ChatGPT

Escribir un libro con LaTeX en Ubuntu desde la terminal

Instalación de LaTeX en Ubuntu

Para preparar un libro en LaTeX, primero necesitas instalar una distribución de LaTeX en tu sistema Ubuntu. La opción típica es **TeX Live**. Puedes instalar el metapaquete completo (que incluye *todas* las herramientas y paquetes) ejecutando en la terminal:

```
sudo apt update
sudo apt install texlive-full
```

Esto instalará el compilador LaTeX (por ejemplo `pdflatex`, `xelatex`, etc.) y una amplia colección de paquetes adicionales (aprox. 6 GB de datos) [1]. La instalación completa garantiza que tendrás **todas** las funcionalidades necesarias (soporte para imágenes, índices, referencias, *fonts* especiales, conversiones a EPUB, etc.) sin tener que añadir paquetes manualmente más adelante. *Nota:* Si el espacio es un problema, podrías instalar una selección más básica (`texlive` a secas o `texlive-latex-recommended`), pero en ese caso puede que debas añadir paquetes extra según los vayas necesitando [1]. En general, para evitar complicaciones se suele preferir la instalación completa con `texlive-full` [2].

Además de LaTeX en sí, puedes instalar un visor de PDF para revisar el resultado. Por ejemplo, **Evince** (visor de PDF de GNOME) se instala con `sudo apt install evince` [3], o puedes usar Okular, etc. Para formato EPUB, podrías instalar **FBReader** u otro lector de eBooks (`sudo apt install fbreader`) para previsualizar los EPUB antes de subirlos a Amazon [2].

Edición de texto en la terminal

Los documentos LaTeX son simples archivos de texto plano (extensión `.tex`), por lo que **puedes usar cualquier editor de texto**. Dado que prefieres un editor en terminal, **Nano** es perfectamente válido para escribir y editar tus archivos `.tex`. Simplemente crea los archivos `.tex` con Nano (u otro editor CLI como Vim o Emacs, si lo deseas). No necesitas un editor gráfico especializado – aunque existen (TeXstudio, Texmaker, etc.) – ya que tu flujo de trabajo será 100% en la terminal. Esto significa que todo el contenido es legible y modificable como texto, ideal si planeas automatizar tareas con scripts. *(De hecho, al ser texto plano y usar comandos CLI para compilar, puedes integrar fácilmente herramientas de automatización o asistentes de línea de comandos como Codex CLI para ejecutar tareas automáticamente.)*

Recuerda guardar tus archivos `.tex` con codificación **UTF-8** para evitar problemas con caracteres especiales (tildes, eñes, etc.). LaTeX moderno por defecto asume UTF-8, pero por seguridad puedes incluir en el preámbulo del documento `\usepackage[utf8]{inputenc}` (si usas PDFLaTeX) y `\usepackage[T1]{fontenc}` para un manejo correcto de acentos y caracteres en español. También es

aconsejable cargar el paquete de idioma español: `\usepackage[spanish]{babel}`, para que elementos como los títulos de capítulos, fechas, numeración y guiones se adapten al español. Por ejemplo, Babel traducirá "Chapter" a "Capítulo" automáticamente y ajustará la separación silábica del español correctamente.

Organización del proyecto (múltiples archivos y carpetas)

Para un libro con varios capítulos, es buena idea organizar el proyecto en múltiples archivos y carpetas. Una posible estructura recomendada sería:

- **Directorio raíz del proyecto**: crea una carpeta que contendrá todo el libro (por ejemplo, `milibro/`).
- Dentro de esta, puedes tener una subcarpeta para los textos LaTeX, por ejemplo `milibro/tex/` donde estarán los archivos `.tex` de cada capítulo, prólogo, apéndices, etc. (esto ayuda a mantener el proyecto ordenado) [4].
- Otra subcarpeta para **imágenes**, por ejemplo `milibro/img/`, donde colocarás los archivos gráficos (PNG, JPG o PDF) que vayas a incluir [4].
- En el directorio raíz estará el archivo principal, por ejemplo `milibro.tex`, que será el **documento maestro** que reúne todo el libro, así como otros archivos auxiliares (por ej., puedes tener un archivo de estilo personalizado `.sty` si lo necesitas, aunque no es obligatorio).

Archivos por capítulo: Es recomendable dividir cada capítulo o sección importante en un archivo `.tex` separado [5]. Por ejemplo, podrías tener `capitulo1.tex`, `capitulo2.tex`, etc., almacenados en la carpeta `tex/`. Esto hace más manejable la edición (cada capítulo por separado) y te permite reusar o reordenar capítulos fácilmente en diferentes compilaciones si lo necesitaras. LaTeX facilita esta modularidad mediante los comandos `\input{...}` o `\include{...}` para **incluir** el contenido de un archivo dentro del principal [5]. En tu archivo maestro `milibro.tex`, podrás escribir algo así:

```
\documentclass[11pt,openright,twoside]{book} % documento tipo libro, 11pt de
tamaño de fuente
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage[spanish]{babel}
\usepackage{graphicx}    % para insertar imágenes
\usepackage{hyperref}    % opcional, para enlaces clicables en PDF (índice,
refs)
% (aquí podrías cargar más paquetes si los necesitas, p.ej. makeidx para índice
alfabético, etc.)

\title{Título del Libro}
\author{Tu Nombre}

\begin{document}
\frontmatter % opcional: si quieres páginas de intro sin numeración de capítulo
\maketitle   % genera portada con título/autor (necesita \title y \author
definidos)
```

```
\tableofcontents            % genera el índice (tabla de contenidos)
automáticamente [6]

\mainmatter % aquí comienzan los capítulos numerados

\include{tex/capitulo1}     % incluye el contenido de capitulo1.tex [7]
\include{tex/capitulo2}     % incluye el contenido de capitulo2.tex (y así
sucesivamente) [7]

\appendix % si tienes apéndices, opcional
%\include{tex/apendiceA}

\backmatter % para bibliografía o índices, si hubiera
%\printindex % si hubieras creado un índice alfabético de términos

\end{document}
```

En el ejemplo anterior, usamos `\include{...}` para anexar capítulos completos. La diferencia entre `\include` y `\input` es sutil: `\include` maneja ciertos aspectos (como saltos de página y archivos auxiliares propios) y no permite anidar includes dentro de otros, mientras que `\input` simplemente inserta el contenido tal cual [8]. Para capítulos separados, cualquiera de los dos sirve; `\include` es común para capítulos. (Si tienes un documento no tan extenso, también podrías no usar carpetas separadas y guardar todo en unos pocos `.tex`, pero la separación propuesta te da más flexibilidad para reorganizar o compilar selectivamente capítulos.)

Consejo: Es importante que las imágenes que incluyas estén **dentro del directorio del proyecto o sus subdirectorios**, no referenciadas con rutas absolutas fuera de él. Si las imágenes están fuera, LaTeX podría no encontrarlas al compilar, y en particular al convertir a eBook podrías perder esas imágenes en el EPUB final [9] (por ejemplo, Amazon KDP no incluirá imágenes vinculadas fuera del paquete del libro). Lo habitual es guardar todas las imágenes en la carpeta `img/` (u otra dentro del proyecto) y referenciarlas por su ruta relativa.

Escritura del contenido en LaTeX (capítulos, imágenes, referencias)

Dentro de cada archivo de capítulo (por ejemplo `capitulo1.tex`), escribirás el texto utilizando la sintaxis de LaTeX. Algunos puntos clave para lograr las características que buscas (índice, referencias internas, citas, etc.):

- **Capítulos y secciones:** En documentos de clase `book` o `report`, usar `\chapter{Título del Capítulo}` para iniciar cada capítulo. Dentro de capítulos puedes usar `\section{...}`, `\subsection{...}`, etc. para subdividir. LaTeX numerará automáticamente capítulos y secciones, y esos títulos aparecerán en el **índice** general (tabla de contenidos). Al haber incluido `\tableofcontents` en el documento principal, LaTeX generará automáticamente el índice/tabla de contenido con los números de página de cada capítulo y sección [6]. (Este índice se actualiza en la segunda compilación, más sobre compilación abajo.)

- **Índice (Tabla de contenido):** Como se indicó, `\tableofcontents` produce una tabla de contenidos automática. No necesitas escribirla manualmente; LaTeX va llenándola con los títulos de capítulos/secciones conforme aparecen en el documento. Asegúrate de compilar dos veces al menos para que se generen los números de página correctos en el índice.

- **Referencias cruzadas internas:** Puedes crear **referencias internas** (cross-references) dentro del texto fácilmente. Por ejemplo, si quieres referirte a un capítulo o sección más adelante, usa `\label{etiqueta}` justo después del título de ese capítulo/ sección, y luego `\ref{etiqueta}` donde quieras citarlo. LaTeX reemplazará `\ref` por el número correspondiente. Lo mismo aplica para figuras y tablas (colocas `\label{fig:loquesea}` dentro de `\begin{figure}... \end{figure}` después del \caption, y luego `\ref{fig:loquesea}` dará el número de figura). Esto te permite decir cosas como "En el *Capítulo 3* se discute X" sin escribir tú el número. LaTeX también puede generar listas de figuras o tablas automáticamente con `\listoffigures` o `\listoftables` si las necesitas.

- **Imágenes y gráficos:** Para incluir imágenes en el documento, asegúrate de cargar el paquete **graphicx** (`\usepackage{graphicx}` ya lo pusimos en el preámbulo). Luego, en el lugar del texto donde quieras la imagen, puedes usar:

```
\begin{figure}[htb]
  \centering
  \includegraphics[width=0.8\textwidth]{img/mi_imagen.png}
  \caption{Leyenda o descripción de la imagen.}
  \label{fig:mi_imagen}
\end{figure}
```

Esto inserta la imagen `mi_imagen.png` (buscada en la carpeta `img/`) escalada al 80% del ancho del texto. La figura tendrá un número automático y la leyenda indicada debajo. Con `\centering` la centramos. Más tarde podrás referirte a "Figura \ref{fig:mi_imagen}" en el texto, y LaTeX pondrá el número correcto. Si no necesitas numeración ni leyenda, puedes omitir el entorno `figure` y solo usar `\includegraphics` (por ejemplo para imágenes decorativas). Ten en cuenta que LaTeX por defecto coloca las figuras en la posición óptima según su algoritmo (*posiblemente no exactamente donde la pones en el código*, a menos que uses opciones como [h] que indica "here"). En un libro de texto plano (poemas, etc.) quizás quieras las imágenes en posiciones específicas; puedes experimentar con la opción [H] de `\usepackage{float}` para forzar la posición exacta, si fuera necesario. En cualquier caso, **verifica en el PDF** que las imágenes queden donde quieres.

- **Citas bibliográficas o notas:** Si en tus "reflexiones" necesitas citar fuentes externas o añadir notas, LaTeX ofrece varias opciones. Dado que mencionaste que *no vas a manejar bibliografía extensa*, probablemente no necesites un archivo `.bib` ni usar BibTeX. Puedes simplemente usar **notas al pie** con `\footnote{texto de la nota}` para aclaraciones o referencias cortas. LaTeX numerará automáticamente las notas al pie en cada página. Si quisieras incluir una lista de referencias bibliográficas al final, podrías usar el entorno `thebibliography` o integrar BibTeX/BibLaTeX, pero para un libro de poemas/reflexiones personales esto seguramente no es necesario (y complica el flujo). En la experiencia de otro autor, cuando necesitó referencias en un eBook de LaTeX, optó por

ponerlas como notas al pie en lugar de estilo IEEE por simplicidad [10]. Así que para citas textuales o referencias breves, una footnote es suficiente.

- **Índice alfabético (opcional):** Si necesitas un índice analítico (por temas, al final del libro), puedes generarlo con LaTeX también. Esto requiere incluir `\usepackage{makeidx}` en el preámbulo, luego poner `\makeindex` antes de `\begin{document}`, marcar términos con `\index{palabra}` en el texto, y finalmente al final del documento llamar a `\printindex`. Después de compilar, tendrías que ejecutar la herramienta `makeindex` para generar el índice. Sin embargo, en un libro de poemas probablemente no necesites un índice alfabético de términos, así que esto es solo si se requiriera. La tabla de contenidos (índice general) es normalmente suficiente para libros de prosa, poemas, etc.

En resumen, LaTeX te permitirá lograr **un resultado de alta calidad profesional**, con todos estos elementos (índice, imágenes, formateo consistente). De hecho, LaTeX produce documentos con una calidad tipográfica excepcional, superior a la de procesadores de texto como Word [11]. Por ejemplo, manejará bien la justificación del texto, la separación de palabras, fuentes escalables, etc., evitando problemas típicos de Word (espaciados irregulares, fuentes estiradas, formato inconsistente al mover imágenes) [12]. Muchos libros, tesis y documentos profesionales se componen en LaTeX justamente por esa calidad en el resultado final.

Compilación del documento a PDF en terminal

Una vez que tengas tu contenido escrito en los archivos `.tex`, el siguiente paso es **compilar** el documento para generar el PDF. Esto se hace mediante herramientas de línea de comando incluidas con TeX Live:

- La forma más directa: posiciona tu terminal en el directorio del proyecto (donde está tu archivo principal, e.g. `milibro.tex`) y ejecuta:

```
pdflatex milibro.tex
```

Esto ejecutará el compilador PDFLaTeX, procesando el archivo maestro y los capítulos incluidos, produciendo un archivo `milibro.pdf`. Si durante la compilación LaTeX necesita múltiples pasadas (por ejemplo, para resolver referencias cruzadas, generar la tabla de contenido con números de página correctos, etc.), **deberás compilar varias veces**. En la práctica, para obtener un PDF final correcto se suele ejecutar `pdflatex` **dos veces** como mínimo. Por ejemplo, la primera pasada genera el índice (TOC) incompleto y marca referencias pendientes, la segunda pasada llena esos datos. Si hubiera bibliografía o índice alfabético, incluso se requerirían pasos adicionales (ejecutar `bibtex` o `makeindex` en medio). Un ciclo típico de compilación completo podría ser: `pdflatex -> bibtex (si aplica) -> makeindex (si aplica) -> pdflatex -> pdflatex` [13]. En tu caso, sin bibliografía, bastará con dos pasadas de `pdflatex` para asegurarse de que el índice y las referencias internas queden actualizados.

- **Herramienta *latexmk* (opcional):** En lugar de ejecutar manualmente varias veces, puedes usar `latexmk`, una utilidad que automatiza el proceso de compilación. Viene incluida en muchas instalaciones TeX Live. Con el comando `latexmk -pdf milibro.tex`, la herramienta detectará

automáticamente cuántas pasadas son necesarias (ejecutará pdflatex las veces requeridas, llamará a bibtex o makeindex si hace falta, etc.) [14]. Al final, obtendrás el `milibro.pdf`. Esto simplifica mucho la tarea. Si `latexmk` no estuviera instalado por defecto, puedes instalarlo vía `sudo apt install latexmk`.

- **Errores y depuración:** Durante la compilación, LaTeX mostrará mensajes en la terminal. Si hay *errores* (por ej., sintaxis incorrecta, alguna imagen no encontrada, etc.), la compilación se detendrá esperando tu intervención. En Nano no los verás mientras editas, pero en la terminal podrás leerlos. Presta atención a esos mensajes para corregir tu `.tex` en caso de errores. Warnings (avisos) también pueden aparecer, por ejemplo sobre referencias no definidas (tras la primera pasada) o sobre *Overfull hbox* (líneas muy largas); estos últimos no impiden generar el PDF pero conviene revisarlos para ajustar formato si es necesario.

Tras compilar con éxito, obtendrás un **PDF de alta calidad** con tu libro. Este PDF puedes abrirlo con un visor (Evince, Okular, etc.) para revisar el formato, o incluso imprimirlo. Ten en cuenta que si planeas publicar en **Amazon KDP** una versión en papel (*paperback*), Amazon te pedirá subir un PDF como manuscrito final para la impresión [15]. La buena noticia es que el PDF de LaTeX cumple perfectamente para eso (Amazon tiene requisitos de tamaño de página y márgenes, los cuales puedes ajustar en LaTeX usando el paquete `geometry` si es necesario). Por ejemplo, podrías configurar `\usepackage[paperwidth=6in, paperheight=9in, margin=...]{geometry}` para adaptar a un tamaño específico de impresión de KDP. En cualquier caso, **KDP acepta PDFs** para libros impresos, así que podrás usar directamente el PDF generado por LaTeX [16]. Asegúrate de *incrustar las fuentes* (fonts) en el PDF – por defecto, pdflatex ya incrusta las fuentes utilizadas, así que no suele haber problema en este aspecto.

Generación de un eBook (EPUB) desde LaTeX

Además del PDF, deseas obtener un **libro electrónico** (formato EPUB) a partir del mismo contenido, para publicar en Amazon (Kindle Direct Publishing para ebooks). Amazon KDP prefiere el formato **EPUB** para libros Kindle [17] (también aceptan .mobi o incluso .docx, pero EPUB es el estándar recomendado). LaTeX no genera EPUB de forma nativa, pero existen herramientas para convertir tu documento LaTeX en un eBook. Tienes principalmente **dos enfoques**:

1. **Usar Pandoc** – *Pandoc* es una herramienta de conversión universal de formatos. Puede tomar un `.tex` y transformarlo en `.epub` (entre muchos otros formatos). La ventaja es que es relativamente simple de usar. Por ejemplo, podrías instalar pandoc (`sudo apt install pandoc`) y luego ejecutar:

```
pandoc -s -o milibro.epub milibro.tex
```

Esto intentará generar `milibro.epub`. Sin embargo, Pandoc tiene que *entender* tu LaTeX, y según la complejidad del documento, puede que algunas cosas no se conviertan perfectamente. En documentos principalmente de texto (como poemas, sin fórmulas matemáticas complejas), pandoc suele funcionar bien. **Nota:** Si usas comandos muy específicos de LaTeX o paquetes inusuales, Pandoc podría ignorarlos o requerir ajustes. En la experiencia de algunos usuarios, Pandoc tuvo dificultades con fórmulas matemáticas en LaTeX [18] (lo cual en tu caso tal vez no aplique). Para texto sencillo debería ir bien. Puedes agregar la

opción `--toc` para que Pandoc incluya la tabla de contenidos en el EPUB, aunque normalmente si tu .tex tiene secciones marcadas, Pandoc las convertirá en capítulos/secciones navegables del eBook. Tras convertir, prueba el EPUB en un lector (por ejemplo, con FBReader o con el *Kindle Previewer* de Amazon) para ver si todo se ve correcto.

1. **Usar tex4ebook (LaTeX a EPUB directo)** – Otra opción muy poderosa es la herramienta **tex4ebook**, que forma parte del entorno TeX (es un derivado de TeX4ht). *Tex4ebook* compila tu documento LaTeX y lo convierte en un EPUB, preservando mejor elementos de LaTeX que Pandoc a veces no maneja. Por ejemplo, tex4ebook tiende a representar correctamente fórmulas matemáticas y varios entornos especiales [19]. En Ubuntu, **tex4ebook viene incluido** en el paquete `texlive-extra-utils` (que ya instalaste con texlive-full) [20], así que seguramente ya lo tienes disponible. Para usarlo, basta un comando:

```
tex4ebook milibro.tex
```

Este comando generará `milibro.epub` en tu directorio actual (junto con varios archivos temporales). Tex4ebook utiliza por detrás un proceso similar a compilar LaTeX varias veces y aplicar hojas de estilo para empaquetar el EPUB. Si todo va bien, obtendrás un EPUB que puedes abrir con tu lector. Por ejemplo, en el blog de Matt Carrick muestran cómo usar tex4ebook y luego abrir el resultado en FBReader automáticamente [21]. Conviene **limpiar los temporales** entre ejecuciones de tex4ebook, ya que este produce muchos archivos (.aux, .log, .png temporales de imágenes rendereadas, etc.) – podrías borrarlos manualmente o mediante un script (ej. `rm *.aux *.log *.png ...`) antes de recompilar para asegurarte de empezar fresco [22].

En ambos métodos, ten en cuenta que los eBooks **no tienen noción de página fija**: el texto es *refluido* según el tamaño de letra y pantalla del dispositivo [23]. Esto significa que algunos refinamientos de maquetación que aplicaste en PDF (como saltos de página manuales, alineación vertical, etc.) pueden no trasladarse al EPUB. Es recomendable mantener el formato del LaTeX **lo más sencillo y semántico posible** cuando vayas a hacer un EPUB. Por ejemplo, evita comandos de espacio forzado, posiciones absolutas, fuentes extrañas – ya que el lector electrónico controlará la apariencia en buena medida [24] [25]. Estilos como *cursiva*, *negrita*, listas, encabezados, etc. sí se traducen bien. Las imágenes se incluirán en el EPUB (como recursos externos) y se escalarán a la pantalla del dispositivo; por eso, asegúrate de que tengan una resolución adecuada y quizás considera versiones en escala de grises si esperas que se lean en e-ink (los Kindle estándar son blanco y negro).

Después de obtener el EPUB, **pruébalo**. Sube el `.epub` al previewer de Amazon KDP (o utiliza la herramienta *Kindle Previewer* de Amazon en tu PC) para ver cómo se muestra en distintos dispositivos [26] [27]. Ajusta en tus fuentes LaTeX cualquier detalle si ves que algo se descoloca en la versión eBook. Puede ser un proceso de iteración, como comenta el autor del blog: lograr que el formato quede correcto en KDP puede requerir prueba y error [28], porque un EPUB no es WYSIWYG como el PDF.

Automatización de tareas de compilación (scripts en Shell)

Dado que tu flujo de trabajo es en la terminal, puedes aprovechar para crear **scripts de shell (bash)** o **Makefiles** que automaticen las tareas frecuentes: compilar el PDF, generar el EPUB, etc. Esto te ahorrará escribir múltiples comandos cada vez y reduce errores.

Por ejemplo, podrías crear un fichero `compilar_pdf.sh` con contenido:

```
#!/bin/bash
pdflatex milibro.tex
pdflatex milibro.tex   # segunda pasada para refs/TOC
echo "PDF compilado."
```

y darle permisos de ejecución (`chmod +x compilar_pdf.sh`). Así, cada vez que quieras actualizar el PDF solo ejecutas `./compilar_pdf.sh`. De igual forma, puedes crear un script `compilar_epub.sh` que contenga los pasos para el eBook. En el blog mencionado antes, usaron un script similar a: primero borrar temporales, luego ejecutar `tex4ebook`, luego abrir el epub en el visor [22] [21]. Un ejemplo simplificado podría ser:

```
#!/bin/bash
rm -f *.aux *.log *.toc *.out *.fls *.xml  # limpiar archivos auxiliares
principales
tex4ebook milibro.tex
echo "EPUB generado."
```

(*Nota:* La lista de temporales a borrar puede ampliarse según veas qué archivos deja tex4ebook en tu proyecto; en el ejemplo de Matt Carrick incluyen también *.png,* .css, *.html, etc. generados durante la conversión [22]).

Si prefieres, un **Makefile** también puede manejar estas tareas con diferentes *targets*. Por ejemplo, un Makefile puede tener un target `pdf` para compilar el PDF, otro `epub` para el eBook, incluso uno `clean` para limpiar temporales. Un fragmento de Makefile ilustrativo para PDF podría ser:

```
all: pdf

pdf:
    pdflatex milibro.tex
    pdflatex milibro.tex

epub:
    tex4ebook milibro.tex

clean:
    rm -f *.aux *.log *.toc *.out *.dvi *.bbl *.blg *.xml *.html *.css *.4ct *.
4tc *.tmp *.xref *.png
```

De este modo, con solo escribir `make pdf` o `make epub` en la terminal dentro del directorio, ejecutas todo el proceso correspondiente automáticamente [13]. Ajusta los comandos según tus necesidades (por ejemplo, si tuvieras bibliografía con BibTeX, insertarías una línea con `bibtex` en el target pdf, etc. [13]).

La ventaja de estos scripts es que también puedes integrarlos en otros flujos de automatización. Por ejemplo, podrías configurar una tarea cron, o invocarlos desde un asistente CLI como mencionaste (Codex CLI u otro), para recompilar rápidamente cuando detectes cambios en los archivos. Dado que todo es texto plano y comandos terminal, es fácil para una herramienta externa leer los archivos y ejecutar el comando necesario.

Conclusión

Resumiendo, para escribir un libro en Ubuntu con LaTeX desde la terminal necesitas instalar TeX Live y usar un editor de texto plano (Nano está bien) para preparar tu contenido en archivos `.tex` modulando capítulos por separado. LaTeX te provee estructuras para manejar capítulos, secciones, imágenes, índices y referencias cruzadas de manera muy robusta. La compilación a PDF se hace con `pdflatex` (directamente desde la terminal), obteniendo un PDF de calidad profesional listo para imprimir o subir a Amazon KDP [16]. Luego, con herramientas adicionales como Pandoc o tex4ebook, puedes convertir ese mismo contenido a un eBook en formato EPUB adecuado para Kindle [17] [19]. Todo este flujo se puede orquestar mediante scripts shell, lo que te permite automatizar tareas como compilar, limpiar archivos temporales, convertir formatos e incluso abrir el resultado, con un solo comando. Siguiendo este proceso, tendrás la versatilidad de reutilizar tus textos (por estar en archivos separados) y la confianza de que el resultado final (sea impreso o digital) tendrá la presentación pulida que esperas.

Fuentes: LaTeX en Ubuntu (instalación TeX Live) [2] [3]; Organización de proyectos LaTeX [4] [5]; Calidad tipográfica de LaTeX vs Word [11] [12]; Publicar en Amazon KDP con LaTeX (PDF/EPUB) [15] [17] [19]; Conversión a EPUB con tex4ebook [20] [21].

---

[1] apt - How can I install latex on Ubuntu 20.04.5 - Ask Ubuntu
https://askubuntu.com/questions/1429846/how-can-i-install-latex-on-ubuntu-20-04-5

[2] [6] [7] [9] [10] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26] [27] [28] Writing an eBook for Amazon Kindle with LaTeX on Linux
https://www.wavewalkerdsp.com/2022/04/27/writing-an-ebook-for-amazon-kindle-with-latex-on-linux/

[3] [13] [15] [16] How to Create a LaTeX Paperback for Sale on Amazon
https://www.wavewalkerdsp.com/2022/05/04/how-to-create-a-latex-paperback-for-sale-on-amazon/

[4] [5] [8] Manual de LaTeX/Introducción a la creación de proyectos en LaTeX - Wikilibros
https://es.wikibooks.org/wiki/Manual_de_LaTeX/Introducci%C3%B3n_a_la_creaci%C3%B3n_de_proyectos_en_LaTeX

[11] [12] Club de Matemáticas YT - Descubre LaTeX
https://sites.google.com/yachaytech.edu.ec/club-de-matemticas-yt/blog/descubre-latex

[14] Download and installation of LaTex for Linux Ubuntu [duplicate]
https://tex.stackexchange.com/questions/289458/download-and-installation-of-latex-for-linux-ubuntu