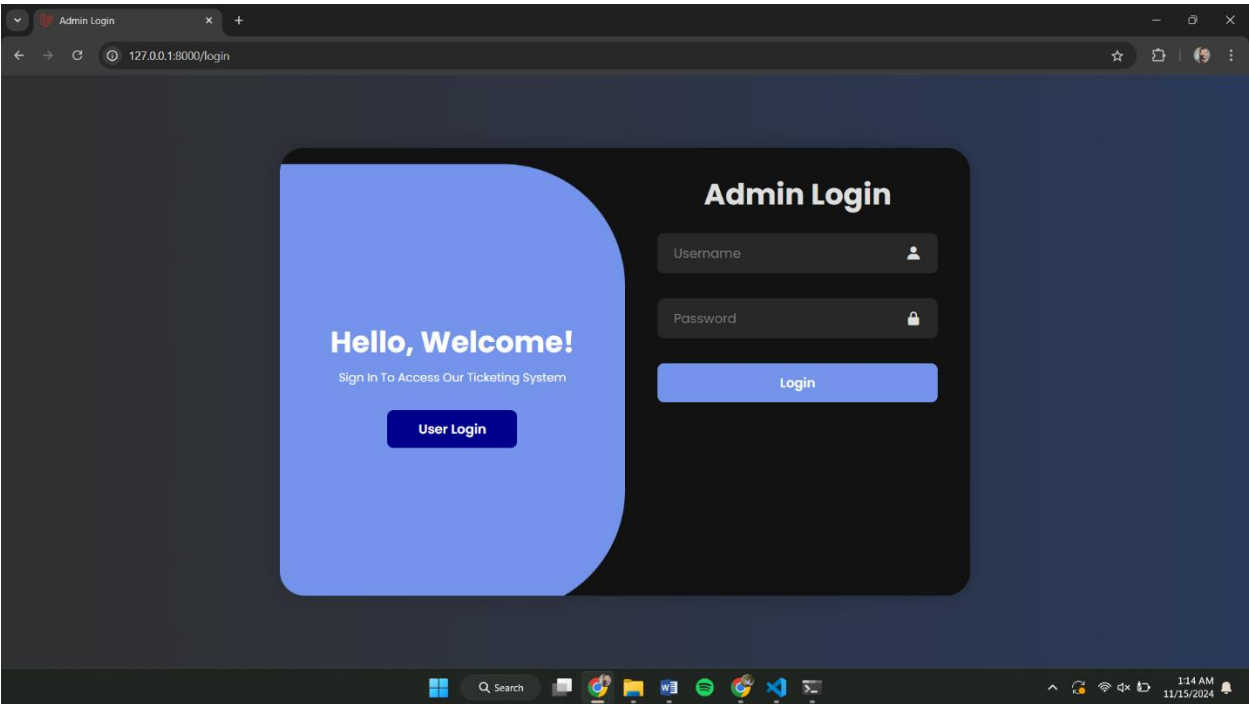
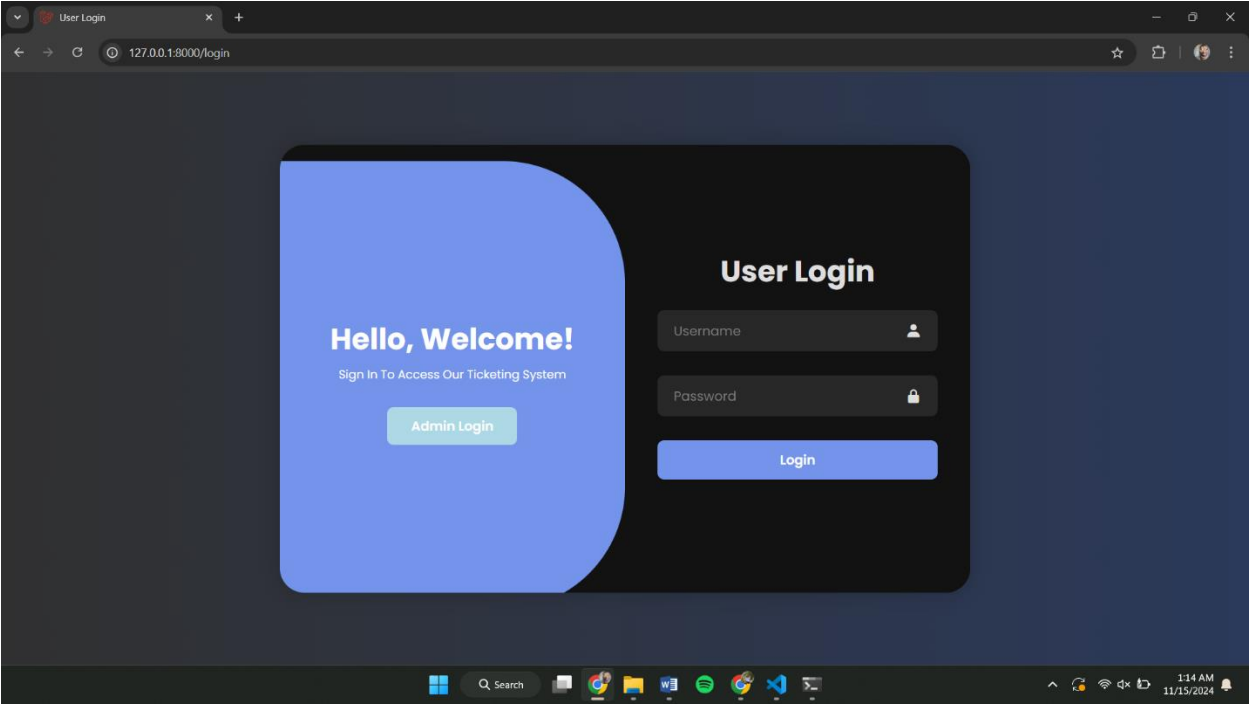


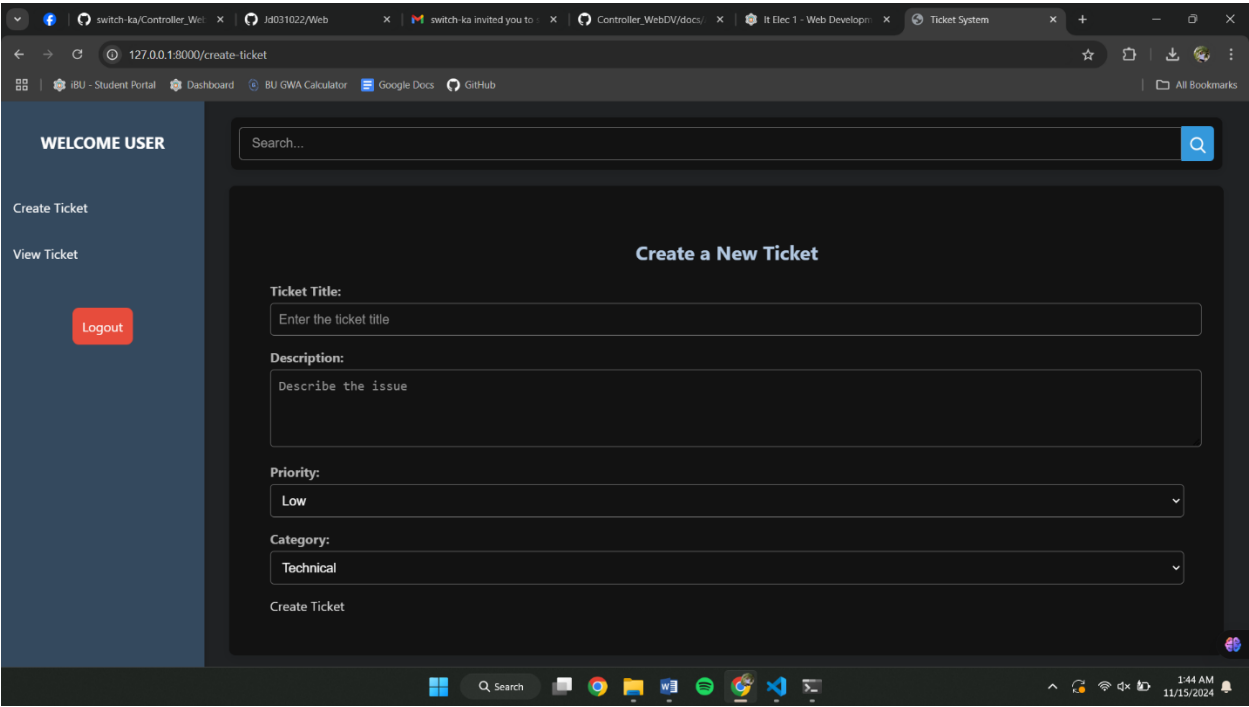
The pages

This is the use interface for the admin and login page where you can input username and password

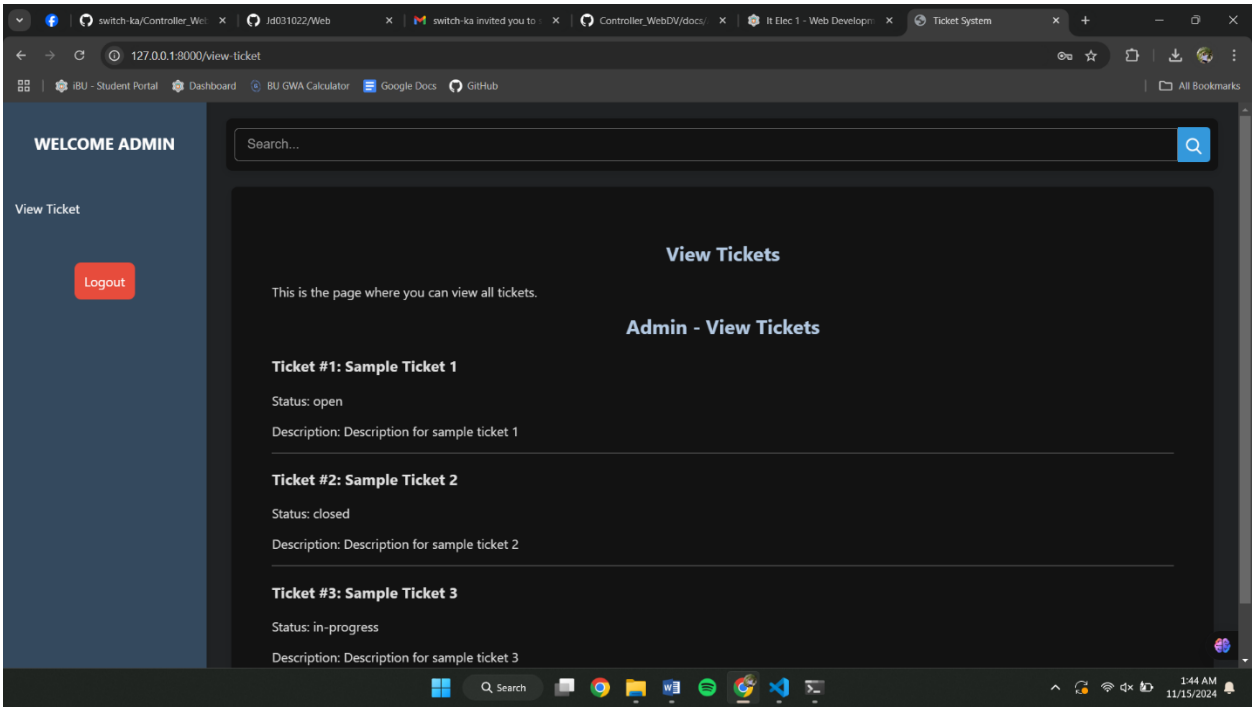


Create Ticket and View Ticket User Interface

This is where the users can create or submit their ticket. (For Users)



This is where the admin can see the ticket that the user created (For Admin)



Routes(web.php)

This Laravel routing setup directs users based on their roles in a ticketing system. The main page (/) redirects to the login page. "User" role users can access create-ticket to create tickets and store-ticket to submit them. The login page has separate routes for regular users (/login) and admins (/admin-login).

```
terminal  Help  Controller_WebDV-main

web.php x
routes > web.php > Closure
1  <?php
2
3  use App\Http\Controllers\AuthController;
4  use App\Http\Controllers\TicketController;
5  use Illuminate\Support\Facades\Route;
6
7  // Make the login page the landing page
8  Route::get(uri: '/', action: function () { return redirect()->route(route: 'login');
9  });
10
11
12 // Route to show the create ticket form - restricted to users with 'user' role
13 Route::get(uri: '/create-ticket', action: [TicketController::class, 'create'])
14     ->name(name: 'create-ticket')
15     ->middleware(middleware: 'role:user');
16
17 // Route to handle form submission (store the ticket) - restricted to users with 'user' role
18 Route::post(uri: '/store-ticket', action: [TicketController::class, 'store'])
19     ->name(name: 'store-ticket')
20     ->middleware(middleware: 'role:user');
21
22 // Login routes
23 Route::get(uri: '/login', action: [AuthController::class, 'showLoginForm'])->name(name: 'login');
24
25 // Separate POST routes for user and admin logins
26 Route::post(uri: '/login', action: [AuthController::class, 'loginUser'])->name(name: 'login.submit');
27 Route::post(uri: '/admin-login', action: [AuthController::class, 'loginAdmin'])->name(name: 'admin.login');
28
29 // Logout route
30 Route::post(uri: '/logout', action: [AuthController::class, 'logout'])->name(name: 'logout');
31
32 // Admin-specific route to view tickets - restricted to users with 'admin' role
33 Route::get(uri: '/view-ticket', action: [TicketController::class, 'view'])
34     ->name(name: 'view-ticket')
35     ->middleware(middleware: 'role:admin');
36
```

Controllers

AuthController.php

It manages login and logout for users and admins. showLoginForm displays the login page. loginUser and loginAdmin validate user and admin credentials, setting the session and redirecting to create-ticket or view-ticket based on role. If credentials are incorrect, they return an error. logout clears the session and redirects to login. This setup enables role-based access and redirection.

```
terminal  Help  Controller_WebDV-main

AuthController.php x
app > Http > Controllers > AuthController.php > PHP Intelephense > AuthController > loginUser
5 references | 0 implementations
class AuthController extends Controller
{
9  // Show the login form
10 1 reference | 0 overrides
11  public function showLoginForm(): Factory|View
12  {
13      return view(view: 'login');
14  }
15
16 // Handle User login
17 1 reference | 0 overrides
18  public function loginUser(Request $request): mixed|RedirectResponse
19  {
20      $credentials = $request->only(keys: 'username', 'password');
21
22      if ($credentials['username'] === 'user' && $credentials['password'] === 'userpassword') {
23          session(key: ['user_type' => 'user']);
24          return redirect()->route(route: 'create-ticket');
25      }
26
27      return redirect()->back()->with(key: 'error', value: 'Invalid username or password');
28  }
29
30 // Handle Admin login
31 1 reference | 0 overrides
32  public function loginAdmin(Request $request): mixed|RedirectResponse
33  {
34      $credentials = $request->only(keys: 'username', 'password');
35
36      if ($credentials['username'] === 'admin' && $credentials['password'] === 'adminpassword') {
37          session(key: ['user_type' => 'admin']);
38          return redirect()->route(route: 'view-ticket');
39      }
40
41      return redirect()->back()->with(key: 'error', value: 'Invalid username or password');
42  }
43
44 }
```

TicketController.php

It handles ticket creation, storage, and viewing. The create method shows the ticket form for users, while store saves the ticket and redirects with a success message. The view method allows admins to see all tickets.

ninal
Help
Controller_WebDV-main
TicketController.php
app > Http > Controllers > TicketController.php > ...
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
Ln 5, Col 29
Spaces: 4
UTF-8

```

1  <?php
2
3  namespace App\Http\Controllers;
4
5  use Illuminate\Http\Request;
6
7  class TicketController extends Controller
8  {
9      // Show the ticket creation form
10     public function create(): Factory|View
11     {
12         return view(view: 'create-ticket'); // Make sure this view exists
13     }
14
15     // Store the ticket - restricted to users
16     public function store(Request $request): RedirectResponse
17     {
18         // Logic for storing the ticket goes here
19
20         return redirect()->route(route: 'view-ticket')->with(key: 'success', value: 'Ticket created successfully!');
21     }
22
23     // View tickets - restricted to admins
24     public function view(): Factory|View
25     {
26         // Dummy data to simulate tickets
27         $tickets = [
28             [
29                 'id' => 1,
30                 'title' => 'Sample Ticket 1',
31                 'status' => 'open',
32                 'description' => 'Description for sample ticket 1',
33             ],
34         ]

```

ninal
Help
Controller_WebDV-main
TicketController.php
p > Http > Controllers > TicketController.php > ...
7
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
Ln 5, Col 29

```

7  class TicketController extends Controller
21  {
22
23     // View tickets - restricted to admins
24     public function view(): Factory|View
25     {
26         // Dummy data to simulate tickets
27         $tickets = [
28             [
29                 'id' => 1,
30                 'title' => 'Sample Ticket 1',
31                 'status' => 'open',
32                 'description' => 'Description for sample ticket 1',
33             ],
34             [
35                 'id' => 2,
36                 'title' => 'Sample Ticket 2',
37                 'status' => 'closed',
38                 'description' => 'Description for sample ticket 2',
39             ],
40             [
41                 'id' => 3,
42                 'title' => 'Sample Ticket 3',
43                 'status' => 'in-progress',
44                 'description' => 'Description for sample ticket 3',
45             ],
46         ];
47
48         return view(view: 'view-ticket', data: compact(var_name: 'tickets'));
49     }
50 }
51

```