GLFW

http://www.glfw.org/

GLFW is an Open Source, multi-platform library for OpenGL, OpenCL, ES and Vulkan development. It provides a simple API for creating window contexts and waitlists, reopening window events.

GLFW is written in C and has native support for Windows, macOS and many Unix-like systems via the X Window System, such as Linux and FreeBSD.

GLFW is licensed under the dual licensing model.

Gives you a window and OpenGL context with just two function calls.

Support for OpenGL, OpenCL, ES, Vulkan and related options, flags and extensions.

Support for multiple windows, multiple monitors, high DPI and gamma ramps.

Support for keyboard, mouse, gamepad, touch and window event input, via polling or callbacks.

Comes with guides, tutorials, reference documentation, examples and build programs.

Open Source with an OSI certified license allowing commercial use.

Access to native objects and complete APIs for platforms specific features.

Community-maintained bindings for many different languages.

No library can be perfect for everyone. If GLFW isn't what you're looking for, there are alternatives.
void InitGLFW()
{
    glfwInit();
    glfwWindowHint(GLFW_CLIENT_API, GLFW_NO_API);
    glfwWindowHint(GLFW_RESIZABLE, GLFW_FALSE);
    MainWindow = glfwCreateWindow(Width, Height, "Vulkan Sample", NULL, NULL);
    VkResult result = glfwCreateWindowSurface(Instance, MainWindow, NULL, &Surface);
    glfwSetErrorCallback(GLFWErrorCallback);
    glfwSetKeyCallback(MainWindow, GLFWKeyboard);
    glfwSetCursorPosCallback(MainWindow, GLFWMouseMotion);
    glfwSetMouseButtonCallback(MainWindow, GLFWMouseButton);
}

GLFW Keyboard Callback

void GLFWKeyboard( GLFWwindow * window, int key, int scancode, int action, int mods )
{
    if( action == GLFW_PRESS )
    {
        switch( key )
        {
            //case GLFW_KEY_M:
            case 'm':
            case 'M':
            Mode++;
            if( Mode >= 2 )
                Mode = 0;
            break;
            default:
            fprintf(FpDebug, "Unknow key hit: 0x%04x = "v"%c\n", key, key);
            fflush(FpDebug);
        }
    }
}
void GLFWMouseButton ( GLFWwindow *window, int button, int action, int mods )
{
    int b = 0;  // LEFT, MIDDLE, or RIGHT
    // get the proper button bit mask:
    switch( button )
    {
        case GLFW_MOUSE_BUTTON_LEFT:
            b = LEFT;               break;
        case GLFW_MOUSE_BUTTON_MIDDLE:
            b = MIDDLE;             break;
        case GLFW_MOUSE_BUTTON_RIGHT:
            b = RIGHT;              break;
        default:
            b = 0;
            fprintf( FpDebug, "Unknown mouse button: %d\n", button );
            break;
    }
    // button down sets the bit, up clears the bit:
    if( action == GLFW_PRESS )
        {
            double xpos, ypos;
            glfwGetCursorPos( window, &xpos, &ypos);
            Xmouse = (int)xpos;
            Ymouse = (int)ypos;
            ActiveButton |= b;              // set the proper bit
        }
    else
        {
            ActiveButton &= ~b;             // clear the proper bit
        }
}

void GLFWMouseMotion ( GLFWwindow *window, double xpos, double ypos )
{
    int dx = (int)xpos - Xmouse;            // change in mouse coords
    int dy = (int)ypos - Ymouse;
    if( ( ActiveButton & LEFT ) != 0 )
        {
            Xrot += ( ANGFACT*dy );
            Yrot += ( ANGFACT*dx );
        }
    if( ( ActiveButton & MIDDLE ) != 0 )
        {
            Scale += SCLFACT * (float) ( dx - dy );
            // keep object from turning inside-out or disappearing:
            if( Scale < MINSIZE )
                Scale = MINSIZE;
        }
    Xmouse = (int)xpos;                     // new current position
    Ymouse = (int)ypos;
}
Looping and Closing GLFW

```c
while( glfwWindowShouldClose( MainWindow ) == 0 )
{
    glfwPollEvents( );
    Time = glfwGetTime( ); // elapsed time, in double-precision seconds
    UpdateScene( );
    RenderScene( );
}

vkQueueWaitIdle( Queue );
vkDeviceWaitIdle( LogicalDevice );
DestroyAllVulkan( );
glfwDestroyWindow( MainWindow );
glfwTerminate( );
```