



Sekitarku.id - Panduan Penggunaan dan Listing Program

disusun oleh: Rozaq.in



CARI TENAGA KESEHATAN DISEKITAR MU



FITUR UTAMA



Pencarian Profesi Kesehatan Berbasis Geolocation

Temukan tenaga medis terdekat dengan cepat dan akurat di peta interaktif



Sistem Booking Layanan

Pesan jadwal konsultasi atau tindakan medis secara online dengan mudah



Analisis Pemetaan Fasilitas Kesehatan Berbasis GIS

Visualisasikan & analisis data kesehatan untuk perencanaan strategis



CUSTOMER (PASIEIN)



Akses
Layanan
Cepat

Booking
Mudah
24/7

Riwayat
Layanan
Terpusat



PENYEDIA (TENAGA KESEHATAN)



Manajemen
Paktik

Perluasan
Jangkauan

Tingkatkan
Visibilitas



BiMA

Mulai Gunakan Sekarang!
Transformasikan Akses Kesehatan Anda.

Kunjungi:
<https://sekitarku.id>

Sekitarku.id - Panduan Penggunaan dan Listing Program

1. Pendahuluan

Perkembangan teknologi informasi dan komunikasi telah mendorong percepatan transformasi digital di berbagai sektor layanan, termasuk sektor jasa dan layanan profesional. Selain itu, masyarakat sering mengalami kesulitan dalam menentukan layanan kesehatan yang sesuai dengan kebutuhan mereka, karena sistem yang tersedia umumnya hanya menyediakan informasi statis tanpa dukungan analisis berbasis kebutuhan pengguna. Teknologi Location-Based Service (LBS) atau geolocation memungkinkan sistem informasi untuk menyediakan layanan berdasarkan posisi geografis pengguna secara real-time.

Integrasi GIS dalam sistem kesehatan mampu memberikan gambaran komprehensif terkait pemerataan fasilitas dan membantu pengambilan keputusan berbasis wilayah. Meskipun demikian, sebagian besar platform layanan kesehatan yang tersedia saat ini masih berfokus pada fungsi direktori atau pencarian layanan tanpa integrasi yang kuat dengan analisis kebutuhan pengguna maupun pendekatan spasial.

Platform tersebut umumnya belum mampu memberikan rekomendasi layanan yang dipersonalisasi, serta belum memanfaatkan potensi analisis GIS untuk mengevaluasi pemerataan fasilitas kesehatan secara sistematis. Oleh karena itu, diperlukan suatu pendekatan yang mampu mengintegrasikan berbagai komponen tersebut dalam satu sistem yang terpadu.

platform berbasis web yang dapat menggabungkan pencarian profesi kesehatan berbasis geolocation, sistem booking layanan, serta analisis pemetaan fasilitas kesehatan berbasis GIS.

Dokumen ini menyediakan panduan langkah-demi-langkah untuk menavigasi dan menggunakan aplikasi sekitarku.id, yang mencakup proses masuk (login) dan fungsionalitas inti untuk peran Pengguna (Customer), Penyedia (Provider), dan Administrator. Setiap bagian menyertakan instruksi dan bantuan visual yang disimulasikan (tangkapan layar) untuk kejelasan.

2. Informasi Login

Harap gunakan kredensial berikut untuk mengakses aplikasi untuk setiap peran yang bersangkutan:

Akun Pelanggan (Customer) :

- **Nama Pengguna (Username) :** customer1@example.com
- **Kata Sandi (Password) :** qwerty
- **Peran (Role) :** Customer

- **Akun Penyedia (Provider) :**

- **Nama Pengguna (Username) :** dr.ahmad.hussain@clinic.com
- **Kata Sandi (Password) :** qwerty
- **Peran (Role) :** Provider

- **Akun Admin:**

- **Nama Pengguna (Username) :** admin@example.com
 - **Kata Sandi (Password) :** qwerty
 - **Peran (Role) :** Admin
-

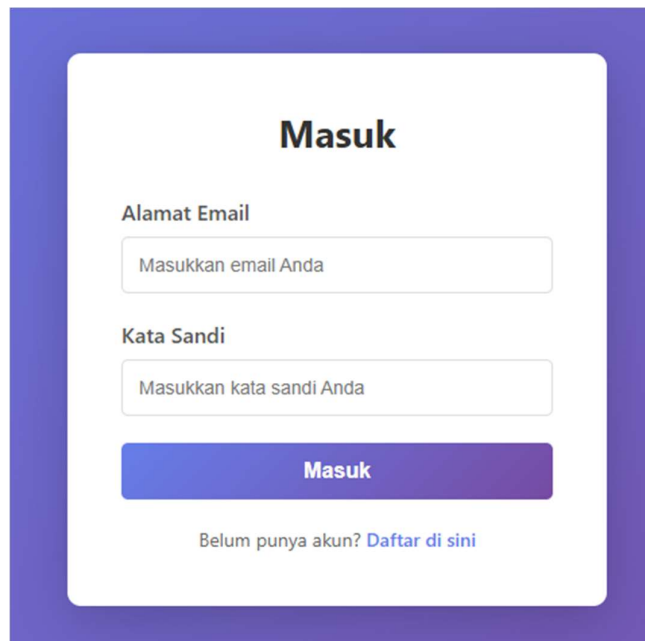
3. Alur Pengguna Pelanggan (Customer) : Mencari dan Memesan Layanan

3.1. Masuk (Login) sebagai Pelanggan

Halaman: Halaman Login (<https://sekitarku.id/auth/login>)

Deskripsi: Ini adalah titik masuk utama untuk semua pengguna. Anda akan memasukkan kredensial pelanggan Anda di sini.

Tangkapan Layar:



Masuk

Alamat Email

Masukkan email Anda

Kata Sandi

Masukkan kata sandi Anda

Masuk

Belum punya akun? [Daftar di sini](#)

Langkah-langkah:

1. **Akses** Halaman Login menggunakan URL: <https://sekitarku.id/auth/login>
2. Di kolom "Alamat Email", **masukkan** customer1@example.com.
3. Di kolom "Kata Sandi", **masukkan** qwerty.
4. **Klik** Tombol Login.

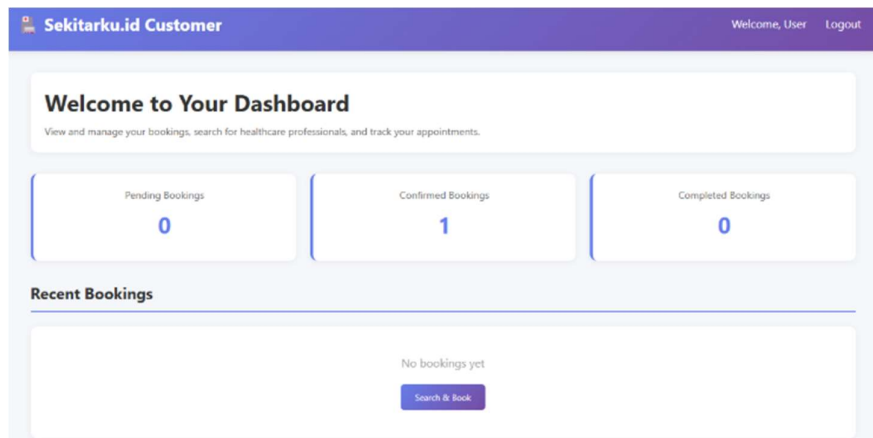
3.2. Menavigasi Dashboard Pelanggan

Halaman: Dashboard Pelanggan

<https://sekitarku.id/customer/dashboard>

Deskripsi: Setelah masuk, Anda akan mendarat di dashboard pribadi Anda, menampilkan ringkasan aktivitas Anda.

Tangkapan Layar:



Langkah-langkah:

1. **Tinjau** jumlah "Pesanan Tertunda" dan "Pesanan Dikonfirmasi" untuk gambaran umum singkat.
2. **Telusuri** tabel "Pesanan Terbaru" untuk melihat janji temu terbaru Anda.
3. Untuk mencari layanan baru, **klik** Tombol Cari Penyedia di bagian "Tindakan Cepat" atau tautan Cari di bilah navigasi.
4. Untuk melihat semua pesanan Anda yang lalu dan yang akan datang, **klik** Tombol Lihat Semua Pesanan atau tautan Pesanan Saya di bilah navigasi.
5. Untuk keluar, **klik** tautan Logout di bilah navigasi.

3.3. Melihat Pesanan Saya

Halaman: Pesanan Saya (<https://sekitarku.id/customer/bookings>)

Deskripsi: Halaman ini menampilkan daftar lengkap semua pesanan layanan Anda.

[Dashboard](#) [Search](#) [Logout](#)

My Bookings

All Statuses ▾

Provider	Service	Date	Time	Status	Action
Dr. Sarah Chen Pediatrics	Newborn Check-up	2026-08-23	15:00:00	Confirmed	View

© 2026 sekitarku.id. All rights reserved.

Langkah-langkah:

1. Untuk memfilter pesanan berdasarkan statusnya, **klik** dropdown `Semua Status` dan **pilih** opsi (misalnya, "Tertunda", "Dikonfirmasi").
2. Untuk melihat informasi detail untuk pesanan tertentu, **klik** Tombol Lihat di kolom "Tindakan" untuk pesanan tersebut.
3. Untuk pesanan yang sudah selesai, Anda mungkin melihat Tombol Ulas; **klik** ini untuk memberikan umpan balik kepada penyedia.
4. Untuk membuat pesanan baru (jika tersedia), **klik** Tombol Pesanan Baru.
5. Untuk menavigasi antar halaman pesanan, **gunakan** tautan paginasi `[Pertama]` `[Sebelumnya]` `[1]` `[2]` `[3]` `[Berikutnya]` `[Terakhir]`.

3.4. Melihat Detail Pesanan dan Mengambil Tindakan

Halaman: Detail Pesanan

`https://sekitarku.id/customer/booking_detail/{booking_id}`

Deskripsi: Halaman ini menyediakan semua informasi spesifik tentang satu pesanan dan memungkinkan Anda untuk melakukan tindakan terkait dengannya.

Tangkapan Layar:

My Bookings

Booking #4

Service: N/A
Date: 2026-08-23
Time: 15:00:00
Status: Confirmed

Provider

Business: Dr. Sarah Chen Pediatrics
Name: N/A

[Back](#)

Langkah-langkah:

1. **Tinjau** bagian "Informasi Pesanan" dan "Informasi Penyedia" untuk semua detail yang relevan.
2. Jika pesanan sudah selesai, **klik** Tombol Beri Ulasan untuk mengirimkan umpan balik Anda.
3. Jika pesanan dalam status tertunda atau dikonfirmasi, dan Anda ingin membatalkannya, **klik** Tombol Batalkan Pesanan.
4. Untuk kembali ke daftar pesanan Anda, **klik** tautan <-- Kembali ke Pesanan atau Tombol Kembali ke Pesanan.

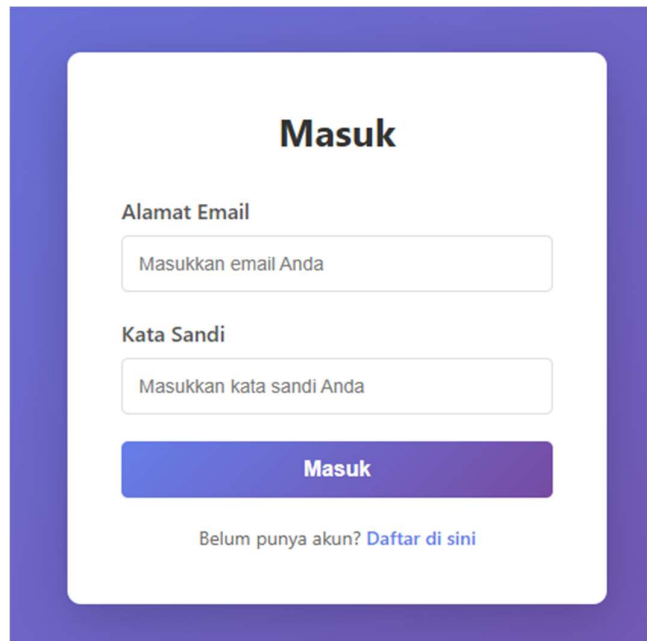
4. Alur Pengguna Penyedia (Provider): Mengelola Layanan dan Pesanan

4.1. Masuk (Login) sebagai Penyedia

Halaman: Halaman Login (<https://sekitarku.id/auth/login>)

Deskripsi: Gunakan halaman ini untuk masuk dengan kredensial penyedia Anda.

Tangkapan Layar:

A login form titled "Masuk" (Login) with a purple border. It contains two input fields: "Alamat Email" (Email Address) and "Kata Sandi" (Password). Below the fields is a purple "Masuk" button. At the bottom, there is a link "Belum punya akun? Daftar di sini" (Don't have an account? Sign up here).

Langkah-langkah:

1. **Akses** Halaman Login: <https://sekitarku.id/auth/login>
2. Di kolom "Alamat Email", **masukkan** dr.ahmad.hussain@clinic.com.
3. Di kolom "Kata Sandi", **masukkan** `qwerty`.
4. **Klik** Tombol Login.

4.2. Menavigasi Dashboard Penyedia

Halaman: Dashboard Penyedia
<https://sekitarku.id/profesi/dashboard>

Deskripsi: Pusat utama Anda sebagai penyedia, menampilkan statistik penting dan aktivitas terbaru.

Langkah-langkah:

1. **Tinjau** statistik "Pesanan Tertunda" dan "Pesanan Dikonfirmasi".
2. **Periksa** tabel "Pesanan Terbaru" dan "Ulasan Terbaru" untuk interaksi terbaru Anda.
3. Untuk mengelola semua pesanan Anda, **klik** Tombol Lihat Semua Pesanan atau tautan Pesanan di bilah navigasi.
4. Untuk mengelola layanan yang Anda tawarkan, **klik** Tombol Lihat Semua Layanan atau tautan Layanan Saya di bilah navigasi.
5. Untuk melihat semua ulasan pelanggan, **klik** tautan Ulasan di bilah navigasi.

6. Untuk keluar, **klik** tautan Logout.

4.3. Mengelola Pesanan Penyedia

Halaman: Pesanan Saya (Penyedia)
<https://sekitarku.id/profesi/bookings>

Deskripsi: Halaman ini menampilkan semua pesanan untuk layanan Anda, memungkinkan Anda mengelola statusnya.

Langkah-langkah:

1. Untuk memfilter pesanan, **pilih** status dari dropdown Semua Status.
2. Untuk melihat detail pesanan, **klik** Tombol Lihat di kolom "Tindakan" pesanan tersebut.
3. Untuk pesanan tertunda, **klik** Konfirmasi untuk menyetujui pesanan atau Batalkan untuk menolaknya.
4. Untuk pesanan yang dikonfirmasi, **klik** Selesaikan setelah layanan diberikan, atau Batalkan jika perlu.
5. **Gunakan** tautan paginasi untuk menelusuri beberapa halaman pesanan.

4.4. Mengelola Detail Pesanan Penyedia

Halaman: Detail Pesanan (Penyedia)
https://sekitarku.id/profesi/booking_detail/{booking_id}

Deskripsi: Halaman ini menyediakan detail mendalam tentang satu pesanan tertentu dan memungkinkan Anda untuk memperbarui statusnya.

Langkah-langkah:

1. **Tinjau** bagian "Informasi Pesanan" dan "Informasi Pelanggan".
2. Berdasarkan status pesanan saat ini:
 - o Jika Tertunda, **klik** Tombol Konfirmasi Pesanan untuk menerima atau Tombol Batalkan Pesanan untuk menolak.
 - o Jika Dikonfirmasi, **klik** Tombol Selesaikan Pesanan setelah pengiriman layanan atau Tombol Batalkan Pesanan jika diperlukan.
2. Untuk kembali ke daftar pesanan, **klik** tautan <-- Kembali ke Pesanan atau Tombol Kembali ke Pesanan.

4.5. Mengelola Layanan Saya

Halaman: Layanan Saya (<https://sekitarku.id/profesi/services>)

Deskripsi: Halaman ini menampilkan semua layanan yang Anda berikan, memungkinkan Anda untuk menambah, mengedit, atau menghapusnya.

Langkah-langkah:

1. Untuk menambahkan layanan baru, **klik** Tombol Tambah Layanan Baru.
2. Untuk melihat detail layanan yang ada, **klik** Lihat di kolom "Tindakan" layanan tersebut.
3. Untuk mengubah layanan, **klik** Edit di kolom "Tindakan" layanan tersebut.
4. Untuk menghapus layanan, **klik** Hapus di kolom "Tindakan" layanan tersebut (konfirmasi mungkin akan muncul).
5. **Gunakan** tautan paginasi untuk menavigasi layanan jika Anda memiliki banyak.

4.6. Menambah Layanan Baru

Halaman: Tambah Layanan
(https://sekitarku.id/profesi/add_service)

Deskripsi: Sebuah formulir untuk membuat dan menerbitkan layanan baru.

Langkah-langkah:

1. **Masukkan** Nama Layanan di kolom input yang tersedia.
2. **Pilih** Kategori dari daftar dropdown.
3. **Masukkan** Harga untuk layanan tersebut.
4. **Berikan** Deskripsi rinci tentang layanan di textarea.
5. **Klik** Tombol Simpan Layanan untuk menambahkan layanan baru Anda.
6. Untuk kembali tanpa menyimpan, **klik** tautan <-- Kembali ke Layanan.

4.7. Mengedit Layanan yang Ada

Halaman: Edit Layanan
(https://sekitarku.id/profesi/edit_service/{service_id})

Deskripsi: Sebuah formulir untuk mengubah detail layanan Anda yang sudah ada.

Langkah-langkah:

1. **Ubah** kolom yang sudah terisi sebelumnya (Nama Layanan, Kategori, Harga, Deskripsi) sesuai kebutuhan.
2. **Klik** Tombol Perbarui Layanan untuk menyimpan perubahan Anda.
3. Untuk membatalkan perubahan dan kembali, **klik** tautan <-- Kembali ke Layanan.

4.8. Melihat Ulasan Pelanggan

Halaman: Ulasan (<https://sekitarku.id/profesi/reviews>)

Deskripsi: Halaman ini menampilkan semua ulasan dan peringkat pelanggan untuk layanan Anda.

Langkah-langkah:

1. **Tinjau** Rating Rata-rata dan jumlah total ulasan.
2. **Baca** ulasan individual di "Tabel Ulasan" untuk memahami umpan balik pelanggan.
3. **Gunakan** tautan paginasi untuk menavigasi semua ulasan.

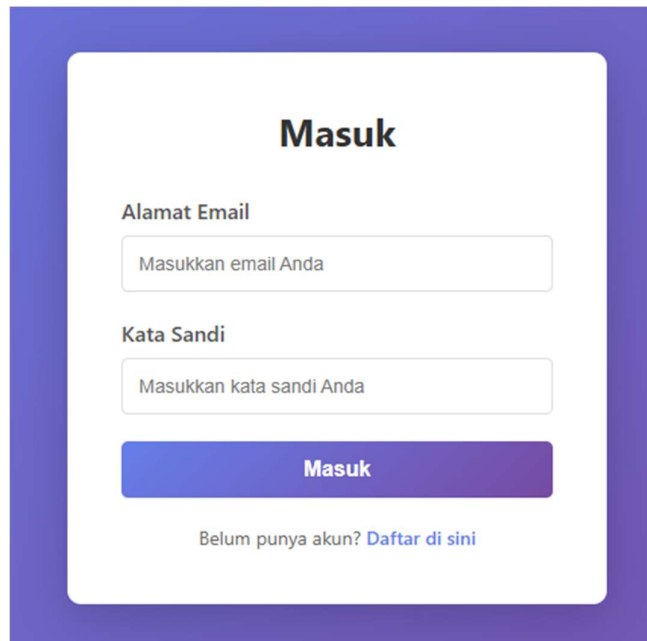
5. Alur Pengguna Admin: Manajemen Platform

5.1. Masuk (Login) sebagai Administrator

Halaman: Halaman Login (<https://sekitarku.id/auth/login>)

Deskripsi: Akses backend administratif menggunakan kredensial admin Anda.

Tangkapan Layar:



Masuk

Alamat Email

Masukkan email Anda

Kata Sandi

Masukkan kata sandi Anda

Masuk

Belum punya akun? [Daftar di sini](#)

Langkah-langkah:

1. **Akses** Halaman Login: <https://sekitarku.id/auth/login>
2. Di kolom "Alamat Email", **masukkan** at.ozkula@gmail.com.
3. Di kolom "Kata Sandi", **masukkan** `qwerty`.
4. **Klik** Tombol Login.

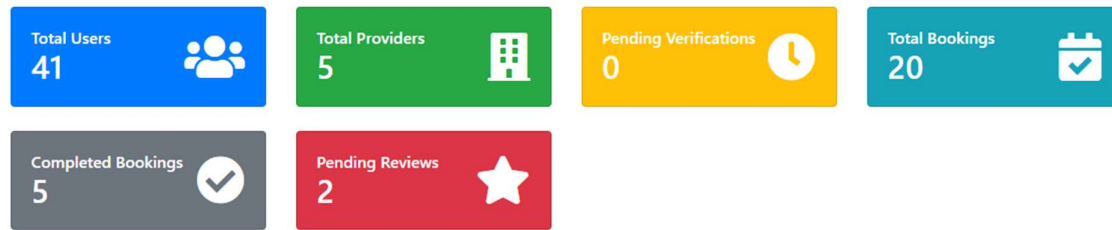
5.2. Menavigasi Dashboard Admin

Halaman: Dashboard Admin
(<https://sekitarku.id/admin/dashboard>)

Deskripsi: Panel kontrol pusat untuk seluruh platform sekitarku.id, menampilkan metrik utama dan akses ke alat manajemen.

Tangkapan Layar:

Dashboard



Recent Users				Recent Bookings				
Name	Email	Role	Register	Customer	Provider	Service	Status	Date
Yanto Hermawan	pharmacist.yanto@pharmacy.com	service_provider	2026-08-25 23:05:44	Rozaq Isnaini Nugraha	XOTh	BBB	confirmed	2026-08-14 22:50:09
Jane Smith	jane.smith@email.com	customer	2026-08-25 23:05:44	Rozaq Isnaini Nugraha	Dr. Emily White Clinic	Medical Check-up	pending	2026-08-14 22:50:09
John Doe	john.doe@email.com	customer	2026-08-25	Admin User	Lee Nursing Services	Post-operative	completed	2026-08-14

Langkah-langkah:

1. **Tinjau** statistik ringkasan untuk Total Pengguna, Total Penyedia, Total Pesanan, dan Ulasan Tertunda.
2. **Periksa** tabel "Aktivitas Terbaru" untuk log peristiwa platform terbaru.
3. Untuk mengelola aspek-aspek tertentu dari platform, **gunakan** tautan navigasi di header:
 - o Pengguna: Untuk mengelola semua akun pelanggan dan admin.
 - o Penyedia: Untuk mengelola profil penyedia layanan.
 - o Pesanan: Untuk mengawasi semua pesanan di platform.
 - o Ulasan: Untuk memoderasi ulasan pelanggan.
 - o Kategori: Untuk mengelola kategori layanan.
 - o Jenis Bisnis: Untuk mengelola jenis-jenis bisnis.
 - o Logout: Untuk keluar dari panel admin.

5.3. Mengelola Pengguna

Halaman: Kelola Pengguna (<https://sekitarku.id/admin/users>)

Deskripsi: Halaman ini memungkinkan administrator untuk melihat, mengedit, dan mengelola semua akun pengguna dalam sistem, termasuk pelanggan, penyedia, dan admin lainnya.

Tangkapan Layar:

Manage Users

[Add New User](#)Filter by Role: All

ID	Name	Email	Role	Status	Registered	Actions
1	Rozaq Isnaini Nugraha	at.ozkula@gmail.com	admin	active	2026-07-17 16:42:37	View Edit Delete Deactivate
4	Admin User	admin@example.com	admin	active	2026-08-14 22:50:08	View Edit Delete Deactivate
3	Rozaq Isnaini Nugraha	CC@uwhs.ac.id	customer	active	2026-07-19 08:27:49	View Edit Delete Deactivate
5	Alice Smith	customer1@example.com	customer	active	2026-08-14 22:50:08	View Edit Delete Deactivate
6	Bob Johnson	customer2@example.com	customer	active	2026-08-14 22:50:08	View Edit Delete Deactivate
7	Charlie Brown	customer3@example.com	customer	active	2026-08-14 22:50:08	View Edit Delete Deactivate
8	Diana Prince	customer4@example.com	customer	active	2026-08-14 22:50:08	View Edit Delete Deactivate
9	Eve Adams	customer5@example.com	customer	active	2026-08-14 22:50:08	View Edit Delete Deactivate
10	Frank White	customer6@example.com	customer	active	2026-08-14 22:50:08	View Edit Delete Deactivate

Langkah-langkah:

1. Untuk memfilter daftar pengguna, **pilih** peran dari dropdown Filter berdasarkan Peran.
2. Untuk melihat detail pengguna, **klik** Lihat di kolom "Tindakan".
3. Untuk mengubah informasi pengguna, **klik** Edit.
4. Untuk menonaktifkan atau mengaktifkan akun pengguna, **klik** Nonaktifkan (atau Aktifkan, jika berlaku).
5. **Gunakan** tautan paginasi untuk menelusuri daftar pengguna.

5.4. Mengelola Penyedia

Halaman: Kelola Penyedia
(<https://sekitarku.id/admin/providers>)

Deskripsi: Halaman ini memungkinkan administrator untuk mengelola profil penyedia layanan, termasuk status verifikasi mereka.

Langkah-langkah:

1. Untuk memfilter penyedia, **pilih** status dari dropdown Filter berdasarkan Status Verifikasi.
2. Untuk melihat profil lengkap penyedia, **klik** Lihat.
3. Untuk mengedit informasi penyedia, **klik** Edit.

4. Untuk mengubah status aktivasi penyedia, **klik** Nonaktifkan (atau Aktifkan).
5. Untuk penyedia yang belum terverifikasi, **klik** Verifikasi untuk menyetujui akun mereka.
6. **Gunakan** tautan paginasi untuk menavigasi daftar penyedia.

5.5. Mengelola Pesanan (Admin)

Halaman: Kelola Pesanan (<https://sekitarku.id/admin/bookings>)

Deskripsi: Halaman ini menyediakan gambaran umum yang komprehensif tentang semua pesanan di platform, memungkinkan administrator untuk memantau dan mengelolanya.

Langkah-langkah:

1. Untuk memfilter pesanan berdasarkan statusnya saat ini, **pilih** opsi dari dropdown Filter berdasarkan Status.
2. Untuk melihat detail lengkap pesanan apa pun, **klik** Lihat di kolom "Tindakan" nya.
3. Untuk membatalkan pesanan (jika diperlukan), **klik** Batalkan.
4. **Gunakan** tautan paginasi untuk menelusuri semua pesanan.

5.6. Mengelola Ulasan (Admin)

Halaman: Kelola Ulasan (<https://sekitarku.id/admin/reviews>)

Deskripsi: Halaman ini memungkinkan administrator untuk memoderasi dan mengelola semua ulasan pelanggan yang dikirimkan untuk layanan.

Langkah-langkah:

1. Untuk memfilter ulasan berdasarkan status moderasi mereka, **pilih** opsi dari dropdown Filter berdasarkan Status.
2. Untuk melihat detail lengkap ulasan, **klik** Lihat.
3. Untuk ulasan tertunda, **klik** Setujui untuk menerbitkannya atau Tolak untuk menolaknya.
4. Untuk ulasan yang disetujui, Anda masih dapat Tolak jika perlu.
5. **Gunakan** tautan paginasi untuk menelusuri semua ulasan.

5.7. Mengelola Kategori

Halaman: Kelola Kategori
(<https://sekitarku.id/admin/categories>)

Deskripsi: Halaman ini memungkinkan administrator untuk menentukan dan mengelola kategori layanan yang tersedia di platform.

Langkah-langkah:

1. Untuk menambahkan kategori baru, **klik** Tombol Tambah Kategori Baru.
2. Untuk mengubah kategori yang sudah ada, **klik** Edit di kolom "Tindakan" nya.
3. Untuk menghapus kategori, **klik** Hapus.

5.8. Mengelola Jenis Bisnis

Halaman: Kelola Jenis Bisnis
(https://sekitarku.id/admin/business_types)

Deskripsi: Halaman ini memungkinkan administrator untuk menentukan dan mengelola jenis bisnis yang dapat didaftarkan oleh penyedia.

Langkah-langkah:

1. Untuk menambahkan jenis bisnis baru, **klik** Tombol Tambah Jenis Bisnis Baru.
2. Untuk mengubah jenis bisnis yang sudah ada, **klik** Edit di kolom "Tindakan" nya.
3. Untuk menghapus jenis bisnis, **klik** Hapus.

Program Aplikasi
Controller/Guest.php

```
<?php
```

```
defined('BASEPATH') OR exit('No direct script access  
allowed');
```

```
class Guest extends CI_Controller {
```

```
    public function __construct() {
```

```
        parent::__construct();
```

```
        $this->load->model('Booking_model');
```

```
        $this->load->model('ServiceProvider_model');
```

```
        $this->load->model('Category_model');
```

```
        $this->load->helper('geolocation');
```

```
        $this->load->helper('url');
```

```
        $this->load->library('session');
```

```
    }
```

```
    /**
```

```
     * Homepage / Search
```

```
     */
```

```
    public function index() {
```

```
        $data['categories'] = $this->Category_model->  
>get_all();
```

```
        // Fetch some providers to display on the map on the  
homepage
```

```
        // Using default coordinates (e.g., Jakarta) and a  
broad radius for initial display
```

```
        $default_lat = -6.966667; // Example: Jakarta latitude
```

```
        $default_lon = 110.416664; // Example: Jakarta  
longitude
```

```
        $default_radius = 50; // Broad radius to show some
providers
        $limit = 10; // Limit to 10 providers for the homepage
map
```

```
        $data['providers'] = $this->Booking_model-
>find_nearest_providers(
            $default_lat,
            $default_lon,
            $default_radius,
            $limit,
            0 // offset
        );
```

```
        // Pass session data to view
        $data['is_logged_in'] = $this->session-
>userdata('user_id') ? TRUE : FALSE;
        $data['user_role'] = $this->session-
>userdata('user_role');
        $data['user_name'] = $this->session-
>userdata('user_name');

        $this->load->view('guest/index', $data);
    }
```

```
/**
```

```
 * Search results
```

```
 */
```

```
public function search() {
    $keyword = $this->input->get('q');
    $category_id = $this->input->get('category');
    $city = $this->input->get('city');
```

```

$user_lat = $this->input->get('lat');
$user_lon = $this->input->get('lon');
$radius_km = $this->input->get('radius', 10);
$page = $this->input->get('page', 1);

$limit = 20;

$offset = max(0, ($page - 1) * $limit); // Ensure
offset is not negative

// Build filters
$filters = [
    'keyword' => $keyword,
    'category_id' => $category_id,
    'city' => $city,
    'user_lat' => $user_lat,
    'user_lon' => $user_lon,
    'radius_km' => $radius_km
];

// Search

$results = $this->Booking_model-
>search_with_filters($filters, $limit, $offset);

// Get total for pagination
$total = $this->Booking_model->search_count($filters);

$data = [
    'providers' => $results,
    'total' => $total,
    'page' => $page,

```

```

        'limit' => $limit,
        'total_pages' => ceil($total / $limit),
        'categories' => $this->Category_model->get_all(),
        'filters' => $filters
    ];

    $this->load->view('guest/search_results', $data);
}

/**
 * View provider profile
 */
public function provider($provider_id) {
    $provider = $this->ServiceProvider_model->get_full_profile($provider_id);

    if (!$provider || $provider->verification_status !==
'verified') {
        show_404();
    }

    $this->load->model('Review_model');
    $this->load->model('Service_model');

    $data = [
        'provider' => $provider,
        'reviews' => $this->Review_model->get_provider_reviews($provider_id, 5),
        'average_rating' => $this->Review_model->get_average_rating($provider_id),
        'services' => $this->Service_model->get_provider_services($provider_id, 10),
    ];
}

```

```

        'total_reviews' => $this->Review_model-
>count_provider_reviews($provider_id)
    ];

    $this->load->view('guest/provider_profile', $data);
}

/**
 * Auto-complete search (AJAX)
 */
public function autocomplete() {
    $q = $this->input->get('q');

    if (strlen($q) < 2) {
        echo json_encode([]);
        return;
    }

    $results = $this->Booking_model-
>search_providers_keyword($q, 10);

    $output = [];
    foreach ($results as $provider) {
        $output[] = [
            'id' => $provider->id,
            'name' => $provider->business_name,
            'type' => $provider->business_type,
            'city' => $provider->city
        ];
    }
}

```

```

        header('Content-Type: application/json');
        echo json_encode($output);
    }

    /**
     * Get categories (AJAX)
     */
    public function get_categories() {
        $categories = $this->Category_model->get_all();

        header('Content-Type: application/json');
        echo json_encode($categories);
    }

    /**
     * Geolocation detection (AJAX)
     */
    public function detect_location() {
        // Note: Actual geolocation requires browser API or
        IP-based service

        // This endpoint returns success for client-side
        implementation

        $response = [
            'success' => true,
            'message' => 'Use browser Geolocation API for
precise location'
        ];

        header('Content-Type: application/json');
        echo json_encode($response);
    }

```

```

    }
}
?>
Program Model/Booking_model.php

<?php

defined('BASEPATH') OR exit('No direct script access
allowed');

class Booking_model extends MY_Model {

    protected $table = 'bookings';
    protected $primary_key = 'id';

    public function __construct() {
        parent::__construct();
    }

    /**
     * Find nearest service providers using Haversine formula
     * Returns providers within specified radius of user
location
     *
     * @param float $user_lat - User's latitude
     * @param float $user_lon - User's longitude
     * @param int $radius_km - Search radius in kilometers
(default: 10km)
     * @param int $limit - Number of results to return
     * @param int $offset - Pagination offset
     * @return array Array of nearby providers with distance
     */

```

```

    public function find_nearest_providers($user_lat,
    $user_lon, $radius_km = 10, $limit = 20, $offset = 0) {

        $radius_earth_km = 6371;

        // Haversine formula for distance calculation
        $query = "
            SELECT
                sp.id,
                sp.user_id,
                sp.business_name,
                sp.business_type_id,
                sp.latitude,
                sp.longitude,
                sp.address,
                sp.city,
                u.full_name,
                u.phone,
                u.email,
                u.profile_photo_url,
                bt.name AS business_type,
                c.name AS category,
                (
                    {$radius_earth_km} * ACOS(
                        COS(RADIANS({$user_lat})) *
                        COS(RADIANS(sp.latitude)) *
                        COS(RADIANS({$user_lon}) -
RADIANS(sp.longitude)) +
                        SIN(RADIANS({$user_lat})) *
                        SIN(RADIANS(sp.latitude))
                    )
                )
        
```

```

        ) AS distance_km,
        COALESCE(AVG(r.rating), 0) AS average_rating,
        COUNT(DISTINCT r.id) AS total_reviews
    FROM service_provider_profiles sp
    INNER JOIN users u ON sp.user_id = u.id
    INNER JOIN business_types bt ON
sp.business_type_id = bt.id
    LEFT JOIN categories c ON bt.category_id = c.id
    LEFT JOIN reviews r ON sp.id = r.provider_id AND
r.deleted_at IS NULL AND r.is_approved = 1
    WHERE
        sp.verification_status = 'verified'
        AND u.role_id = (SELECT id FROM roles WHERE
name = 'service_provider')
        AND u.status = 'active'
        AND sp.deleted_at IS NULL
        AND u.deleted_at IS NULL
    GROUP BY sp.id
    HAVING distance_km <= {$radius_km}
    ORDER BY distance_km ASC, average_rating DESC
    LIMIT {$limit} OFFSET {$offset}
";

```

```

    return $this->db->query($query)->result();
}

```

```

/**
 * Count nearest providers for pagination
 *
 * @param float $user_lat - User's latitude
 * @param float $user_lon - User's longitude

```

```

* @param int $radius_km - Search radius in kilometers
* @return int Total count of providers within radius
*/

public function count_nearest_providers($user_lat,
$user_lon, $radius_km = 10) {
    $radius_earth_km = 6371;

    $query = "
        SELECT COUNT(DISTINCT sp.id) as total
        FROM service_provider_profiles sp
        INNER JOIN users u ON sp.user_id = u.id
        WHERE
            sp.verification_status = 'verified'
            AND u.role_id = (SELECT id FROM roles WHERE
name = 'service_provider')
            AND u.status = 'active'
            AND sp.deleted_at IS NULL
            AND u.deleted_at IS NULL
            AND (
                {$radius_earth_km} * ACOS(
                    COS(RADIANS({$user_lat})) *
                    COS(RADIANS(sp.latitude)) *
                    COS(RADIANS({$user_lon}) -
RADIANS(sp.longitude)) +
                    SIN(RADIANS({$user_lat})) *
                    SIN(RADIANS(sp.latitude))
                )
            ) <= {$radius_km}
    ";

    $result = $this->db->query($query)->row();

```

```

        return $result->total;
    }

/**
 * Count search results for pagination
 *
 * @param array $filters - Filter parameters
 * @return int Total count
 */
public function search_count($filters = array()) {
    $radius_earth_km = 6371;
    $user_lat = $filters['user_lat'] ?? 0;
    $user_lon = $filters['user_lon'] ?? 0;
    $radius_km = $filters['radius_km'] ?? 10;

    $this->db->select("COUNT(DISTINCT sp.id) as total");
    $this->db->from('service_provider_profiles sp');
    $this->db->join('users u', 'sp.user_id = u.id');
    $this->db->join('business_types bt',
'sp.business_type_id = bt.id');
    $this->db->join('categories c', 'bt.category_id =
c.id', 'left');

    $this->db->where('sp.verification_status',
'verified');
    $this->db->where('u.status', 'active');
    $this->db->where('sp.deleted_at IS NULL', NULL,
FALSE);
    $this->db->where('u.deleted_at IS NULL', NULL, FALSE);

    if (isset($filters['business_type_id'])) {

```

```

        $this->db->where('sp.business_type_id',
$filters['business_type_id']);
    }

    if (isset($filters['category_id'])) {
        $this->db->where('c.id', $filters['category_id']);
    }

    if (isset($filters['city'])) {
        $this->db->where('sp.city', $filters['city']);
    }

    if (isset($filters['keyword'])) {
        $this->db->group_start();

        $this->db->like('sp.business_name',
$filters['keyword']);

        $this->db->or_like('u.full_name',
$filters['keyword']);

        $this->db->group_end();
    }

    // Distance filter
    if (!empty($user_lat) && !empty($user_lon)) {
        $distance_formula = "({$radius_earth_km} * ACOS(
            COS(RADIANS({$user_lat})) *
            COS(RADIANS(sp.latitude)) *
            COS(RADIANS({$user_lon}) -
RADIANS(sp.longitude)) +
            SIN(RADIANS({$user_lat})) *
            SIN(RADIANS(sp.latitude))
        ))";
    }

```

```

        $this->db->where("{ $distance_formula } <=",
$radius_km, FALSE);
    }

    return $this->db->get()->row()->total;
}

/**
 * Search providers by category, location, and filters
with pagination
 *
 * @param array $filters - Filter parameters
 *
 *     - business_type_id: Filter by business type
 *
 *     - category_id: Filter by category
 *
 *     - city: Filter by city
 *
 *     - keyword: Search by business name
 *
 *     - radius_km: Search radius
 *
 *     - user_lat: User latitude for distance
calculation
 *
 *     - user_lon: User longitude for distance
calculation
 *
 * @param int $limit - Results per page
 *
 * @param int $offset - Pagination offset
 *
 * @return array Array of providers matching filters
 */

public function search_with_filters($filters = array(),
$limit = 20, $offset = 0) {
    $radius_earth_km = 6371;
    $user_lat = $filters['user_lat'] ?? 0;
    $user_lon = $filters['user_lon'] ?? 0;
    $radius_km = $filters['radius_km'] ?? 10;

```

```

$select = "
    sp.id,
    sp.user_id,
    sp.business_name,
    sp.latitude,
    sp.longitude,
    sp.address,
    sp.city,
    u.full_name,
    u.phone,
    u.email,
    u.profile_photo_url,
    bt.id AS business_type_id,
    bt.name AS business_type,
    c.id AS category_id,
    c.name AS category,
    COALESCE(AVG(r.rating), 0) AS average_rating,
    COUNT(DISTINCT r.id) AS total_reviews
";

if (!empty($user_lat) && !empty($user_lon)) {
    $select .= ", ({ $radius_earth_km} * ACOS(
        COS(RADIANS({ $user_lat})) *
        COS(RADIANS(sp.latitude)) *
        COS(RADIANS({ $user_lon}) -
RADIANS(sp.longitude)) +
        SIN(RADIANS({ $user_lat})) *
        SIN(RADIANS(sp.latitude))
    )) AS distance_km";
}

```

```

    } else {
        $select .= ", 0 AS distance_km";
    }

    $this->db->select($select, FALSE);

    $this->db->from('service_provider_profiles sp');
    $this->db->join('users u', 'sp.user_id = u.id');
    $this->db->join('business_types bt',
'sp.business_type_id = bt.id');
    $this->db->join('categories c', 'bt.category_id =
c.id', 'left');
    $this->db->join('reviews r', 'sp.id = r.provider_id
AND r.deleted_at IS NULL AND r.is_approved = 1', 'left');

    // Verification and status filters
    $this->db->where('sp.verification_status',
'verified');
    $this->db->where('u.status', 'active');
    $this->db->where('sp.deleted_at IS NULL', NULL,
FALSE);
    $this->db->where('u.deleted_at IS NULL', NULL, FALSE);

    // Apply optional filters
    if (isset($filters['business_type_id'])) {
        $this->db->where('sp.business_type_id',
$filters['business_type_id']);
    }

    if (isset($filters['category_id'])) {
        $this->db->where('c.id', $filters['category_id']);
    }

```

```

        if (isset($filters['city'])) {
            $this->db->where('sp.city', $filters['city']);
        }

        if (isset($filters['keyword'])) {
            $this->db->group_start();

            $this->db->like('sp.business_name',
$filters['keyword']);

            $this->db->or_like('u.full_name',
$filters['keyword']);

            $this->db->group_end();
        }

        // Group by and having clause for distance filter
        $this->db->group_by('sp.id, u.id, bt.id, c.id');

        if (!empty($user_lat) && !empty($user_lon)) {
            $this->db->having("distance_km <=", $radius_km);
            $this->db->order_by('distance_km', 'ASC');
        }

        $this->db->order_by('average_rating', 'DESC');
        $this->db->limit($limit, $offset);

        return $this->db->get()->result();
    }

    /**
     * Create new booking

```

```

*
* @param array $booking_data - Booking information
* @return int Booking ID or FALSE on failure
*/

public function create_booking($booking_data) {
    // Validate required fields

    if (empty($booking_data['customer_id']) ||
empty($booking_data['provider_id']) ||
        empty($booking_data['service_id'])) {
        return FALSE;
    }

    // Booking reference will be auto-generated by trigger
    return $this->insert($booking_data);
}

/**
* Get booking by ID with full details
*
* @param int $booking_id - Booking ID
* @return object Booking with provider and service
details
*/

public function get_booking_details($booking_id) {
    $this->db->select('
        b.*,
        c.full_name as customer_name,
        c.phone as customer_phone,
        c.email as customer_email,
        sp.business_name,

```

```

        sp.latitude,
        sp.longitude,
        sp.address as provider_address,
        u.full_name as provider_name,
        u.phone as provider_phone,
        s.service_name,
        s.price,
        s.duration_minutes
    ');

    $this->db->from('bookings b');
    $this->db->join('users c', 'b.customer_id = c.id');
    $this->db->join('service_provider_profiles sp',
'b.provider_id = sp.id');

    $this->db->join('users u', 'sp.user_id = u.id');
    $this->db->join('services s', 'b.service_id = s.id');
    $this->db->where('b.id', $booking_id);
    $this->db->where('b.deleted_at', NULL);

    return $this->db->get()->row();
}

```

```

/**
 * Get bookings for customer
 *
 * @param int $customer_id - Customer user ID
 * @param string $status - Filter by status (optional)
 * @param int $limit - Results limit
 * @param int $offset - Pagination offset
 * @return array Array of bookings
 */

```

```

    public function get_customer_bookings($customer_id,
    $status = NULL, $limit = 20, $offset = 0) {

        $this->db->select('b.*, sp.business_name, u.full_name
as provider_name, s.service_name');

        $this->db->from('bookings b');

        $this->db->join('service_provider_profiles sp',
'b.provider_id = sp.id');

        $this->db->join('users u', 'sp.user_id = u.id');

        $this->db->join('services s', 'b.service_id = s.id');

        $this->db->where('b.customer_id', $customer_id);

        $this->db->where('b.deleted_at', NULL);

        if ($status) {

            $this->db->where('b.status', $status);

        }

        $this->db->order_by('b.booking_date DESC,
b.booking_time DESC');

        $this->db->limit($limit, $offset);

        return $this->db->get()->result();

    }

/**
 * Get bookings for service provider
 *
 * @param int $provider_id - Provider profile ID
 * @param string $status - Filter by status (optional)
 * @param int $limit - Results limit
 * @param int $offset - Pagination offset
 * @return array Array of bookings

```

```

*/

public function get_provider_bookings($provider_id,
$status = NULL, $limit = 20, $offset = 0) {

    $this->db->select('b.*, c.full_name as customer_name,
c.phone as customer_phone, s.service_name');

    $this->db->from('bookings b');

    $this->db->join('users c', 'b.customer_id = c.id');

    $this->db->join('services s', 'b.service_id = s.id');

    $this->db->where('b.provider_id', $provider_id);

    $this->db->where('b.deleted_at', NULL);

    if ($status) {

        $this->db->where('b.status', $status);

    }

    $this->db->order_by('b.booking_date DESC,
b.booking_time DESC');

    $this->db->limit($limit, $offset);

    return $this->db->get()->result();

}

/**
 * Update booking status
 *
 * @param int $booking_id - Booking ID
 * @param string $new_status - New status
 * @param string $notes - Optional notes
 * @return boolean Update success
 */

```

```

    public function update_status($booking_id, $new_status,
$notes = NULL) {

        $data = array(

            'status' => $new_status,

            'updated_at' => date('Y-m-d H:i:s')

        );

        if ($notes) {

            $data['provider_notes'] = $notes;

        }

        $this->db->where('id', $booking_id);
        return $this->db->update($this->table, $data);

    }

/**
 * Cancel booking with reason
 *
 * @param int $booking_id - Booking ID
 * @param string $reason - Cancellation reason
 * @return boolean Cancel success
 */

    public function cancel_booking($booking_id, $reason =
NULL) {

        $data = array(

            'status' => 'cancelled',

            'cancellation_reason' => $reason,

            'updated_at' => date('Y-m-d H:i:s')

        );

```

```

        $this->db->where('id', $booking_id);
        return $this->db->update($this->table, $data);
    }

/**
 * Check if booking can be reviewed (must be completed)
 *
 * @param int $booking_id - Booking ID
 * @return boolean Can be reviewed
 */
public function can_be_reviewed($booking_id) {
    $this->db->select('id');
    $this->db->where('id', $booking_id);
    $this->db->where('status', 'completed');
    $this->db->where('deleted_at', NULL);

    $query = $this->db->get($this->table);
    return $query->num_rows() > 0;
}

/**
 * Get booking statistics for provider
 *
 * @param int $provider_id - Provider profile ID
 * @return object Statistics object
 */
public function get_provider_stats($provider_id) {
    $query = "
        SELECT
            COUNT(*) as total_bookings,

```

```

        SUM(CASE WHEN status = 'completed' THEN 1 ELSE
0 END) as completed_bookings,
        SUM(CASE WHEN status = 'cancelled' THEN 1 ELSE
0 END) as cancelled_bookings,
        SUM(CASE WHEN status = 'pending' THEN 1 ELSE 0
END) as pending_bookings,
        AVG(CASE WHEN status = 'completed' THEN
(SELECT AVG(rating) FROM reviews WHERE provider_id =
{$provider_id}) ELSE NULL END) as average_rating
        FROM bookings
        WHERE provider_id = {$provider_id} AND deleted_at
IS NULL
";

```

```

        return $this->db->query($query)->row();
    }

```

```

/**

```

```

 * Count customer bookings by status
 *
 * @param int $customer_id - Customer user ID
 * @param string $status - Booking status
 * @return int Count of bookings
 */

```

```

public function count_by_customer_status($customer_id,
$status) {
    $this->db->where('customer_id', $customer_id);
    $this->db->where('status', $status);
    $this->db->where('deleted_at', NULL);

    return $this->db->count_all_results($this->table);
}

```

```

/**
 * Count provider bookings by status
 *
 * @param int $provider_id - Provider profile ID
 * @param string $status - Booking status
 * @return int Count of bookings
 */
public function count_by_provider_status($provider_id,
$status) {
    $this->db->where('provider_id', $provider_id);
    $this->db->where('status', $status);
    $this->db->where('deleted_at', NULL);

    return $this->db->count_all_results($this->table);
}

/**
 * Count provider bookings
 *
 * @param int $provider_id - Provider profile ID
 * @param string $status - Booking status (optional)
 * @return int Count of bookings
 */
public function count_provider_bookings($provider_id,
$status = NULL) {
    $this->db->where('provider_id', $provider_id);
    $this->db->where('deleted_at', NULL);

    if ($status) {

```

```

        $this->db->where('status', $status);
    }

    return $this->db->count_all_results($this->table);
}

/**
 * Get recent bookings ordered by creation date
 *
 * @param int $limit - Number of bookings to return
 * @return array Array of recent bookings
 */
public function get_recent($limit = 10) {
    $this->db->select('b.*, c.full_name as customer_name,
c.phone as customer_phone, sp.business_name, s.service_name');
    $this->db->from('bookings b');
    $this->db->join('users c', 'b.customer_id = c.id');
    $this->db->join('service_provider_profiles sp',
'b.provider_id = sp.id');
    $this->db->join('services s', 'b.service_id = s.id');
    $this->db->where('b.deleted_at', NULL);
    $this->db->order_by('b.created_at', 'DESC');
    $this->db->limit($limit);

    return $this->db->get()->result();
}

/**
 * Count all bookings by status (admin use)
 *

```

```

    * @param string $status - Booking status
    * @return int Count of bookings
    */

    public function count_by_status($status) {
        $this->db->where('status', $status);
        $this->db->where('deleted_at', NULL);

        return $this->db->count_all_results($this->table);
    }

    /**
     * Get bookings by status (admin use)
     *
     * @param string $status - Booking status
     * @param int $limit - Results limit
     * @param int $offset - Pagination offset
     * @return array Array of bookings
     */

    public function get_by_status($status, $limit = 20,
    $offset = 0) {
        $this->db->select('b.*, c.full_name as customer_name,
    c.phone as customer_phone, sp.business_name, s.service_name');

        $this->db->from('bookings b');

        $this->db->join('users c', 'b.customer_id = c.id');

        $this->db->join('service_provider_profiles sp',
    'b.provider_id = sp.id');

        $this->db->join('services s', 'b.service_id = s.id');

        $this->db->where('b.status', $status);

        $this->db->where('b.deleted_at', NULL);

        $this->db->order_by('b.booking_date DESC,
    b.booking_time DESC');

```

```

        $this->db->limit($limit, $offset);

        return $this->db->get()->result();
    }

    /**
     * Count customer bookings
     *
     * @param int $customer_id - Customer user ID
     * @param string $status - Booking status (optional)
     * @return int Count of bookings
     */
    public function count_customer_bookings($customer_id,
    $status = NULL) {

        $this->db->where('customer_id', $customer_id);
        $this->db->where('deleted_at', NULL);

        if ($status) {
            $this->db->where('status', $status);
        }

        return $this->db->count_all_results($this->table);
    }
}

```

?>

Tampilan

View/index.php

<!DOCTYPE html>

<html lang="en">

<head>

```
<meta charset="UTF-8">

<meta name="viewport" content="width=device-width,
initial-scale=1.0">

<title>Sekitarku.id - Find Verified Providers</title>

<link rel="stylesheet"
href="https://unpkg.com/leaflet@1.7.1/dist/leaflet.css" />

<style>
    * {
        margin: 0;
        padding: 0;
        box-sizing: border-box;
    }

    body {
        font-family: 'Segoe UI', Tahoma, Geneva, Verdana,
sans-serif;
        background-color: #f5f7fa;
    }

    header {
        background: linear-gradient(135deg, #667eea 0%,
#764ba2 100%);
        color: white;
        padding: 20px 0;
        box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);
    }

    .nav-container {
        max-width: 1200px;
        margin: 0 auto;
        display: flex;
```

```
        justify-content: space-between;
        align-items: center;
        padding: 0 20px;
    }

    .logo {
        font-size: 24px;
        font-weight: 700;
    }

    .nav-links {
        display: flex;
        gap: 20px;
        list-style: none;
    }

    .nav-links a {
        color: white;
        text-decoration: none;
        transition: opacity 0.3s;
    }

    .nav-links a:hover {
        opacity: 0.8;
    }

    .hero {
        background: linear-gradient(135deg, #667eea 0%,
#764ba2 100%);
        color: white;
```

```
        padding: 60px 20px;
        text-align: center;
    }

.hero h1 {
    font-size: 42px;
    margin-bottom: 15px;
}

.hero p {
    font-size: 18px;
    margin-bottom: 40px;
    opacity: 0.95;
}

.search-container {
    max-width: 600px;
    margin: 0 auto;
    background: white;
    padding: 20px;
    border-radius: 10px;
    box-shadow: 0 10px 30px rgba(0, 0, 0, 0.2);
}

.search-form {
    display: grid;
    grid-template-columns: 1fr 1fr;
    gap: 15px;
}
```

```
.form-group {  
    grid-column: 1 / -1;  
}
```

```
.form-group.half {  
    grid-column: span 1;  
}
```

```
input, select {  
    width: 100%;  
    padding: 12px;  
    border: 1px solid #ddd;  
    border-radius: 5px;  
    font-size: 14px;  
}
```

```
input:focus, select:focus {  
    outline: none;  
    border-color: #667eea;  
    box-shadow: 0 0 0 3px rgba(102, 126, 234, 0.1);  
}
```

```
.button-group {  
    grid-column: 1 / -1;  
    display: flex;  
    gap: 10px;  
}
```

```
button {  
    flex: 1;
```

```
padding: 12px;
border: none;
border-radius: 5px;
font-size: 16px;
font-weight: 600;
cursor: pointer;
transition: all 0.3s;
}

.btn-search {
    background: linear-gradient(135deg, #667eea 0%,
#764ba2 100%);
    color: white;
}

.btn-search:hover {
    transform: translateY(-2px);
    box-shadow: 0 5px 15px rgba(102, 126, 234, 0.4);
}

.btn-locate {
    background: white;
    color: #667eea;
    border: 2px solid #667eea;
}

.btn-locate:hover {
    background: #f5f7fa;
}
```

```
.categories-section {
    max-width: 1200px;
    margin: 60px auto 0;
    padding: 0 20px;
}

.section-title {
    font-size: 28px;
    color: #333;
    margin-bottom: 30px;
}

.categories-grid {
    display: grid;
    grid-template-columns: repeat(auto-fit,
minmax(150px, 1fr));
    gap: 15px;
}

.category-card {
    background: white;
    padding: 20px;
    border-radius: 10px;
    text-align: center;
    cursor: pointer;
    transition: all 0.3s;
    box-shadow: 0 2px 10px rgba(0, 0, 0, 0.05);
    text-decoration: none;
    color: inherit;
}
```

```
.category-card:hover {
    transform: translateY(-5px);
    box-shadow: 0 10px 25px rgba(102, 126, 234, 0.15);
}

.category-icon {
    font-size: 40px;
    margin-bottom: 10px;
}

.category-name {
    font-weight: 600;
    color: #333;
    font-size: 14px;
}

.features {
    background: white;
    margin-top: 60px;
    padding: 60px 20px;
}

.features-grid {
    max-width: 1200px;
    margin: 0 auto;
    display: grid;
    grid-template-columns: repeat(auto-fit,
minmax(250px, 1fr));
    gap: 30px;
```

```
}

.feature {
    text-align: center;
}

.feature-icon {
    font-size: 48px;
    margin-bottom: 15px;
}

.feature h3 {
    color: #333;
    margin-bottom: 10px;
}

.feature p {
    color: #666;
    font-size: 14px;
    line-height: 1.6;
}

footer {
    background: #333;
    color: white;
    text-align: center;
    padding: 30px 20px;
    margin-top: 60px;
}
```

```

    @media (max-width: 768px) {
        .hero h1 {
            font-size: 28px;
        }

        .search-form {
            grid-template-columns: 1fr;
        }

        .form-group.half {
            grid-column: 1 / -1;
        }

        .button-group {
            flex-direction: column;
        }
    }
</style>
</head>
<body>
    <header>
        <div class="nav-container">
            <div class="logo">🏠 Healthcare Pro</div>
            <ul class="nav-links">
                <?php if (isset($is_logged_in) &&
$is_logged_in): ?>
                    <li><span style="color: #fff;">Welcome,
<?php echo htmlspecialchars($user_name ?? 'User');
?></span></li>

```

```
        <li><span style="color: #ccc;">(<?php echo
htmlspecialchars(ucfirst(str_replace('_', ' ', $user_role ??
'customer'))); ?>)</span></li>
```

```
        <?php if ($user_role === 'customer'): ?>
```

```
            <li><a href="<?php echo
site_url('customer/dashboard'); ?>">My Dashboard</a></li>
```

```
            <li><a href="<?php echo
site_url('customer/bookings'); ?>">My Bookings</a></li>
```

```
        <?php elseif ($user_role ===
'service_provider'): ?>
```

```
            <li><a href="<?php echo
site_url('profesi/dashboard'); ?>">Provider Dashboard</a></li>
```

```
            <li><a href="<?php echo
site_url('profesi/bookings'); ?>">My Bookings</a></li>
```

```
        <?php elseif ($user_role === 'admin'): ?>
```

```
            <li><a href="<?php echo
site_url('admin/dashboard'); ?>">Admin Dashboard</a></li>
```

```
            <li><a href="<?php echo
site_url('admin/providers'); ?>">Manage Providers</a></li>
```

```
        <?php endif; ?>
```

```
        <li><a href="<?php echo
site_url('auth/logout'); ?>">Logout</a></li>
```

```
        <?php else: ?>
```

```
            <li><a href="<?php echo
site_url('auth/login'); ?>">Login</a></li>
```

```
            <li><a href="<?php echo
site_url('auth/register'); ?>">Register</a></li>
```

```
        <?php endif; ?>
```

```
    </ul>
```

```
</div>
```

```
</header>
```

```
<section class="hero">
```

```
    <h1>Find Trusted Healthcare Professionals</h1>
```

<p>Search, book, and connect with verified healthcare providers near you</p>

```
<div class="search-container">

    <form class="search-form" id="searchForm"
action="<?php echo site_url('guest/search'); ?>" method="get">

        <div class="form-group">

            <input type="text" name="q"
placeholder="Search by name or specialty" id="searchKeyword">

        </div>

        <div class="form-group half">

            <select name="category"
id="categorySelect">

                <option value="">All
Categories</option>

                <?php if (isset($categories)): ?>
                    <?php foreach ($categories as
$cat): ?>
                        <option value="<?php echo
$cat->id; ?>"><?php echo $cat->name; ?></option>
                    <?php endforeach; ?>
                <?php endif; ?>
            </select>
        </div>

        <div class="form-group half">

            <input type="text" name="city"
placeholder="City" id="cityInput">

        </div>

        <div class="form-group half">
```

```
        <input type="text" name="radius"
placeholder="Radius (km)" value="10" id="radiusInput">
    </div>
```

```
    <div class="form-group half">
        <input type="hidden" name="lat"
id="latInput">
        <input type="hidden" name="lon"
id="lonInput">
    </div>
```

```
    <div class="button-group">
        <button type="submit" class="btn-
search">Search</button>
        <button type="button" class="btn-locate"
id="locateBtn"> 📍 Use My Location</button>
    </div>
</form>
</div>
</section>
```

```
<section id="map-section" class="categories-section">
    <h2 class="section-title">Find Providers Near You</h2>
    <div id="map" style="height: 400px; width: 100%;
border-radius: 8px;"></div>
</section>
```

```
<?php if (isset($categories) && count($categories) > 0):
?>
<section class="categories-section">
    <h2 class="section-title">Browse by Category</h2>
    <div class="categories-grid">
```

```

        <?php foreach ($categories as $cat): ?>

            <a href="<?php echo
site_url('guest/search?category=' . $cat->id); ?>"
class="category-card">

                <div class="category-icon"> 📁 </div>

                <div class="category-name"><?php echo
$cat->name; ?></div>

            </a>

        <?php endforeach; ?>

    </div>

</section>

<?php endif; ?>

<section class="features">
    <div class="features-grid">
        <div class="feature">

            <div class="feature-icon"> 📍 </div>

            <h3>Location-Based Search</h3>

            <p>Find providers near you using GPS or search
by city</p>

        </div>

        <div class="feature">

            <div class="feature-icon"> ✅ </div>

            <h3>Verified Professionals</h3>

            <p>All providers are verified and approved by
admins</p>

        </div>

        <div class="feature">

            <div class="feature-icon"> ⭐ </div>

            <h3>Ratings & Reviews</h3>

            <p>See real reviews from verified
customers</p>

```

```

    </div>

    <div class="feature">

        <div class="feature-icon">🗓️</div>

        <h3>Easy Booking</h3>

        <p>Book appointments online instantly</p>
    </div>
</div>
</section>

<footer>

    <p>© 2026 sekitarku.id. All rights reserved.</p>
</footer>

<script
src="https://unpkg.com/leaflet@1.7.1/dist/leaflet.js"></script
>

<script>

    // Default map center (e.g., Jakarta, Indonesia)
    var defaultLat = -6.2088;
    var defaultLon = 106.8456;
    var defaultZoom = 13;

    var map = L.map('map').setView([defaultLat,
defaultLon], defaultZoom);

    L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}
/{y}.png', {
        attribution: '&copy; <a
href="https://www.openstreetmap.org/copyright">OpenStreetMap</
a> contributors'
    }).addTo(map);

```

```

        <?php if (!empty($providers)): ?>
            <?php foreach ($providers as $provider): ?>
                L.marker([<?php echo $provider->latitude; ?>,
<?php echo $provider->longitude; ?>])
                    .addTo(map)
                    .bindPopup("<b><?php echo
html_escape($provider->business_name); ?></b><br><?php echo
html_escape($provider->address); ?>");
            <?php endforeach; ?>
        <?php else: ?>
            // If no providers are loaded, maybe show a
            general message or just the default view
            console.log("No providers to display on map.");
        <?php endif; ?>

        document.getElementById('locateBtn').addEventListener(
'click', function() {
            if (navigator.geolocation) {
                navigator.geolocation.getCurrentPosition(function(position) {
                    document.getElementById('latInput').value
= position.coords.latitude;
                    document.getElementById('lonInput').value
= position.coords.longitude;
                    alert('Location detected: ' +
position.coords.latitude.toFixed(4) + ', ' +
position.coords.longitude.toFixed(4));
                }, function(error) {
                    alert('Unable to get your location. Please
enter manually.');
```

```
});

// Auto-complete for search
document.getElementById('searchKeyword').addEventListener('input', function(e) {
    if (e.target.value.length < 2) return;

    fetch('<?php echo site_url("guest/autocomplete");
    ?>?q=' + encodeURIComponent(e.target.value))
        .then(r => r.json())
        .then(data => console.log('Suggestions:',
data))
        .catch(err => console.error('Error:', err));
});
</script>
</body>
</html>
```