Annya OS And Lousine Kernel Documentation
Main Document version 1.01s
Written By Tyler Grenier on 11/30/23 At 9:50 PM

Hello World
Welcome To AnnyaOS™
Based On The Lousine Kernel™

Hello all, Welcome to Annya OS™, built on the Lousine kernel™. The Lousine Kernel is a brand new kernel written from scratch with compatibility for Microsoft Corp Windows™ Software And Drivers while creating a friendly User Interface And A User Focused Designed While being Completely Open Source And Free From Any Cooperate Control or Microsoft Violations Similar To React OS But This One Is Entirely From Scratch and is not Using Any WINE Or translation software.

Introduction From The Creator:

Hello World my name is Tyler Grenier I am the writer of this Kernel/OS, I would like to start by telling Microsoft I am not a threat, I am not trying to steal clients from them what so ever, I would like to bring this up because Microsoft is full with a bunch of egotistic maniacs who this that because one thing that happens to be something they do for profit get an open source or any type of competition then they assume and jump the gun on sueing that person for everything they have even though Microsoft owns ninety percent of all computer businesses in the world and they get more money than anyone else per day. So if your reading this and you are from Microsoft and your not happy because I am doing this just remember I am doing this without profit and I am doing this without knowledge on company secrets so GET OFF YOUR FUCKING HIGH HORSE…

As you can tell from the first part of my introductions I have not time for being politically correct. In fact I have absolutly no time for politics at all so if your here and you do anything politically you will be banished from the earth never to participate in any projects I own… P.S. I have a lot of projects not just this one although this is so far the biggest one and the one I am most passionate for. I started my programming carrier making operating systems one of the reasons is it teaches you proper coding right off the bat because if you don't do proper coding the system will surely not work… The first operating system I made myself was a 32 bit kernel written in c plus plus and it used several of the Lousine kernel systems however the kernel was a prototype that never came to life because due to the fact that everything was written in c it was very non-versatile linking was nearly impossible dynamically because my exports would be in c plus plus format that was garbled by the compiler and due to this there was never a shell that I could integrate simply so the project was scrapped and abandoned and there are now no remnants of it anywhere due to my computer breaking last year. The second operating system that I made was a simple dos kernel in 32 bit mode it was never named do to the fact that it was never actually targeted for anything other than something I made because I was bored and it ran a shell called TGDOS which is my original shell for running old 16 bit programs on a 32 bit machine without having to switch back to 16 bit mode in order to handle system calls or interrupts. That kernel was also fairly small and has no remnants of it today. Finally today we have the Lousine Kernel that is mainly based off of C for its core components such as IO through Ports, System and memory management, Dynamic Linking, and last but not least security… Though I will hopefully expand to other languages for security but thats what I have for now. The second main language I use for the kernel Is c plus plus however the c plus plus is made for two things and both are higher level kernel systems that use the kernels C functions to do various tasks. These Two Main Functions Are to Create Internal Kernel Drivers/Modules that need to be handled internal to the kernel such as storage drivers to load kernel components to the system as well as files system drivers to keep the systems integrity safe so drivers cannot access certain files even in kernel mode which has the highest privilege in all software. The C Plus Plus Also does Driver Management/IO Management and is important to run windows Drivers For most Windows Drivers are significantly Hard To Work With in just C. The second use for C Plus Plus In the kernel is to create the same thing as the IO Manager/Driver Manager Principle but with User Mode System calls and UserMode handling Which is done by either dynamically linked Dll Therefore linked by the kernels c code and then jumped into by the C Plus Plus UserMode Abstraction layer To make sure that all dependencies for the User Mode Software is Attended to for most User mode Programs Are Written in C Plus Plus.

Now that we have basic introductions out of the way lets talk about the significants and the importance of the project for me and everyone using this operating system. Now you may ask what is the point of this, the point

of this is???

1. To give regular people and computer enthusiasts the ability to run a fair amount of windows programs and maybe one day all windows programs.
2. To teach people about computers and how they work and give them the opportunity to work on them themselves.
3. Give People with very low incomes an opportunity to have their own windows experience without having to pay hundreds of dollars of dollars for windows.

Now You may be asking why is it named Annya OS, and Why is my kernel named the “Lousine Kernel”? The name of the OS and kernel is based off of my dog Annya Lousine, she is a black lab beagle mix. And not only is the name of the project after her she is also my projects mascot.

Some func facts about the Lousine kernel are, that the kernel is actually not a COFF EXE like the executables built at Microsoft like the NT Kernel NTOSKRNL.EXE but the kernel is actually a multi boot header ELF Format binary file louoskrnl.bin but when it is transferred to the project it is copied byte for byte then the extension is changed to an executable file extension LOUOSKRNL.EXE, and because of the fact that when it is in the project it is an ELF multi boot executable with an executable file extension GRUB (GRand Unified Boot loader), is able to detect load and execute the kernel as any other multi boot kernel. The dynamically loadable libraries (.dll), System Files/Driver Files (.sys), and executables (.exe) other than the kernel itself is made by a Microsoft Software Development Kits. The kits are for drivers and system files I use the Windows WDK Kit, and for windows executables and dynamic loadable libraries I use the standard Windows SDK for windows with standard System Calls. Along Side This the System also Is Going to be able to run its own sdk and driver kit when I decide to make them

Figure 1: Annya Lousine on her first day with us