**Тема: “Знайомство з інтерфейсом та можливостями ОС Linux”**

**Мета роботи:**

1. Знайомство з інтерфейсами ОС Linux.
2. Отримання практичних навиків роботи в середовищах ОС Linux та мобільної ОС – їх графічною оболонкою, входом і виходом з системи, ознайомлення зі структурою робочого столу, вивчення основних дій та налаштувань при роботі в системі

**Матеріальне забезпечення занять:**

1. ЕОМ типу IBM PC.

2. ОС сімейства Windows та віртуальна машина Virtual Box (Oracle).

3. ОС GNU/Linux (будь-який дистрибутив).

4. Сайт мережевої академії Cisco netacad.com та його онлайн курси по Linux

**Короткі теоретичні відомості:**

**Getting to the Command Line**

The command line interface (CLI) is a simple text input system for entering anything from single-word commands to complicated scripts. Most operating systems have a CLI that provides a direct way of accessing and controlling the computer.

On systems that boot to a GUI, there are two common ways of accessing the command line—a GUI-based terminal, and a virtual terminal:

* A GUI terminal is a program within the GUI environment that emulates a terminal window. GUI terminals can be accessed through the menu system. For example, on a CentOS machine, you could click on Applications on the menu bar, then System Tools > and, finally, Terminal. If you have a search tools, you can search for terminal, as shown here.
* A virtual terminal can be run at the same time as a GUI but requires the user to log in via the virtual terminal before they can execute commands (as they would before accessing the GUI interface).

Each Linux desktop distribution is slightly different, but the application terminal or x-term will open a terminal window from the GUI. While there are subtle differences between the terms console and terminal window sessions, they are all the same from an administrators standpoint and require the same knowledge of commands to use.

Ordinary command line tasks are starting programs, parsing scripts, and editing text files used for system or application configuration. Most servers boot directly to a terminal, as a GUI can be resource intensive and is generally not needed to perform server-based operations.‌⁠​​

**Applications**

The kernel of the operating system is like an air traffic controller at an airport, and the applications are the airplanes under its control. The kernel decides which program gets which blocks of memory, it starts and kills applications, and it handles displaying text or graphics on a monitor. Applications make requests to the kernel and in return receive resources, such as memory, CPU, and disk space. If two applications request the same resource, the kernel decides which one gets it, and in some cases, kills off another application to save the rest of the system and prevent a crash. The kernel also abstracts some complicated details away from the application. For example, the application doesn’t know if a block of disk storage is on a solid-state drive, a spinning metal hard disk, or even a network file share. Applications need only follow the kernel’s Application Programming Interface (API) and therefore don’t have to worry about the implementation details. Each application behaves as if it has a large block of memory on the system; the kernel maintains this illusion by remapping smaller blocks of memory, sharing blocks of memory with other applications, or even swapping out untouched blocks to disk.

The kernel also handles the switching of applications, a process known as multitasking. A computer system has a small number of central processing units (CPUs) and a finite amount of memory. The kernel takes care of unloading one task and loading a new one if there is more demand than resources available. When one task has run for a specified amount of time, the CPU pauses it so that another may run. If the computer is doing several tasks at once, the kernel is deciding when to switch focus between tasks. With the tasks rapidly switching, it appears that the computer is doing many things at once. When we, as users, think of applications, we tend to think of word processors, web browsers, and email clients, however, there are a large variety of application types. The kernel doesn’t differentiate between a user-facing application, a network service that talks to a remote computer, or an internal task. From this, we get an abstraction called a process. A process is just one task that is loaded and tracked by the kernel. An application may even need multiple processes to function, so the kernel takes care of running the processes, starting and stopping them as requested, and handing out system resources.

**Major Applications**

The Linux kernel can run a wide variety of software across many hardware platforms. A computer can act as a server, which means it primarily handles data on others’ behalf, or as a desktop, which means a user interacts with it directly. The machine can run software or be used as a development machine in the process of creating software. A machine can even adopt multiple roles as Linux makes no distinction; it’s merely a matter of configuring which applications run. One resulting advantage is that Linux can simulate almost all aspects of a production environment, from development to testing, to verification on scaled-down hardware, which saves costs and time. A Linux administrator could run the same server applications on a desktop or inexpensive virtual server that are run by large internet service providers. Of course, a desktop would not be able to handle the same volume as a major provider would, but almost any configuration can be simulated without needing powerful hardware or server licensing.

Linux software generally falls into one of three categories:

**Server Applications:** Software that has no direct interaction with the monitor and keyboard of the machine it runs on. Its purpose is to serve information to other computers, called clients. Sometimes server applications may not talk to other computers but only sit there and crunch data.

**Desktop Applications:** Web browsers, text editors, music players, or other applications with which users interact directly. In many cases, such as a web browser, the application is talking to a server on the other end and interpreting the data. This is the “client” side of a client/server application.

**Tools:** A loose category of software that exists to make it easier to manage computer systems. Tools can help configure displays, provide a Linux shell that users type commands into, or even more sophisticated tools, called compilers, that convert source code to application programs that the computer can execute.

The availability of applications varies depending on the distribution. Often application vendors choose a subset of distributions to support. Different distributions have different versions of key libraries, and it is difficult for a company to support all these different versions. Some applications, however, like Firefox and LibreOffice are widely supported and available for all major distributions.

The Linux community has come up with lots of creative solutions for both desktop and server applications. These applications, many of which make up the backbone of the Internet, are critical to understanding, and utilizing the power of Linux. Most computing tasks can be accomplished by any number of applications in Linux. There are many web browsers, web servers, database servers, and text editors from which to choose. Evaluating application software is an important skill to be learned by the aspiring Linux administrator. Determining requirements for performance, stability, and cost are just some of the considerations needed for a comprehensive analysis.

**Завдання для попередньої підготовки.**

1. \*Прочитайте короткі теоретичні відомості до лабораторної роботи та зробіть невеликий словник базових англійських термінів з питань призначення команд та їх параметрів.
2. Вивчіть матеріали онлайн-курсу академії Cisco “NDG Linux Essentials”:

* Chapter 3 - Working in Linux
* Chapter 4 - Open Source Software and Licensing

1. Пройдіть тестування у курсі NDG Linux Essentials за такими темами:

* Chapter 03 Exam
* Chapter 04 Exam

1. Дайте визначення наступним поняттям:

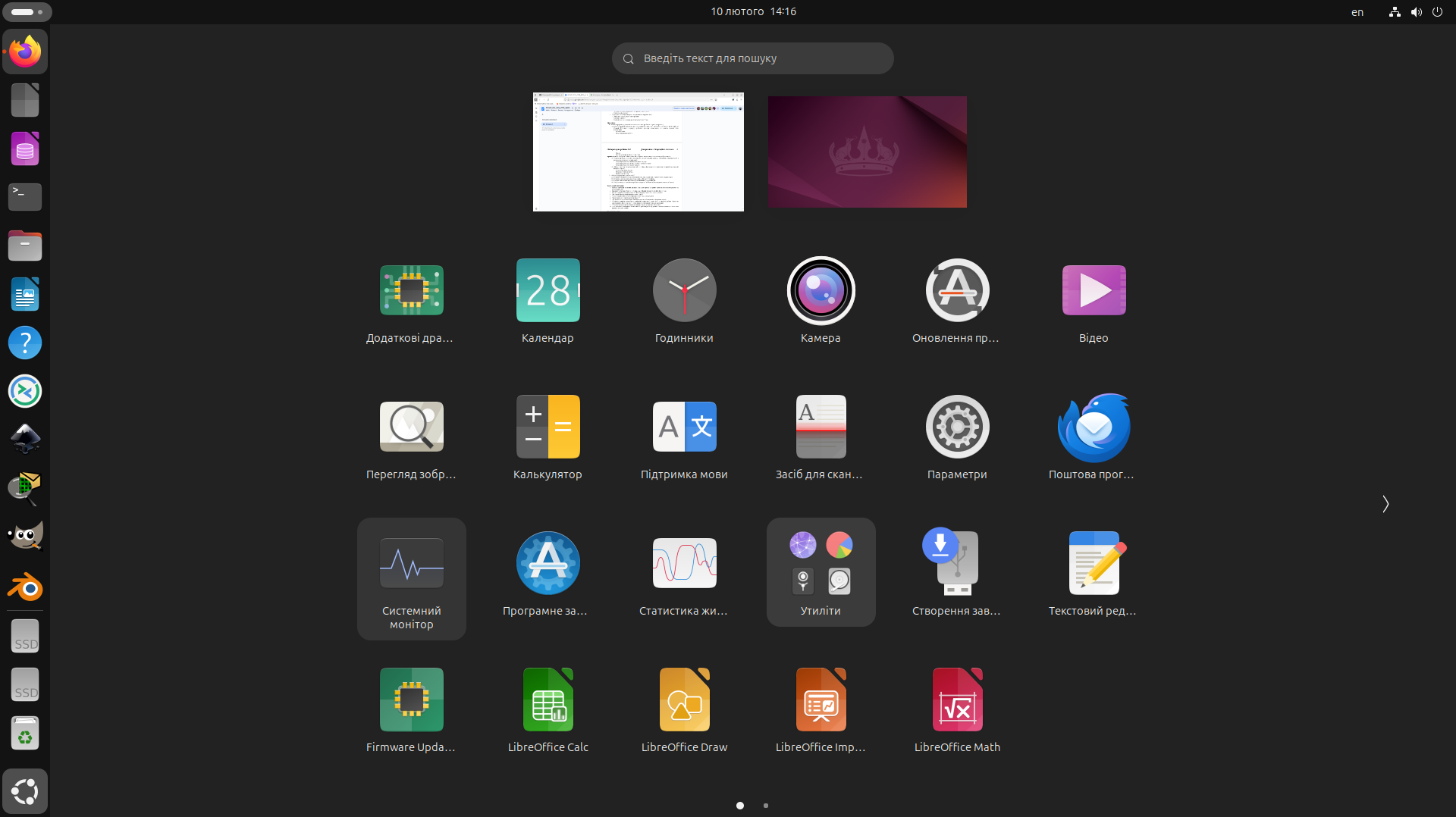
* CLI-режим
* Термінал на основі графічного інтерфейсу користувача
* Віртуальний термінал
* **CLI-режим (Command Line Interface mode)**  
  Режим роботи операційної системи, у якому користувач взаємодіє з комп’ютером через текстові команди, що вводяться з клавіатури, без використання графічних елементів.
* **Термінал на основі графічного інтерфейсу користувача (GUI-terminal)**  
  Програма у графічному середовищі, яка відкриває вікно емуляції консолі та дозволяє виконувати текстові команди так само, як у звичайному командному рядку.
* **Віртуальний термінал (Virtual terminal)**  
  Окрема текстова консоль, що працює паралельно з графічним середовищем і потребує входу користувача; дозволяє виконувати команди незалежно від GUI.

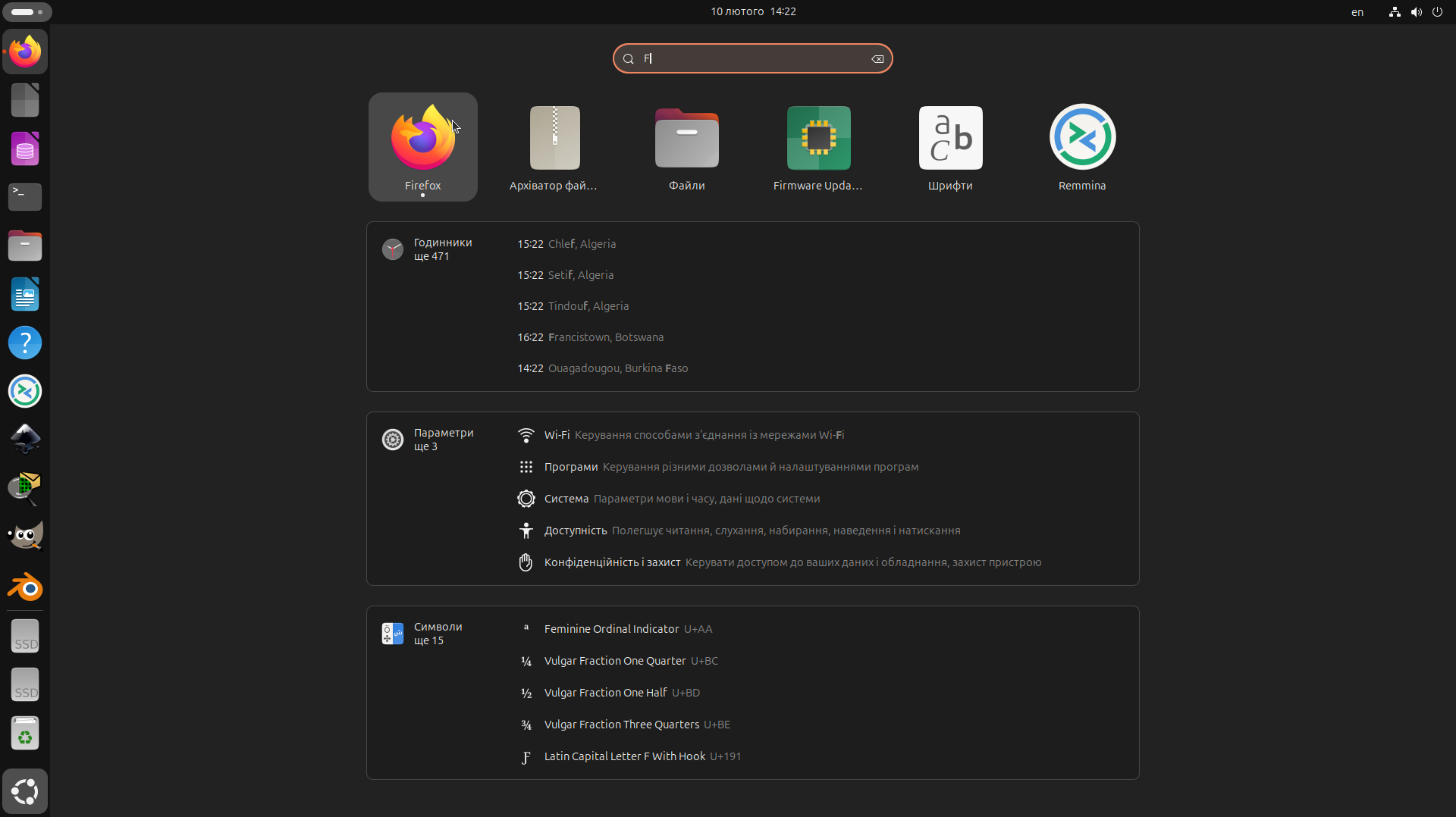
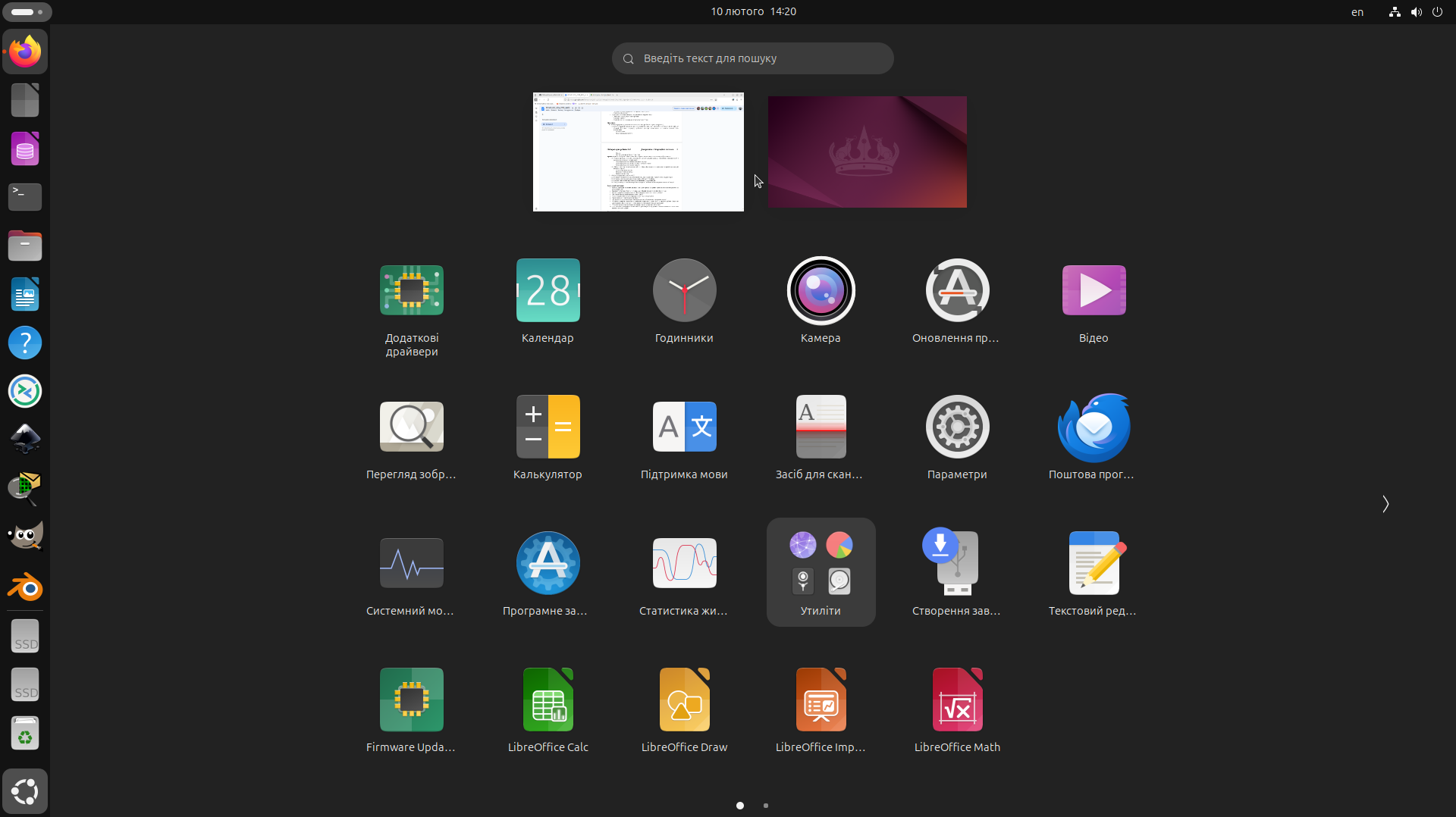
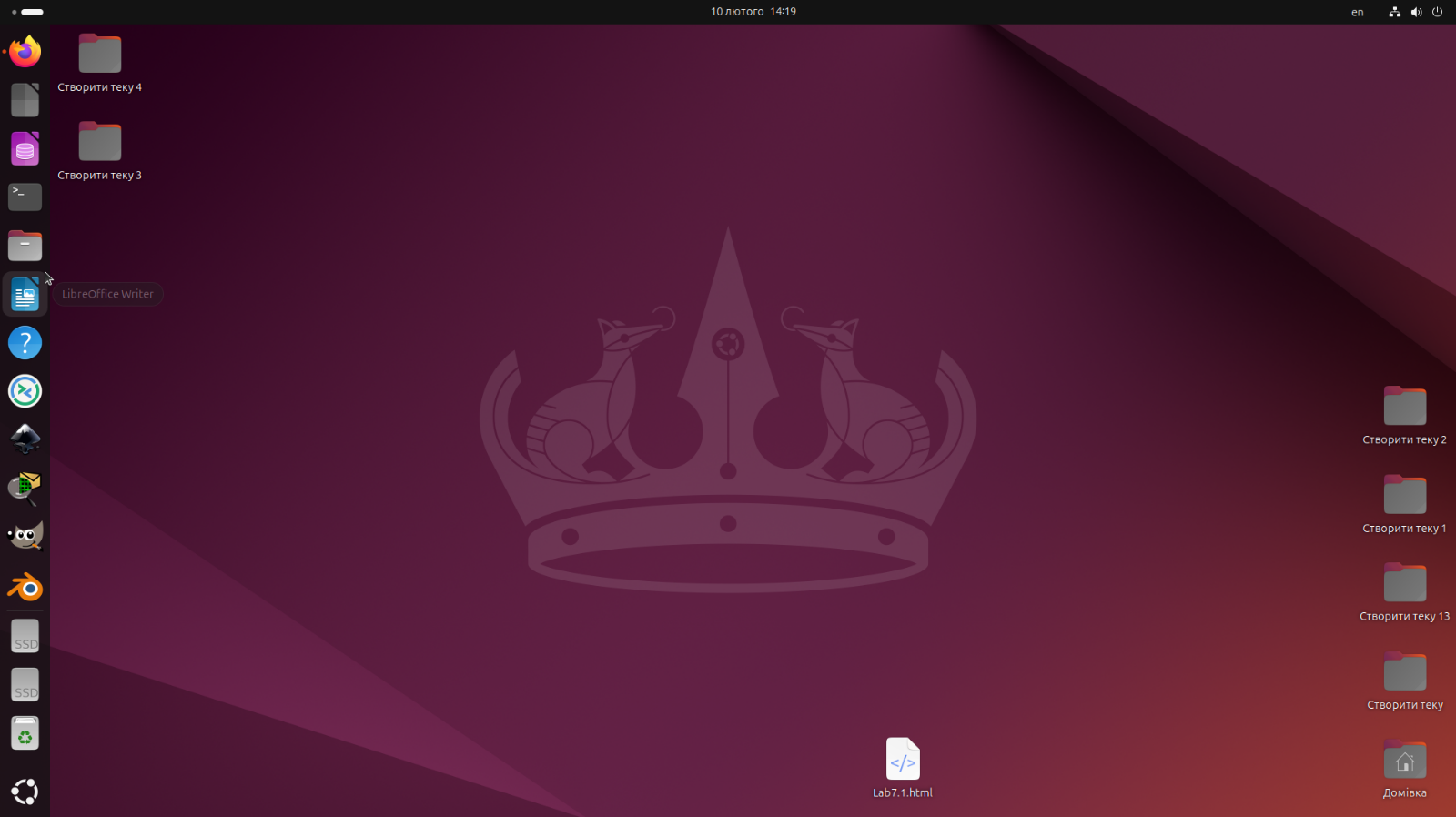
**Словник**

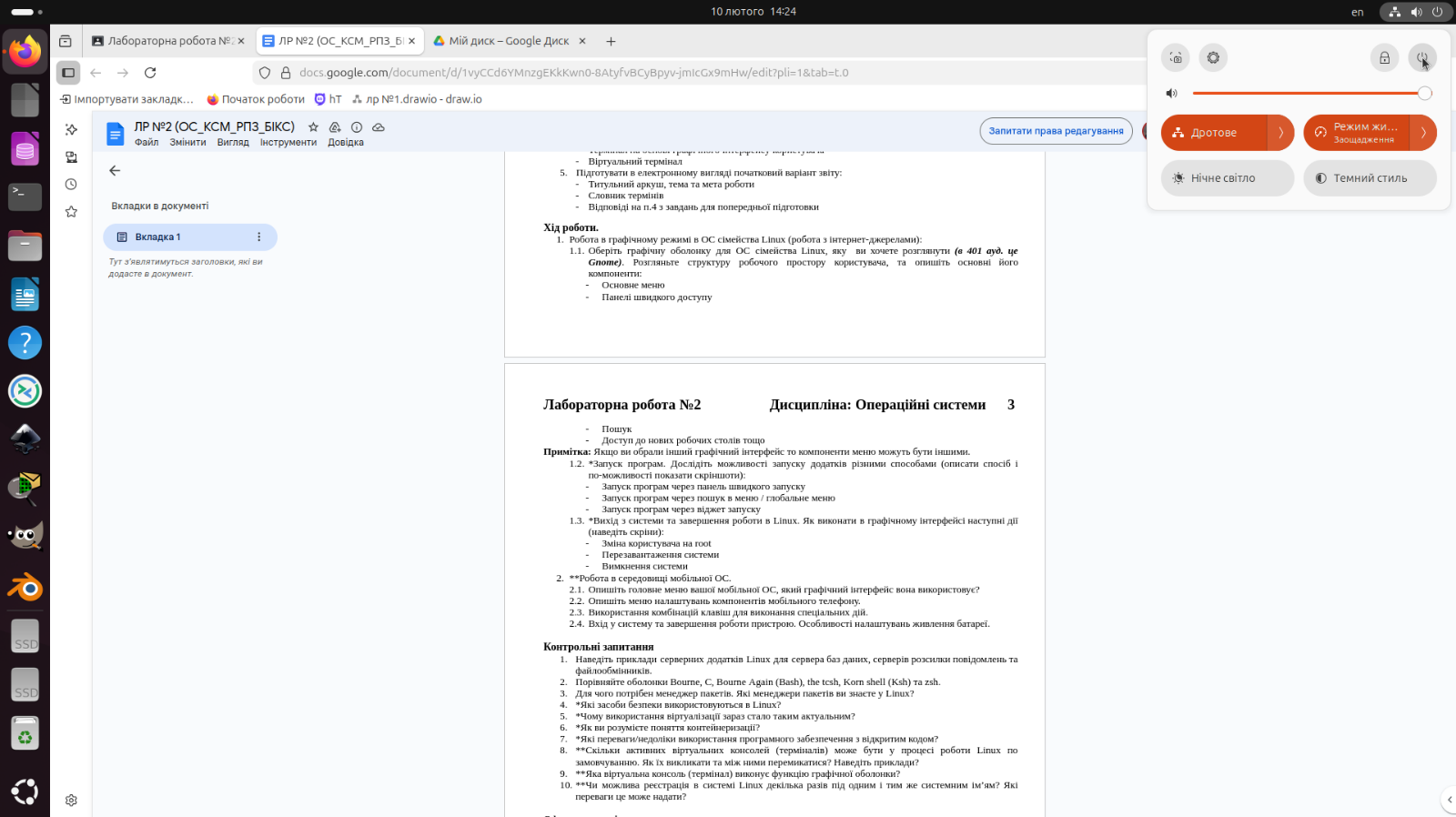
|  |  |
| --- | --- |
| **Command Line Interface (CLI)** | A text-based interface where a user types commands to interact directly with the operating system. |
| **Terminal** | A program or environment that allows users to access the command line and execute commands. |
| **Kernel** | The core part of an operating system that manages hardware resources such as CPU, memory, and devices, and controls running programs. |
| **Process** | A single running task or instance of a program that is managed and tracked by the kernel. |
| **Multitasking** | The ability of the operating system to run multiple processes by rapidly switching CPU attention between them. |
| **Application Programming Interface (API)** | A defined set of rules and functions that applications use to request services from the operating system kernel. |
| **Server Application** | Software that provides data or services to other computers (clients) over a network without direct user interaction. |

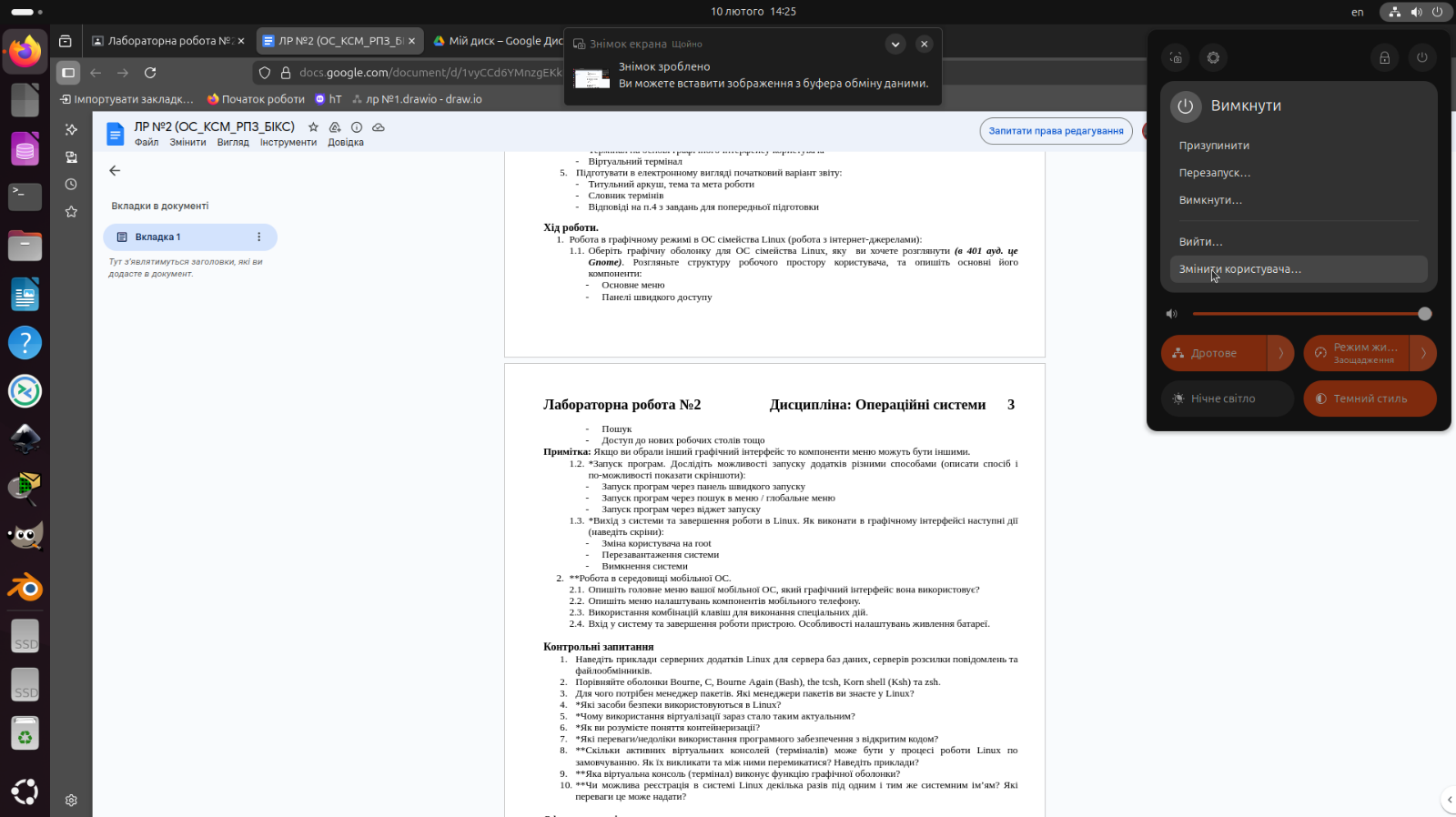
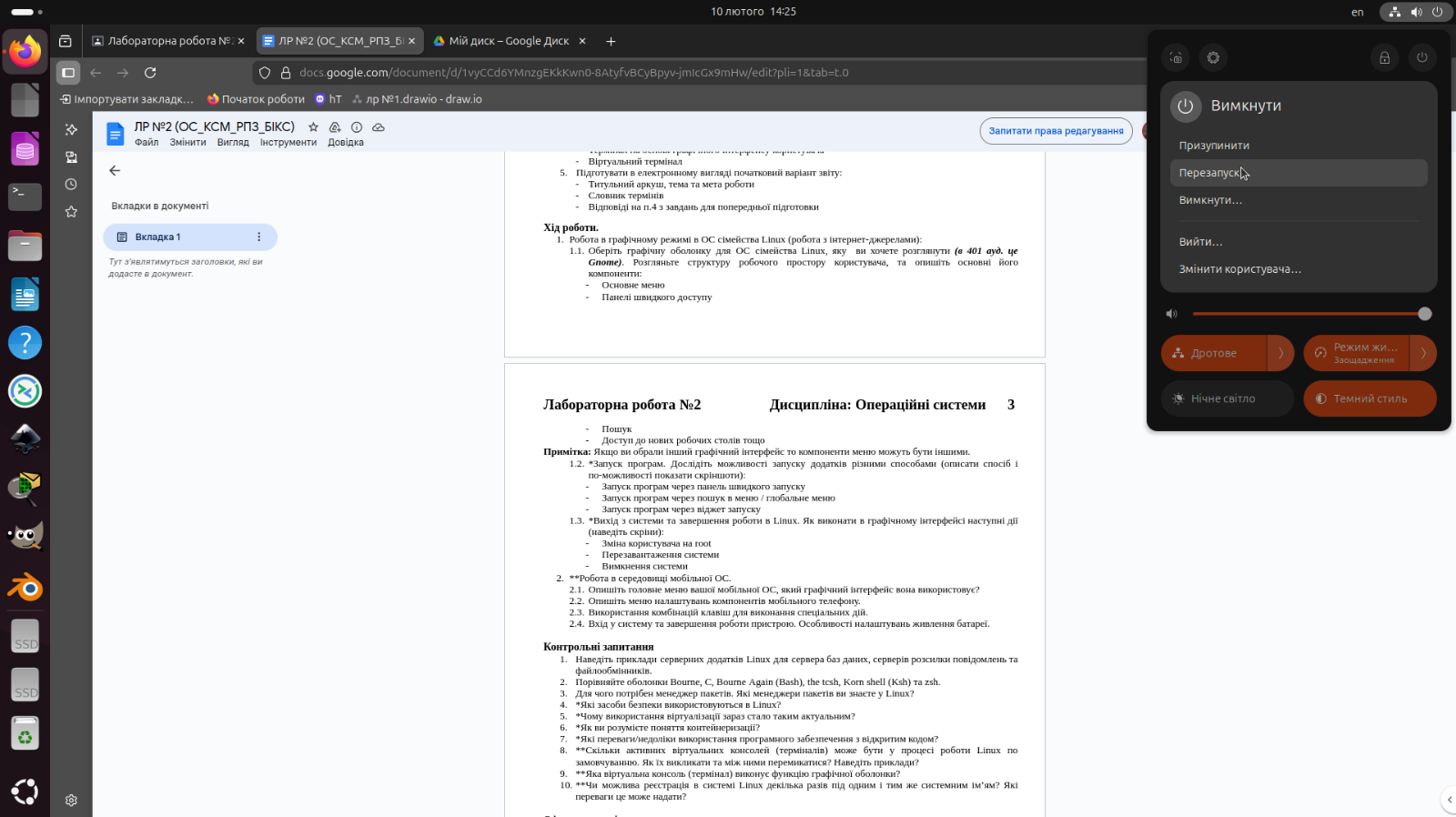
**Хід роботи.**

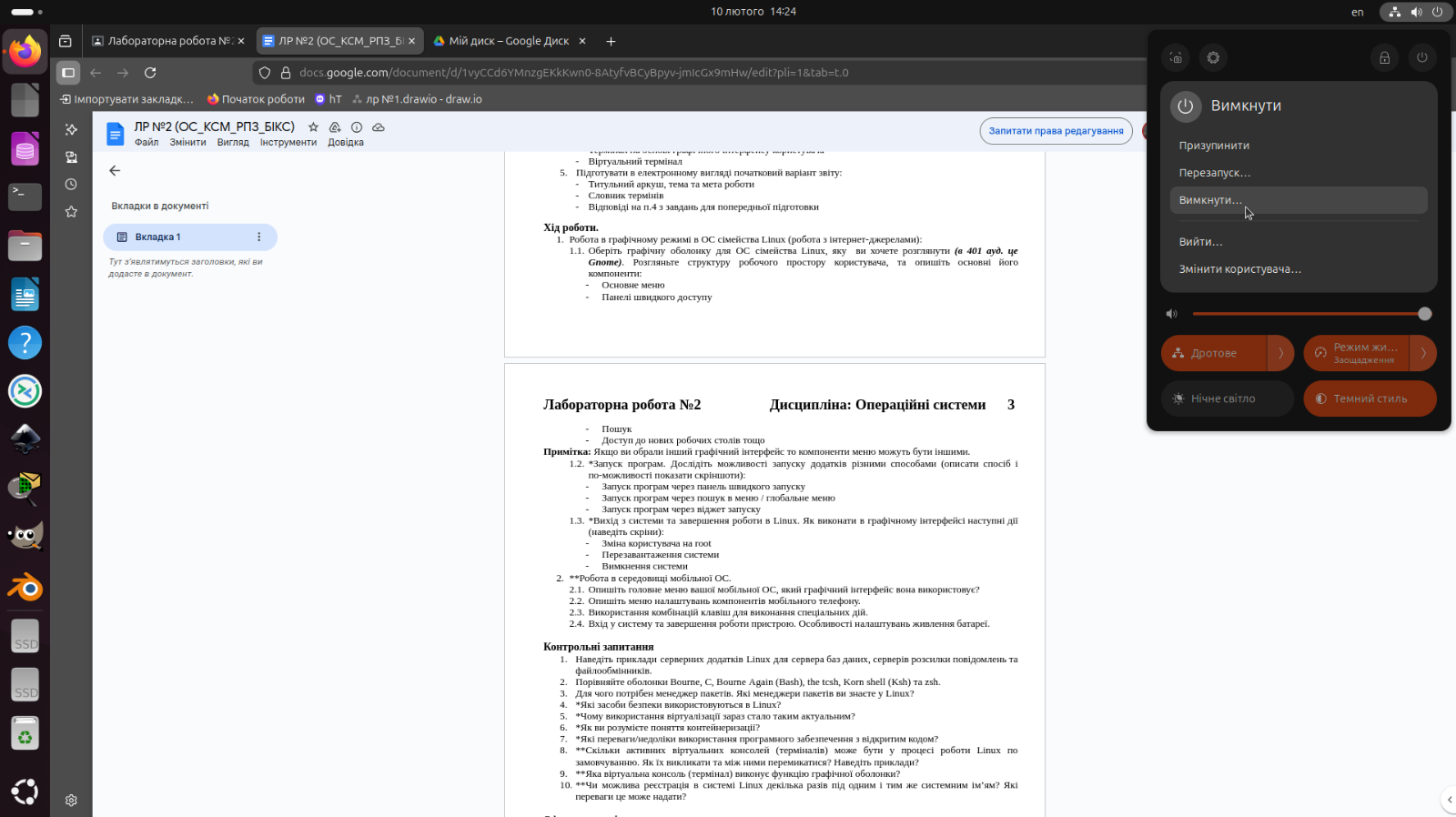
* 1. Робота в графічному режимі в ОС сімейства Linux (робота з інтернет-джерелами):
  2. Оберіть графічну оболонку для ОС сімейства Linux, яку ви хочете розглянути ***(в 401 ауд. це Gnome)***. Розгляньте структуру робочого простору користувача, та опишіть основні його компоненти:
* Основне меню
* Панелі швидкого доступу
* Пошук
* Доступ до нових робочих столів тощо



* 1. \*Запуск програм. Дослідіть можливості запуску додатків різними способами (описати спосіб і по-можливості показати скріншоти):
* Запуск програм через панель швидкого запуску
* Запуск програм через пошук в меню / глобальне меню
* Запуск програм через віджет запуску 
  1. \*Вихід з системи та завершення роботи в Linux. Як виконати в графічному інтерфейсі наступні дії (наведіть скріни):
* Зміна користувача на root
* Перезавантаження системи
* Вимкнення системи







1. \*\*Робота в середовищі мобільної ОС.

## Робота в середовищі мобільної ОС — ****Android (остання версія)****

### ****2.1. Головне меню та графічний інтерфейс****

Головний екран (Home Screen) — це робочий простір користувача.

**Особливості інтерфейсу:**

* сенсорний (touch UI) — керування жестами і свайпами;
* робочі столи з ярликами програм;
* віджети (погода, годинник, календар);
* панель повідомлень (Notification Shade);
* панель швидких налаштувань (Quick Settings);
* меню всіх додатків (App Drawer);
* нижня панель навігації: жести або кнопки (Назад / Додому / Останні).

Інтерфейс орієнтований на роботу однією рукою та швидкий доступ до функцій.

### ****2.2. Меню налаштувань компонентів телефону****

У додатку **Settings** налаштовуються всі апаратні та програмні компоненти:

**Основні розділи:**

* **Network & Internet** — Wi-Fi, мобільна мережа, точка доступу
* **Connected devices** — Bluetooth, USB, NFC
* **Apps** — дозволи, сповіщення, фонова робота
* **Display** — яскравість, темна тема, частота оновлення
* **Sound & vibration** — гучність, режими дзвінка
* **Security & privacy** — PIN, відбиток, Face Unlock, дозволи
* **Battery** — використання енергії, економія заряду
* **Storage** — пам’ять пристрою
* **System** — мова, дата, оновлення ОС

### ****2.3. Комбінації клавіш (апаратні кнопки)****

Основні швидкі дії:

* **Power + Volume Down** → скріншот
* **Power (утримання)** → меню живлення
* **Power + Volume Up** → Recovery / Boot меню (на більшості пристроїв)
* **Двічі Power** → запуск камери
* **Довге Volume Down** → беззвучний режим
* **3× Power** → екстрений виклик (SOS) на деяких моделях

### ****2.4. Вхід у систему та завершення роботи. Живлення батареї****

**Вхід у систему:**

* PIN-код
* графічний ключ (Pattern)
* пароль
* відбиток пальця
* розпізнавання обличчя

**Завершення роботи:**

* Перезавантаження (Restart)
* Вимкнення (Power off)
* Режим блокування (Lockdown — без біометрії)

**Налаштування батареї:**

* режим енергозбереження (Battery Saver)
* адаптивна батарея (обмежує фонові процеси)
* обмеження роботи додатків у фоні
* оптимізація заряджання (захист акумулятора)
* статистика використання енергії по додатках

**Контрольні запитання**

## 1. Приклади серверних додатків Linux

**Сервери баз даних:**

* MySQL
* PostgreSQL
* MariaDB

**Сервери розсилки повідомлень (mail):**

* Postfix
* Sendmail
* Exim

**Файлообмін (file servers):**

* Samba
* vsftpd
* NFS

## 2. Порівняння оболонок

| **Оболонка** | **Особливості** |
| --- | --- |
| **Bourne shell (sh)** | Найпростіша класична оболонка UNIX, базова сумісність |
| **C shell (csh)** | Синтаксис схожий на C, зручна для інтерактивної роботи |
| **Bash** | Розширення sh, найпопулярніша, автодоповнення, історія команд |
| **tcsh** | Покращена версія csh, автодоповнення та редагування команд |
| **Korn shell (ksh)** | Швидка, стабільна, використовується в корпоративних UNIX |
| **zsh** | Найфункціональніша: теми, плагіни, розумне автодоповнення |

## 3. Менеджер пакетів

**Призначення:**  
Система встановлення, оновлення, видалення та керування залежностями програм.

**Відомі менеджери:**

* **APT / dpkg** (Debian, Ubuntu)
* **DNF / YUM / RPM** (Fedora, RHEL, CentOS)
* **Pacman** (Arch Linux)
* **Zypper** (openSUSE)
* **Snap, Flatpak, AppImage** (універсальні пакети)

## 4. Засоби безпеки в Linux

* права доступу (rwx, власник/група)
* sudo / root-доступ
* SELinux / AppArmor
* firewall (iptables, nftables)
* шифрування диску (LUKS)
* SSH-ключі
* sandboxing процесів

## 5. Чому актуальна віртуалізація

* економія обладнання
* ізоляція середовищ
* тестування без ризику
* хмарні сервіси
* масштабування сервісів

## 6. Контейнеризація

Контейнеризація — це запуск програм в ізольованих середовищах (контейнерах), які використовують спільне ядро ОС, але мають власні бібліотеки та залежності.  
Легша і швидша за віртуальні машини.

## 7. Open Source: плюси/мінуси

**Переваги:**

* безкоштовність
* безпека (відкритий код)
* гнучкість
* велика спільнота

**Недоліки:**

* складніше налаштування
* іноді немає офіційної підтримки
* сумісність програм

## 8. Кількість віртуальних консолей

**За замовчуванням: 6 текстових + 1 графічна**

**Перемикання:**

* Ctrl + Alt + F1 … F6 → текстові
* Ctrl + Alt + F7 (або F2/F1 у нових дистрибутивах) → GUI

## 9. Яка консоль є графічною

Графічна оболонка працює у **tty7** (інколи tty1 або tty2 у сучасних системах з systemd).

## 10. Кілька входів під одним ім’ям

**Так, можливо.**

**Переваги:**

* адміністрування з різних терміналів
* віддалений + локальний доступ одночасно
* робота з кількома задачами паралельно

In this work we examined the basic principles of working in a Linux environment, including server applications, shells, package managers, security mechanisms, virtualization, and containerization.  
Linux provides a flexible and powerful system where resources are efficiently managed and multiple users and processes can work simultaneously.

Virtualization and containerization make modern infrastructure scalable and easy to maintain, while open-source software gives freedom of modification and transparency of operation.  
Overall, Linux remains one of the most reliable operating systems for servers, development, and administration due to its stability, security, and configurability.