|  |
| --- |
| using System; |
|  | using System.Collections.Generic; |
|  | using System.Threading.Tasks; |
|  | using ChatApplication.DataService; |
|  | using ChatApplication.Models; |
|  | using Microsoft.AspNetCore.Mvc; |
|  | using Microsoft.Extensions.Logging; |
|  |  |
|  | namespace ChatApplication.Apis |
|  | { |
|  | [Produces("application/json")] |
|  | [Route("api/login")] |
|  | public class LoginController : Controller |
|  | { |
|  | private LoginService \_logindata; |
|  | private ILogger \_Logger; |
|  |  |
|  | public LoginController(LoginService logindata, ILoggerFactory loggerFactory) |
|  | { |
|  | \_logindata = logindata; |
|  | \_Logger = loggerFactory.CreateLogger(nameof(LoginController)); |
|  | } |
|  |  |
|  | [HttpGet] |
|  | [ProducesResponseType(typeof(List<UserLogin>), 200)] |
|  | [ProducesResponseType(typeof(ApiResponse), 400)] |
|  | public async Task<ActionResult> Employees() |
|  | { |
|  | try |
|  | { |
|  | var users = await \_logindata.GetUsersAsync(); |
|  | return Ok(users); |
|  | } |
|  | catch (Exception exp) |
|  | { |
|  | \_Logger.LogError(exp.Message); |
|  | return BadRequest(new ApiResponse { Status = false }); |
|  | } |
|  | } |
|  |  |
|  | [HttpGet("{name}")] |
|  | [ProducesResponseType(typeof(UserLogin), 200)] |
|  | public async Task<ActionResult> GetUser(string name) |
|  | { |
|  | try |
|  | { |
|  | var user = await \_logindata.GetUserAsync(name); |
|  | return Ok(user); |
|  | } |
|  | catch (Exception exp) |
|  | { |
|  | \_Logger.LogError(exp.Message); |
|  | return BadRequest(new ApiResponse { Status = false }); |
|  | } |
|  | } |
|  |  |
|  | [HttpPost] |
|  | [ProducesResponseType(typeof(ApiResponse), 201)] |
|  | [ProducesResponseType(typeof(ApiResponse), 400)] |
|  | public async Task<ActionResult> AddUser([FromBody]UserLogin info) |
|  | { |
|  | if (!ModelState.IsValid) |
|  | { |
|  | return BadRequest(new ApiResponse { Status = false, ModelState = ModelState }); |
|  | } |
|  |  |
|  | try |
|  | { |
|  | var newuser = await \_logindata.AddUserAsync(info); |
|  | if (newuser == null) |
|  | { |
|  | return BadRequest(new ApiResponse { Status = false }); |
|  | } |
|  | return CreatedAtRoute("GetUserRoute", new { id = newuser.id }, newuser); |
|  | } |
|  | catch (Exception exp) |
|  | { |
|  | \_Logger.LogError(exp.Message); |
|  | return BadRequest(new ApiResponse { Status = false }); |
|  | } |
|  | } |
|  | [HttpPut("{id}")] |
|  | [ProducesResponseType(typeof(ApiResponse), 200)] |
|  | [ProducesResponseType(typeof(ApiResponse), 400)] |
|  | public async Task<ActionResult> UpdateUser(int id, [FromBody]UserLogin info) |
|  | { |
|  | if (!ModelState.IsValid) |
|  | { |
|  | return BadRequest(new ApiResponse { Status = false, ModelState = ModelState }); |
|  | } |
|  |  |
|  | try |
|  | { |
|  | var status = await \_logindata.UpdateUserStatusAsync(info); |
|  | if (!status) |
|  | { |
|  | return BadRequest(new ApiResponse { Status = false }); |
|  | } |
|  | return Ok(new ApiResponse { Status = true, User = info }); |
|  | } |
|  | catch (Exception exp) |
|  | { |
|  | \_Logger.LogError(exp.Message); |
|  | return BadRequest(new ApiResponse { Status = false }); |
|  | } |
|  | } |
|  | } |
|  | } |