach leads to better time utilization and ultimately improves patient care.

Furthermore, our application fosters enhanced patient engagement through personalized communication, timely reminders, and relevant information. This empowers patients to feel informed and cared for, motivating them to adhere to their treatment plans and leading to improved health outcomes.

Committed to open collaboration, we share our source code freely to encourage innovation and customization within the healthcare technology sector. This approach allows developers, designers, and healthcare professionals to work together to continually improve the application.

Our development process combines agile and kanban methodologies, ensuring flexibility while maintaining project discipline. Frequent releases enable tocologists and midwives to benefit from incremental improvements, adapting to their evolving needs.

In summary, our specialized web application effectively addresses the challenges faced by tocologists and midwives worldwide. By combining efficient scheduling, enhanced patient engagement, open collaboration, and agile development, we empower healthcare providers to deliver exceptional care.

References

[1] L. U. Oghenekaro. (2023, May). Web-based Integrated Restaurant Management System. *International Journal of Applied Information Systems (IJAIS).* [Online]. *Volume 12– No.40.* Available: https://www.ijais.org/archives/volume12/number40/oghenekaro-2023-ijais-451945.pdf

[2] R. A. Egigogo, M. T. Naniya, A. A. Abubakar, A. Mansir (2024, Apr.). Design and Implementation of Computerized Restaurant Table Booking System. *Ceddi Journal of Information Systems and Technology (JST).* [Online]. *Vol. 3 No. 1.* Available: http://journal.ceddi.id/index.php/jst/article/view/64/53

[3] C. Halkiopoulos, H.. Antonopoulou , D. Papadopoulos , I. Giannoukou (2020, Dec.). Online Reservation Systems in E-Business: Analyzing Decision Making in E-Tourism. *Journal of Tourism, Heritage & Services Marketing (JTHSM).* [Online]. *Vol. 6, No. 1, pp. 9-16.* Available: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3747958

[4] P. Senjani (2024, Feb.). Website-Based Customer Relationship Management Application at Mitra Palembang Clinic. *COMSERVA: Jurnal Penelitian dan Pengabdian Masyarakat.* [Online]. *Volume 3 No. 10.* Available: https://comserva.publikasiindonesia.id/index.php/comserva/article/view/1215/1598

[5] E. C. Daraojimba, C. N. Nwasike, A. O. Adegbite, C. A. Ezeigweneme, and J. O. Gidiagbai (2024, Jan.). COMPREHENSIVE REVIEW OF AGILE METHODOLOGIES IN PROJECT MANAGEMENT. *Computer Science & IT Research Journal.* [Online]. *5(1), pp. 190-218.*. Available: https://fepbl.com/index.php/csitrj/article/view/717

[6] D. O. Ogundipe, O. A.Odejide, and T. E. Edunjobi (2024, Mar.). Agile methodologies in digital banking: Theoretical underpinnings and implications for customer satisfaction. *Open Access Research Journal of Science and Technology.* [Online]. *Volume 10, Issue 1.*. Available: https://oarjst.com/content/agile-methodologies-digital-banking-theoretical-underpinnings-and-implications-customer

[7] A. F. Pratama, O. K. Manurung (2023, Jun.). Web-Based Student Lecture Data Information System Using PHP and MySQL. *Jurnal Pendidikan LLDIKTI Wilayah I (JUDIK).* [Online]. *Volume 03*, *Issue 1*. Available: https://lldikti1.kemdikbud.go.id/jurnal/index.php/judik/article/view/379/152

[8] S. Sotnik, V. Manakov, V. Lyashenko (2023, Jan.). Overview: PHP and MySQL Features for Creating Modern Web Projects. *International Journal of Academic Information Systems Research (IJAISR).* [Online]. *Vol. 7 Issue 1, pp: 11-17*. Available: https://openarchive.nure.ua/handle/document/21601

[9] B. I. Uktamovich and K. R. Mirzarahmon (2023, Apr.). THE ADVANTAGE AND DISADVANTAGES OF USING PHP, JAVASCRIPT TECHNOLOGY IN CREATING THE SERVER PART OF THE PLATFORM. *GALAXY INTERNATIONAL INTERDISCIPLINARY RESEARCH JOURNAL (GIIRJ).* [Online]. *Vol. 11, Issue 04*. Available: https://openarchive.nure.ua/handle/document/21601

[10] M. Rahmadani, R. E. Putra, Y. Fadillah, and S. Saputra (2024, Jun.). Perancangan Sistem Pencatatan Aset Inventaris Pada LKSA Ar-Ridho Berbasis Web Dan Database Mysql. *BIN : Bulletin Of Informatics.* [Online]. *Vol. 2, Issue 01, pp 144-146*. Available: https://ojs.jurnalmahasiswa.com/ojs/index.php/bin/article/view/344

[11] B.D.D Arianti, H. Kuswanto, H. A. Januari, and J. Jamaluddin (2024, Jun.). The design of a letter archiving application using the Model View Controller (MVC) concept. Presented at Conference *BIN : Bulletin Of Informatics.* [Online]. *Vol. 1869*. Available: https://iopscience.iop.org/article/10.1088/1742-6596/1869/1/012083/meta