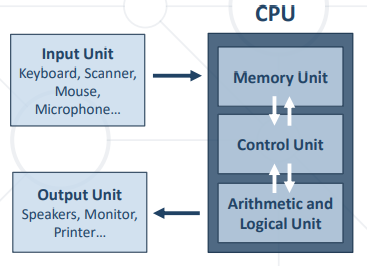
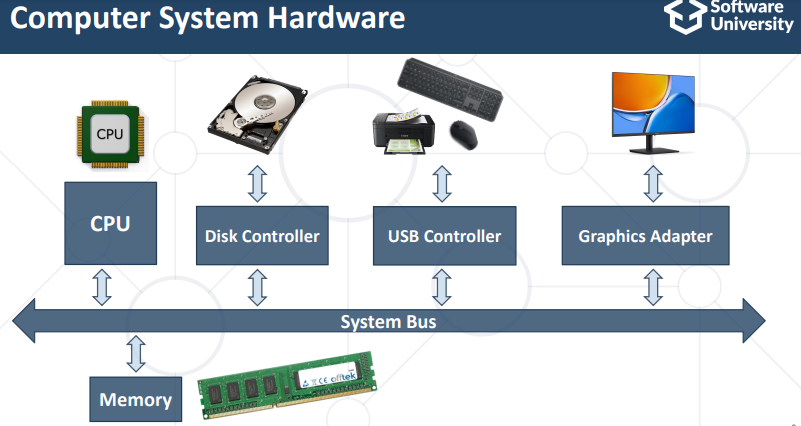
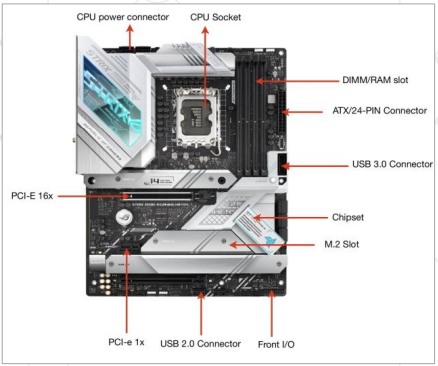
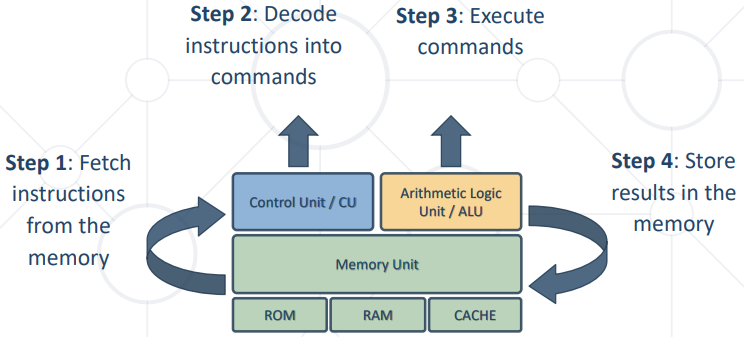
Software Technologies Fundamentals  
  
Computer Systems and Software

Hardware: Motherboard, CPU, RAM, Storage, Peripherals   
Software: Firmware, System, Server-Side, Applications  
  
▪ **Компютърна система**: състав от хардуер и софтуер. Система която съчетава хардуерни и софтуерни компоненти (лаптоп, компютър,смартфон и тнт)  
▪ Позволява ефикасно въвеждане на данни, процесване и външна производителност (звук, картина)   
▪ Включва взаимосвързани софтуерни и хардуерни компоненти   
▪ извършва компютърно-човешки интеракци/действия за потребителя

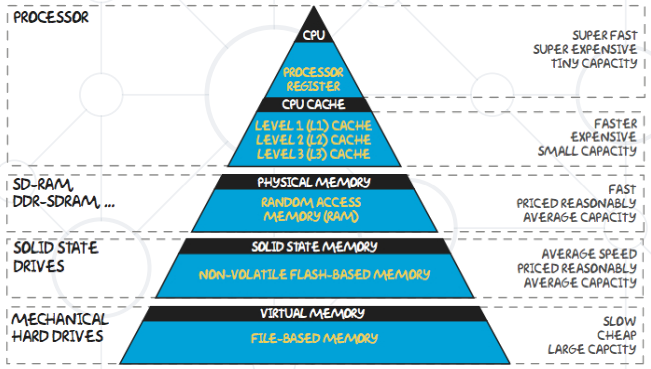
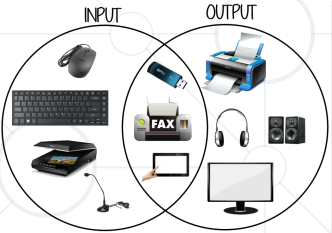
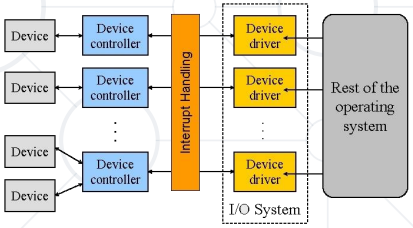
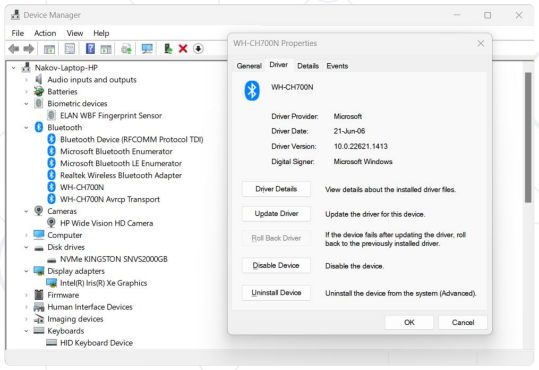
▪ **Ключови елементи:**   
 ▪ Hardware: RAM(памет), входни/изходни устройства,дънна платка(motherboard),CPU (процесор)  
 ▪ Software: операционни системи, и, апликаций,игри

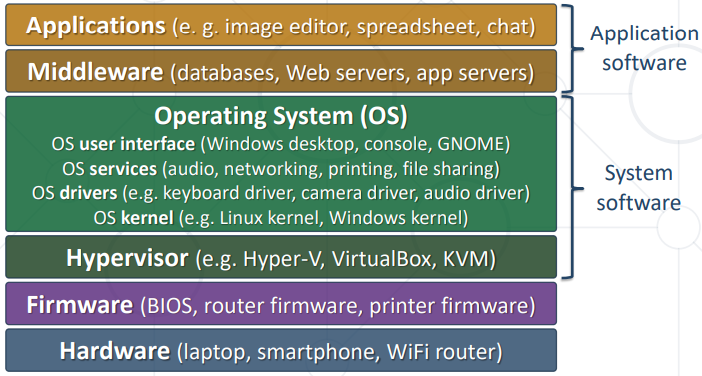
**Компютърен Hardware** – дънна платка, процесор, памет , съхранение, периферни устройства  
▪ Хардуерът са физическите компоненти на компютъра   
1. Central Processing Unit (CPU) – микропроцесор   
▪ изпълнява кодове (програми)   
▪ извършва действия по обработка на данни  
2. Входни устройства ▪ за въвеждане на данни (мишка, клавиатура)  
3. Изходни устройства ▪ за възпроизвеждане на данни/информация – картина,звук  
 

  
  
4 **Дънна платка** == централен хъб където се свързват всички хардуерни компоненти   
 ▪ извършва комуникация между всички хардуерни компоненти   
▪ Различна съвместимост   
 ▪ всяко дъно е разработено да функционира с различен тип процесори и памет   
▪ Разширени слотове за по-добра функционалност   
 ▪ Видео карти (GPU) за по-добра картина   
 ▪ Звукови карти за подобрени звукови възможности   
 ▪ Мрежови карти за по-добра интернет връзка  
  
**Компоненти на дънна платка** ▪ гнездо за процесор ▪ слотове за РАМ памет ▪ Конектори за захранване ▪ Чипсет ▪ разширителни слотове ▪ САТА конектори ▪ USB конектори ▪ Bluetooth модул  


▪ **CPU (процесор)– мозъкът на компютъра**   
 ▪ извършва изчисления, действия и изпълнява програми   
 ▪ осигурява мощност и контрол на инструкциите(кодове)   
▪ 3 основни компонента   
 ▪ Контролен елемент **(CU)** ▪ управлява потока от инструкций и координира хардуерните функций   
 ▪ **Аритметичен и логически елемент (ALU)** ▪ извършва аритметични и логически операций   
 ▪ **Памет елемент (MU)** ▪ съхранява данни, програми и информация  
  
  
  
**Памет и съхранение – съхранява данни в компютъра**

**▪ Главна памет**   
 ▪ RAM – четене и запис: съхранява данни необходими на процесора по време на изпълнение на програма   
 ▪ ROM – само за четене: съхранява важни данни за работата на системата като (booting) зареждането на компютъра при включване

**▪ Вторична памет**   
 ▪ няма директен достъп от процесора   
 ▪ примери: хард диск, SSD, USB drive   
**▪ Кеширана (временна) памет**   
▪ паст от процесора: временно съхранява често използвани данни и инструкции за ускоряване на извършването на действия  
  
  
**Периферни устройства** – разширяват функционалността на компютъра  
▪ 3 основни категории:   
 ▪ **входни устройства** → клавиатура, мишка, микрофон   
 ▪ **изходни устройства** → принтер, монитор   
 ▪ **Смесени** → хард диск, мрежова карта, тъчскрийн монитор  
  
  
▪ **Контролер на устройство**  
 ▪ физическо устройство за връзка между периферно устройство и компютъра   
 ▪ например USB контролер   
**▪ Драйвер на устройство**   
 ▪ системен софтуер, който позволява комуникация и трансфер на данни между устройствата и ситемата  
  
  
**Компютърен софтуер** - Фърмуер, ситемен софтуер, апликации  
▪ Компютърен софтуер – програми, инструкции и данни който позволяват на компютъра да извършва специфични действия   
▪ **Видове софтуер**:   
 ▪ Приложен софтуер: програми който извършват специфична лична, бизнес или образователна функция

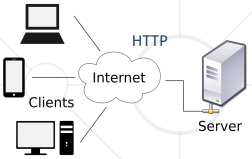
▪ Системен софтуер: взаимодейства и управлява хардуерните компоненти на компютъра   
  
  
  
**Слоеве на софтуера:**  
▪ **Firmware и вграден софтуер**   
 ▪ ниско ниво софтуер което позволява хардуерно устройство да функционира   
▪ **Системен софтуер**   
 ▪ управлява и контролира хардуер, платформа за приложения   
 ▪ Операционна система (OS) – Windows, Linux, macOS, Android   
 ▪ Хипервайзор – управлява виртуални машини   
▪ **приложен софтуер**   
 ▪ бизнес апликации, офис апликаци, мултимедия, комуникация   
 ▪ няколко вида: уеб аппликаци, десктоп апликации, мобилни апликации  
  
**Софтуерни системи**

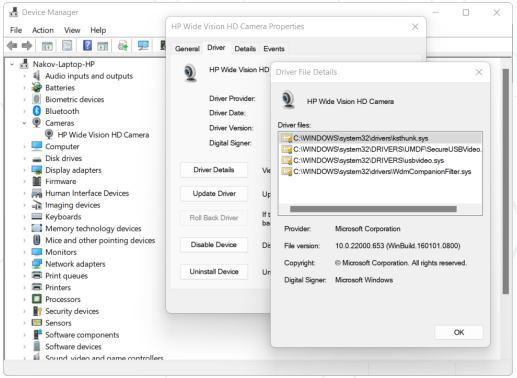
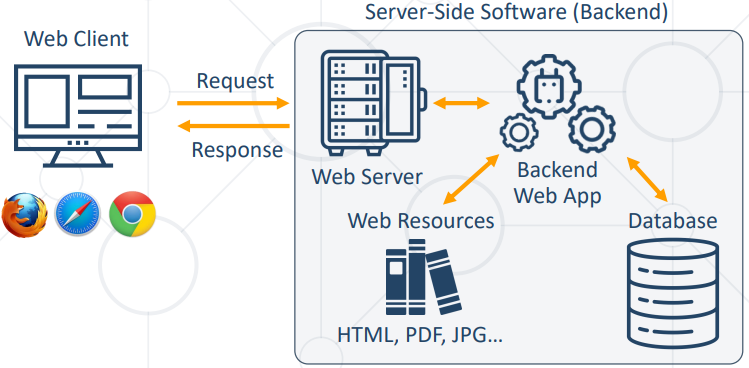
▪ **Самостоятелни проложения**   
 ▪ изпълняват се локално, съхраняват данните си локално, не се нуждаят от интернет   
 ▪ примери: Windows калкулатор, Windows Explorer, Minesweeper   
▪ **Software systems**   
 ▪ състои се от няколко компонента (напр. клиент + сървър)   
 ▪ пример: mail сървър (дистанционно) + mail клиентски апп (локално)   
▪ **Cloud apps:** съхранява потребителски данни в облак + локален клиент

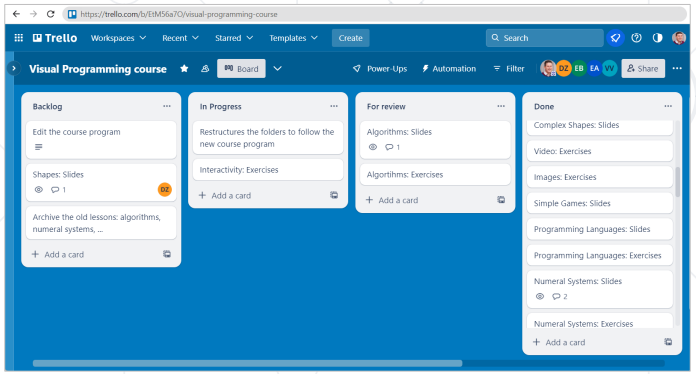
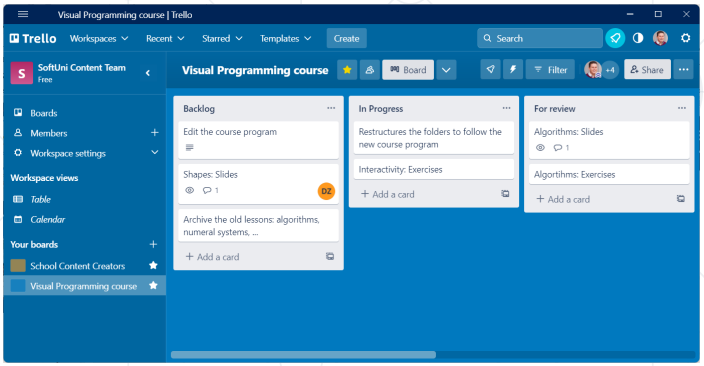
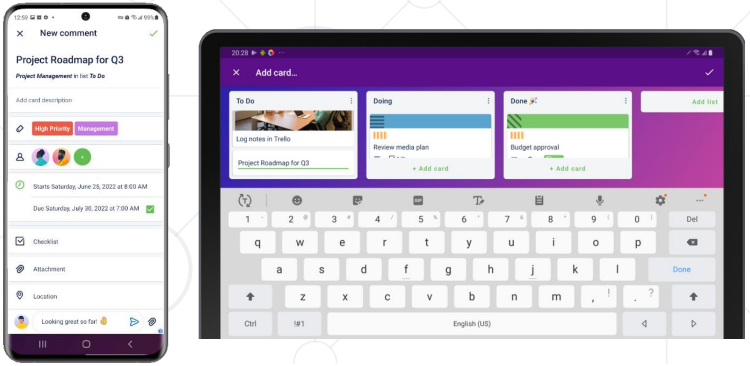
▪ пример: Google Docs, Discord, Trello, Canva  
  
▪ **Front-end and back-end разделят модерните приложения на клиентски и сървърни компоненти**

▪ **Front-end** == компоненти от страна на клиента (Desktop / mobile app / Web browser)   
 ▪ внедрява юзърския интерфейс   
▪ **Back-end** == компоненти от страна на сървър

▪ внедрява съхранение и обработка на данни Front-End/Back-end

Пример : **HTTP свързва frontend (потребителя) със back-end (сървъра)**  
  
**Firmware** – „мостът“ между хардуер и софтуер  
▪ **Firmware** == перманентен (вграден) софтуер от ниско ниво, вграден в паметта само за четене (ROM) на устройството   
 ▪ контролира основните функций на устройството и осигурява стабилна основа за софтуери от по-горно ниво  
 ▪ например: WiFi router's firmware, coffee machine firmware   
▪ **Функции на фирмуера:**   
 ▪ инициализация на хардуера по време на стартирането на компютъра   
 ▪ управлява ниско ниво хардуерни действия (напр.. стартирането на устройство, хардуерна диагностика и системния старт на операционната система)

**Operating Systems (Операционни системи)**  
▪ Windows, macOS, Linux, Android, iOS   
▪ управлява ресурсите на хардуерa и софтуерa  
▪ управлява процеси при едновременно работещи приложения   
▪ разпределя системни ресурси между всички процеси  
▪ управлява файловата система и паметта (RAM)   
▪ управлява юзъри, защита и контрол над достъпът   
▪ системни ъпдейти и поддръжка  
  
**Device Drivers**   
▪ In Windows, the "Device Manager" lists all devices, drivers, etc.  
  
  
**System Utilities**  
▪ Tools that help maintain and optimize a computer system   
 ▪ Antivirus and malware protection (e.g. Winows Defender)   
 ▪ System backup and recovery (e. g. Macrium Reflect)   
 ▪ Disk cleanup and defragmentation (e. g. CCleaner)   
 ▪ Performance monitoring and diagnostics (Task Manager)   
 ▪ Software updates and patches (e. g. Windows Update)   
 ▪ System hardware information (e. g. CPU-Z)   
 ▪ System logs viewer (e. g. Windows Events Viewer)  
  
**Server-Side Software (Backend) -** Facilitating Backend Operations and Web Services  
▪ Server-side software (backend software) runs on a remote server, processes requests and delivers data to client devices   
  
▪ Common types of **server*-*side software**   
 ▪ Web servers (e. g. Apache, Nginx, IIS)   
 ▪ Database servers (e. g. MySQL, PostgreSQL, MongoDB)   
 ▪ Application servers / runtimes (e. g. Tomcat, Node.js, .NET Core)   
 ▪ Mail servers (e. g. Microsoft Exchange Server, Postfix)   
 ▪ File servers (e. g. Windows File Server, Samba)   
 ▪ Authentication servers (e. g. FreeIPA, Active Directory)  
  
**▪ Server-side software (backend software):**  
 ▪ Executes on a remote server, rather than on the user's device   
 ▪ Handles data processing, storage, and retrieval   
 ▪ Powers Web applications, backend APIs, cloud services, etc.   
 ▪ Requires efficient resource management for optimal performance   
**▪ Graphical User Interface (GUI) / front-end apps:**   
 ▪ Executes on the user's device (desktop, mobile, or Web)   
 ▪ Providing seamless and visually appealing user experience   
 ▪ Can be Web apps, desktop apps, or mobile apps  
  
**Application Software** - Apps for the End Users  
▪ Application software is designed for users to perform specific business tasks, catered to their individual needs   
**▪ Examples of application software**   
 ▪ Productivity tools (Microsoft Office, Google Workspace)   
 ▪ Multimedia software (Adobe Photoshop, VLC Media Player)   
 ▪ Communication apps (Zoom, WhatsApp, MS Teams)   
 ▪ Web browsers (Google Chrome, Mozilla Firefox, Safari)   
 ▪ Games (Fortnite, League of Legends)  
  
**Web Apps** - Applications, Accessed from the Web Browser  
▪ What are Web apps?   
 ▪ Accessed through a Web browser with an active Internet connection   
 ▪ **Platform-independent**   
 ▪ Accessible on any device with a Web browser   
 ▪ Desktop/mobile Web browsers   
 ▪ **Automatic updates (always up-to-date)**   
 ▪ No need for manual installation or updating

▪ **Benefits of Web apps**   
 ▪ Scalability: easily accommodate a growing user base   
 ▪ Centralized data storage: simplifies data management and backup   
 ▪ Lower device requirements: minimal hardware needed (processing is done on the server-side)   
 ▪ Easier collaboration: real-time collaboration   
 ▪ Cross-platform compatibility: works across various operating systems and devices  
  
▪ **Compatibility:** if the app works consistently across different Web browsers and different screen sizes (responsive design)   
▪ **Usability:** testing for accessibility, intuitive use on different devices, and ease of navigation   
▪ **Network conditions:** Web apps rely on an active internet connection→ testing under different network conditions   
▪ **Security:** Web apps deal with sensitive data → testing for vulnerabilities such as XSS attacks and SQL injection   
▪ **Performance:** performance can be affected by network speed / server load / browser capabilities → testing for scalability / load capacity  
  
  
  
**Desktop Apps** - Applications Running Locally on Your Laptop  
▪ What are desktop apps?   
 ▪ **Installed and run locally on a user's computer**   
 ▪ Store their data locally or remotely (depends)   
 ▪ **Offline access**   
 ▪ Can be used without an Internet connection   
 ▪ **More features**   
 ▪ Often more feature-rich than Web apps   
 ▪ Better integrated with the host OS  
  
  
▪ Benefits of desktop apps   
 ▪ **Performance:** faster processing and response time, as tasks are executed locally   
 ▪ **Customization:** easily tailored to individual user preferences and needs   
 ▪ **Integration:** compatible with other locally installed software and hardware   
 ▪ **Cost-effective:** one-time purchase or licensing fees, instead of recurring subscription costs (depends)  
  
▪ Installation / uninstallation including any dependencies or prerequisites   
▪ Performance testing on different hardware configurations – processors, memory, and graphic cards ▪ Compatibility testing for different operating systems and their different versions   
▪ User interface testing: desktop apps often have complex UI that need to be thoroughly tested   
▪ Integration testing with other desktop applications  
  
  
  
**Mobile Apps** - Applications Running Locally on Mobile Device  
▪ What are mobile apps?   
 ▪ Designed specifically for smartphones and tablets   
 ▪ Accessible through dedicated app stores (e.g., Google Play, Apple App Store)   
 ▪ Optimized for touchscreen interfaces and mobile device features (adaptable UI design for different screen sizes)   
 ▪ Can work offline, online or mixed  
**▪ Benefits of mobile apps**   
 ▪ **Portability:** access apps and data on-the-go, anytime, anywhere   
 ▪ **Push notifications:** real-time updates and alerts for improved user engagement   
 ▪ **Device-specific features:** leverage device capabilities like GPS, camera, and sensors   
 ▪ **Offline functionality**: some apps can operate without an Internet connection   
 ▪ **Streamlined user experience:** tailored for smaller screens and touch-based interactions  
  
  
  
**Testing Challenges for Mobile Apps**  
▪ **Compatibility** across different devices and OS versions is crucial for mobile apps (many different devices and versions in use)   
▪ **User interface testing** – design and layout has significant impact on the user's experience on a smaller screen   
▪ **Performance testing** – performance may be affected by limited processing power and memory on the user's device   
▪ **Battery life testing** – to ensure that the app does not significantly drain the user's device battery  
  
  
  
**Summary**  
▪ **Hardware** is the physical part, whereas software is a set of instructions for the computer   
 ▪ Main computer parts are the motherboard (ties together all components), CPU (code execution), input / output devices   
▪ **Software** – programs, running in the computer   
 ▪ Firmware and system software (OS, hypervisors)   
 ▪ Server-side software (back-end) vs. GUI / front-end apps   
 ▪ Application software (end-user apps): Web apps, desktop apps, mobile apps   
 ▪ Software systems (client + server) and cloud apps  
  
  
  
  
  
  
  
  
  
  
  
Operating Systems  
OS Overview, Linux Shell, VM and Containers  
  
**Операционни системи (OS) – преглед** – цел и структура  
▪ операционната система управлява апликации(процеси), юзъри, файлова система и ресурси в устройството

▪ Операционната система се зарежда в устройството чрез процес наречен booting (стартиране/зареждане)   
▪ ОС позволява на апликациите да взаимодействат със ресурсите от хардуерните и софтуерните компоненти   
▪ апликациите извършват заявки за услуги чрез дефиниран интефейс наречен иинтерфейс на приложна програма( API)   
▪ поне една ОС трябва да бъде инсталирана на устройство за да може да се ползват основни програми (уеб браузъри, видео плеиър, достъп до папки и файлове и тнт).  
**Основна фунция на ОС**

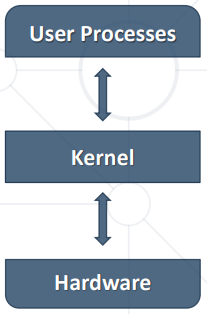
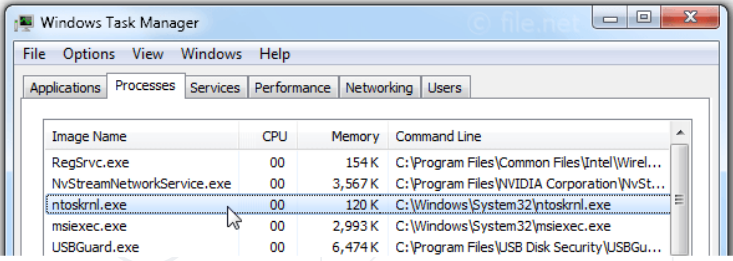
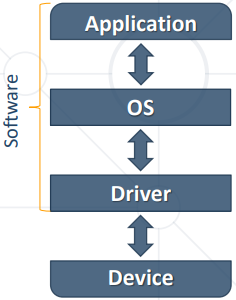
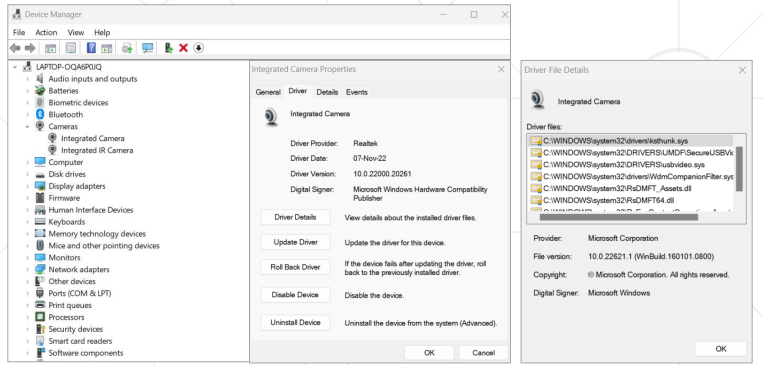
▪ **Booting (стартиране)** – включване на устройството и зареждане на ОС   
▪ **зареждане на апликации и извършване на фунцкиите им**

▪ **управление на процеси** – разпределя ресурси за процеси на ОС, споделянето на данни между процеси, защита и синхронизация

▪ **управление на памет** – контролира и координира разпределянето на памет за аппликациите който работят в ОС

▪ **управление на съхраняването** – manages storage (hard drives, SSD disks, optical disk drives, flash drives) и файловата система  
▪ **контрол на устройствата** – контролира достъпът до физическите устройства (СД/ДВД/USB) и виртуални устройства

▪ **Работа в мрежа** – комуникация между мрежи и интернет   
▪ **Контролиране на принтери** – takes control of printers connected and manages the printing process   
▪ Юзър интефейс **(UI)** – предоставя визуални елементи (интефейс) за потребителя за работа и извършване на действия

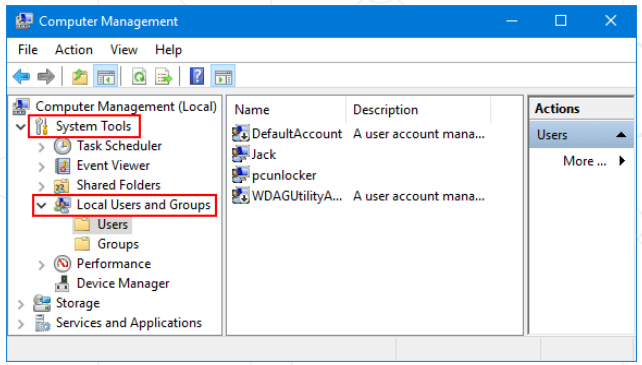
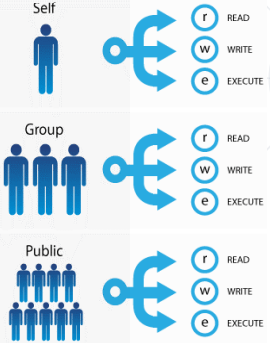
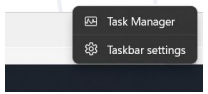
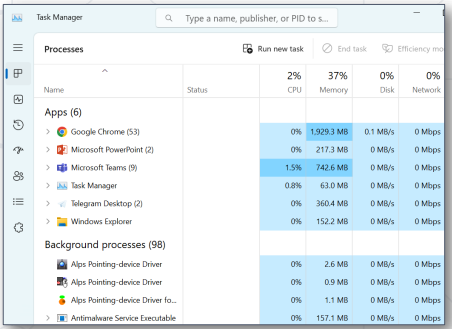
▪ **защита на данни** – изолиращи програми, юзъри и файлове който поддържат данните защитени  
▪ **Kernel** == основния компонент в ОС (сърцето)   
▪ мост (свързване) между хардуерните и софтуерните компоненти   
▪ улеснява комуникацията между различните системни компоненти  
▪ предоставя пълен контрол над системата   
▪ винаги бди над нас   
▪ незаменимо/есенциално за функциониране на операционна система  
  
  
  
▪ **Drivers/драйвъри (Drive – под** == група от системни програми който позволяват на хардуерните компоненти да функционират   
▪ ниско ниво софтуери без интерфейс (UI)   
▪ всички хардуерни устройства изискват драйвъри (e. g. disk drives, printers, keyboards)  
  
  
  
▪ **Shell** == user interface (UI) to the OS   
 ▪ Outermost layer of the operating system, located between the kernel and the apps   
 ▪ Provides a UI and tools to control processes, files, installed software, users, etc.   
▪ Two types of shells:   
 ▪ **Command-line (CLI) shells** – require knowledge of commands, syntax, and concepts about the shell-specific scripting language (e. g. bash)   
 ▪ **Graphical (GUI) shells** – intuitive, easy to use (e. g. Windows Desktop)   
▪ Most GUI-enabled OS provide also CLI shells for advanced users

**Юзъри в операционните системи**  
▪ == лица който взаимодействат със системата като влизат (логват се) и изпълняват задачи. Юзерът има акаунт и се идентифицира в системата със своя юзърнейм

▪ Юзърите могат да имат привилегии върху определени процеси, папки,файлове, устройства, мрежа и други ресурси

▪ Юзърите обикновено са изолирани един от друг   
  
▪ **Акаунтите имат достъп до системните ресурси**   
▪ Удостоверяването е процес на проверка на самоличността на потребителя   
 ▪ чрез креденшъли (удостоверителни данни като пароли/ключове)

▪ Упълномощяването опредлея до какви ресурси има достъп даден потребител

▪ Акаунтите в ОС са важни за счетоводството, сигурността, регистрирането и управлението на данни  
  
  
**Потребителски права**  
▪ ОС контролира използването на системните и мрежовите настроики   
 ▪ Чрез удостоверяване и оторизиране   
 ▪ Въз основа на потребителските разрешения над ресурси (напр. разрешения за ползване на специфични папки, файлове и тнт)   
▪ ОС OS определя дали if an удостоверения юзър има правилните разрешения да използва даден ресурс в системата  
 ▪ Използва вградени технологии за авторизация и контрол на достъпа  
**Юзърски роли (групи)**  
▪ Набори от разрешения които контролират достъпа на ресурси (файлове, папки, процеси, услуги"   
▪ Всеки юзър може да има няколко роли   
▪ Примери за юзър роли - **MS Windows**: Администратор, гост   
▪ Примери за юзър роли **в** **Linux**: root (основен), юзър, никой (гост)  
  
**Access Permissions in OS**  
▪ **Access permissions** determine a user's ability to perform a specific action, or access a feature or object   
▪ Set access permissions to specify which users, groups, or roles can access your content   
▪ The most common permissions are read, write and execute  
  
  
**Processes in OS**  
▪ A process is a program in action (a running app)   
 ▪ Consume CPU time, RAM memory, file handles and other OS resources   
▪ It's the basic unit of work in the operating system   
▪ Unlike files, which are passive, processes are an active entity   
▪ For example, when you open a browser to search the web, that's a process  
  
**Таск мениджър (Диспечер на задачи)**  
▪ В ОС – система която мониторира процеси и задачи   
 ▪ Показва процеси, юзъри, използвани ресурси от системата и тнт   
 ▪ Показва натоварването на RAM, процесора, видеокарта, съхранение, мрежово/интернет натоварване  
 ▪ Може да стартира или терминира активни процеси   
  
  
**Windows Таск мениджър (Диспечер на задачи)**▪ Как се отваря:   
 ▪ [Ctrl + Alt + Delete] → избирате диспечер на задачи (Task manager)  
 ▪ Десен бутон на старт лентата → [Task Manager]  
  
  
  
**Popular Operating Systems -** Windows, Linux, macOS, Android, iOS  
  
**Най-основни Operating Systems**  
▪ 5 основни операционни системи:  
▪ Microsoft Windows ▪ Apple macOS ▪ Google's Android OS ▪ Apple iOS ▪ Linux (open source)  
  
**Microsoft Windows**▪ Proprietary OS, developed by Microsoft   
▪ One of the most popular OS   
 ▪ Typically preinstalled on new PC   
▪ Several versions: Windows 95 / 98 / Vista, Windows 7 / 8 / 10 / 11   
 ▪ Has been around since the 1980s   
▪ Easy-to-use, intuitive GUI shell   
 ▪ Many apps and games  
  
**Apple macOS**  
▪ Apple and Macintosh computers run on macOS and OS X   
 ▪ Proprietary OS developed by Apple   
▪ macOS is a Unix-based OS   
 ▪ Released over 20 years ago   
▪ In 2020, Apple began transitioning to its own 64-bit ARM-based Apple M CPU   
 ▪ Apple M1 / M2 CPU: powerful and silent  
  
  
  
**Android OS**  
▪ Mobile OS, designed for touchscreen mobile devices   
▪ Based on a modified version of the Linux kernel and other open-source software   
▪ Core OS is called Android Open-Source Project (AOSP)   
 ▪ Free and open-source software   
 ▪ Developed and maintained by Google   
▪ Many distributions (by Samsung, Xiaomi)  
  
**Apple iOS**  
▪ Mobile OS, developed by Apple   
 ▪ Exclusively for its hardware devices: iPhone, iPad and iPod Touch   
▪ Closed ecosystem, dominated by Apple   
▪ iOS UI uses multi-touch gestures: swipe, tap, pinch, and reverse pinch   
▪ iOS runs on Apple hardware only   
 ▪ Might run on PC emulators, but is illegal  
  
**Linux**  
▪ Linux is Free and open-source family of operating systems   
▪ Linux's popularity comes from its ease of customization and open license   
▪ Offers CLI shell and many GUI desktops   
▪ Many distributions: Ubuntu, CentOS, Debian, Mint, openSUSE, Alpine, …   
▪ It offers a variety of options for those who understand how to use it  
  
**Virtual Machines & Containers –** дистанционни инстанции и емулатори  
**Virtual Machines (VM) – виртуални машини**  
▪ Виртуалната машина е софтуерно базиран компютърен ресурс който може да стартира операционна система в друга операционна система

▪ Представлява дигитална версия на физически компютър която може да стартира програми и ОС, запаметява данни, свързване към мрежи и други компютърни функции   
▪ **Виртуализация** == стартиране на виртуална машина/среда във физическа хардуерна система ▪ напр. стартиране на Андроид ОС или Линукс във Window-ски хост

▪ **Контейнери и докер**  
▪ **Контейнер (образ)** == софтуер пакетиран със зависимости, разработен да функционира във виртуална среда

**Summary**  
▪ Operating Systems (OS) manage processes, users, files and other resources   
▪ OS Examples: Windows, macOS, Linux, Android, iOS   
▪ Virtual machine (VM) == OS inside another OS   
▪ Container == app image, running in Docker   
▪ Shell commands == execute commands from the console (Linux / Windows shell)

**Network, Internet and Protocols**

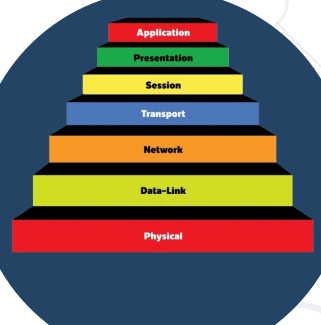
**Network протокол – набор от правила които определят как данните са предавани между различни устройства в една мрежа**  
▪ Позволяват стандартизирана комуникация между устройства и програми   
▪ Обикновено една страна изпраща заявка и получава отговор от другата страна   
▪ Мрежовите протоколи управляват аспектите на предаването на данни, маршрута, контрол на потока и обработка на грешки   
  
**---- Слоеве на мрежата (модели) ----**

▪ Слоевете организират мрежите в структурирана рамка  
 ▪ Улесняват мрежовата комуникация и разрешаване на проблеми   
▪ Насърчават оперативна съвместимост и модулност на протоколите

**Два основни модела :**  
▪ OSI (7 слоя)

▪ TCP (4 слоя)

**OSI модел**



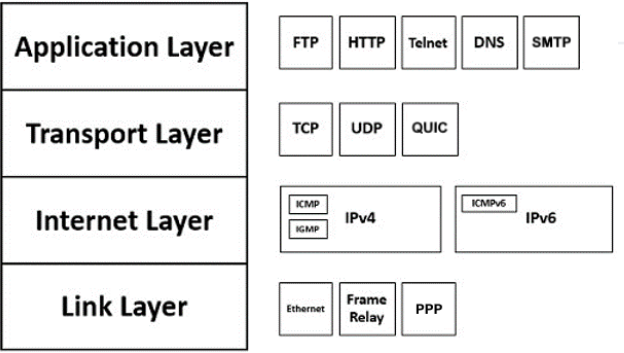
**OSI Модел – съдържа 7 слоя, като всеки слой надгражда предишния и придава нови функционалности**  
▪ (1) **Physical Layer/Физически слой** – преобразува дигитални данни във електроимпулси, радио сигнали или оптични сигнали за трансмисия (антени, хъбове, кабели, радио и тнт)   
▪ (2) **Data Link Layer / Слой за връзка с данни** (МАК адреси) – управлява предаването на данни, засича грешки и коригиране (превключватели, мостове, мрежови интерфейс карти)  
**▪ (3) Network layer / Мрежов слой** (хост и IP адреси) – пренасочване на връзка от Хост > рутер> рутер….>краен хост  
▪ (**4) Transport layer / Транспортиращ слой** – проверява за грешки, контролира потока

▪ (3**) Network layer / Мрежов слой** (хост и IP адреси) – пренасочване на връзка от Хост > рутер> рутер….>краен хост  
▪ **(4) Transport layer / Транспортиращ слой** – проверява за грешки, контролира потока  
▪ **(5) Session Layer / Слой на сесиите** – диалогов контрол, управление на токени, синхронизация

▪ (**6) Presentation Layer / Презентационен слой** – представяне на данни, криптиране, декриптиране, компресия и декомпресия

▪ **(7) Application layer / Приложен слой** – работа в мрежа за приложения напр. уеб браузъри, използване на DNS, HTTP, HTTPS за отварянето на уеб сайт

**TCP модел**



**TCP/IP Protocol -** Transmission Control Protocol / Internet Protocol  
Опростена версия на OSI но с 4 слоя

**▪ (1) Link layer / Свързващ слой** - комбинира функциите на физическия и data-link слоевете на OSI модела

**▪ (2)** **Internet слой -**  отговаря на Network слоя от OSI модела

▪ **(3)** **Transporting/Транспортиращ слой -**  наподобява транспортиращия слой от OSI   
▪ **(4)** **Application/Приложен слой** **–** съчетание наSession, Presentation и Application слоевете от OSI модела

**Media Access Control Address (MAC)** – Хардуер/физически адрес който идентифицира индивидуалните електронни устройства в една мрежа (48 битов формат) – напр. 9c-93-4e-3f-14-f7

**Internet Protocol (IP) address –** 32 битов идентификатор назначен на устройства с цел адресиране и пренасочване (напр. 192.168.0.550)

**Netmask -** 32 битов номер който се използва за прикриване на мрежовата част от IP адрес

**Network address** – идентифицира мрежа

**Gateway (портал) –** IP адрес на рутера който се използва за достъп до Интернет

**Ports/Портове – цифрени идентификатори които се използват за разграничаване на специфични процеси или услуги които работят на устройство в мрежа**

* Улесняват от край до край (end-to-end) комуникацията между апликациите и различните устройства

**▪ Видове портове:**

▪ **TCP ports** – използват се за комуникативно-ориентирана връзка за осигуряване на надеждност и цялост на данните

**▪ UDP ports** – използват се за комуникация без директна връзка които осигуряват по-бързо предаване на данни за минимален разход

**▪ Номера на портове (Port numbers) -**  служат за идентификация на мрежови услуги :

22 – SSH, 53 – DNS, 80 – HTTP, 110 – POP3, 123 – NTP , 143 – IMAP

**WEB Fundamentals**

▪ Domain Name System (DNS) – йерархична разпределена система която преобразува имената на domain в IP адрес

▪ **Domain name** – уникално име за интернет хост, машина, уеб сайт – напр. www.softuni.bg

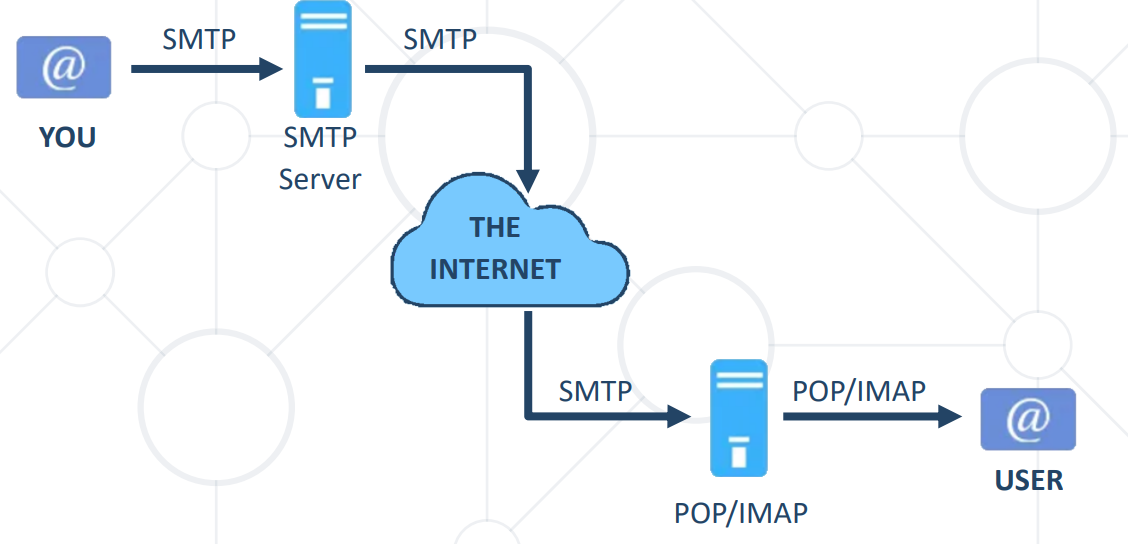
▪ **Uniform Resource Locator (URL)** – уникален адрес/линк който насочва към уебсайт, уебстраница (част от уебсайта) или документ в интернет

▪ **WWW (world wide web**) – глобална мрежа/система от документи, изображения и други ресурси, които се достъпват през Интернет с използването на уеб браузъри.

▪ **Web Service (мрежова услуга)** – осъществяват комуникацията между софтуерни системи или компоненти в мрежата

▪ **HTTP (Hyper Text Transfer Protocol)** – приложен слой предназначен за трансфериране на информация между мрежово свързани устройства. Основа на WWW, използва се за зарежда не на уеб страници като се използва хиперлинк (текст).

**E-mail Protocols**



▪ **SMTP (Simple Mail Transfer Protocol)** – протокол за приемане и изпращане на е-мейли между мейл сървъри

▪ **IMAP (Internet Message Access Protocol)** – извлича е-мейл съобщения от пощенската кутия на сървър. Позволява управлението на е-мейли на сървъра от различни устройства (синхронизация и изтриване)