

## Generative AI Use:

- Used AI as reference to build the front end of the system.
- Used AI (IDE built-in), to fill in comments and doc strings.

## Chat history

You said:

I am working on building a study buddy tool. I want to use streamlit to make the front end. Give me based streamlit app front end, with 5 pages. The idea of the project is that there user can upload documents for courses. Then the other pages are the main pages where user can either select the uploaded document or pre-loaded course material (two dropdown course selection, then lecture selection within that course (can be multiple). All the other pages should have this option. Home, Upload documents, Generate summary, Practice questions, Notes and statistics. For practice questions had additional dropdown for type of questions to generate (mcq, true/false etc). Statistics make a simple eval page. Will later make changes to it.

ChatGPT said:

Here is a **clean, ready-to-run Streamlit multi-page app skeleton** for your Study Buddy tool.

It includes **5 pages**, shared dropdown selectors (course → lecture → uploaded docs), and the extra dropdown for **Practice Questions**.

You can directly copy this structure into your project.

## ✓ Directory Structure

```
study_buddy_app/
  -- streamlit_app.py          # Main entry point (Home)
  -- pages/
    |   1_Upload_Documents.py
    |   2_Generate_Summary.py
    |   3_Practice_Questions.py
    |   4_Notes_and_Statistics.py
  -- utils/
    |   shared_state.py
    |   ui_components.py
```

```
└── data/
    ├── uploaded/
    └── courses/
```

---

## ✓ utils/shared\_state.py

```
import streamlit as st
def init_state():
    if "uploaded_docs" not in st.session_state:
        st.session_state.uploaded_docs = {}
    if "selected_course" not in st.session_state:
        st.session_state.selected_course = None
    if "selected_lecture" not in st.session_state:
        st.session_state.selected_lecture = None
```

---

## ✓ utils/ui\_components.py

Reusable dropdown components.

```
import streamlit as st
def course_and_lecture_selector():
    st.sidebar.header("📚 Course Material") # Example preloaded courses
    courses = {
        "MLIP": ["Lecture 1", "Lecture 2", "Lecture 3"],
        "Intro to AI": ["Agents", "Search", "Expectimax", "Quality Assurance"],
        "Monitoring": ["Testability", "Coverage"]
    }
    selected_course = st.sidebar.selectbox("Select Course", ["None"] + list(courses.keys()), key="selected_course")
    if selected_course != "None":
        selected_lecture = st.sidebar.multiselect("Select Lecture(s)", courses[selected_course], key="selected_lecture")
    # Uploaded documents section
    st.sidebar.header("📄 Uploaded Documents")
    uploaded_list = list(st.session_state.uploaded_docs.keys())
    selected_doc = st.sidebar.selectbox("Select Uploaded Document", ["None"] + uploaded_list)
    return selected_course, st.session_state.get("selected_lecture", []), selected_doc
```

---

## ✓ streamlit\_app.py (Home Page)

```
import streamlit as st from utils.shared_state import init_state from
utils.ui_components import course_and_lecture_selector
st.set_page_config(page_title="Study Buddy", layout="wide") init_state()
st.title("🤖 Study Buddy – AI Learning Assistant") st.write(""" Welcome to the
Study Buddy tool! Upload your course materials or use pre-loaded content to
generate summaries, notes, practice questions, and evaluate your
understanding. """) # Sidebar shared options course_and_lecture_selector()
st.subheader("Project Overview") st.write(""" This tool helps students: –
Upload course material – Generate summaries – Create practice questions –
Take notes and view statistics """)
```

---

## ✓ pages/1\_ 📄 \_Upload\_Documents.py

```
import streamlit as st from utils.shared_state import init_state from
utils.ui_components import course_and_lecture_selector init_state()
st.title("📄 Upload Documents") course_and_lecture_selector() uploaded =
st.file_uploader("Upload a PDF / TXT / DOCX file", type=["pdf", "txt",
"docx"]) if uploaded: st.session_state.uploaded_docs[uploaded.name] =
uploaded st.success(f"Uploaded {uploaded.name}")
```

---

## ✓ pages/2\_ 🖊 \_Generate\_Summary.py

```
import streamlit as st from utils.shared_state import init_state from
utils.ui_components import course_and_lecture_selector init_state()
st.title("📝 Generate Summary") selected_course, lectures, selected_doc =
course_and_lecture_selector() st.write("### Select your source:")
st.write(f"- **Course:** {selected_course}") st.write(f"- **Lecture(s):** {lectures}")
st.write(f"- **Uploaded Doc:** {selected_doc}") if
st.button("Generate Summary"): st.info("Summary generation logic will be
added here.")
```

---

## ✓ pages/3\_ ? \_Practice\_Questions.py

```
import streamlit as st from utils.shared_state import init_state from
utils.ui_components import course_and_lecture_selector init_state()
```

```
st.title("❓ Practice Questions") selected_course, lectures, selected_doc = course_and_lecture_selector() q_type = st.selectbox( "Select Question Type", ["Multiple Choice", "True/False", "Short Answer", "Fill in the Blanks"] ) if st.button("Generate Questions"): st.info(f"Generating {q_type} questions...")
```

---

## ✓ pages/4\_📚\_Notes\_and\_Statistics.py

```
import streamlit as st from utils.shared_state import init_state from utils.ui_components import course_and_lecture_selector init_state() st.title("📝 Notes & Statistics") selected_course, lectures, selected_doc = course_and_lecture_selector() st.subheader("📝 Notes") st.text_area("Write your notes here...", height=200) st.subheader("📊 Statistics (Placeholder)") st.write("Evaluation metrics and usage stats will appear here.")
```

---

## 🎉 You're Set!

This gives you:

- ✓ Multi-page Streamlit app
- ✓ Shared sidebar across all pages
- ✓ Upload + choose documents
- ✓ Preloaded course/lecture dropdowns
- ✓ Summary, Questions, Notes, Stats pages
- ✓ Extendable architecture

If you want, I can next help you with:

- ✓ Adding embeddings + retrieval
- ✓ Adding MongoDB / FAISS support

- ✓ Improving UI with animations + themes
- ✓ Adding chatbot interface
- ✓ Connecting to your real course files

Just tell me!

**You said:**

i solely want to stick to front-end

**ChatGPT said:**

Got it — **pure front-end only**, no backend logic, no embeddings, no processing.  
Just a **clean Streamlit UI skeleton** with **5 pages**, shared dropdowns, and placeholder components.  
Here is a **very simple, lightweight version** you can directly use as your *frontend-only* starting point.

---

## ✓ Minimal Front-End Streamlit App (UI Only)

Everything works visually. No processing happens.

---

### Folder Structure

```
study_buddy/
├── Home.py
├── pages/
│   ├── 1_Upload_Documents.py
│   ├── 2_Generate_Summary.py
│   ├── 3_Practice_Questions.py
│   └── 4_Notes_and_Statistics.py
└── ui_utils.py
```

---

## ui\_utils.py

Shared UI components (just dropdowns + styling).

```
import streamlit as st def sidebar_selectors(): st.sidebar.header("📚 Course Material") courses = { "MLIP": ["Lecture 1", "Lecture 2", "Lecture 3"], "Intro to AI": ["Agents", "Search", "Expectimax"], "Quality Assurance": ["Testability", "Coverage", "Monitoring"] } selected_course = st.sidebar.selectbox( "Select Course", ["None"] + list(courses.keys()), ) selected_lectures = [] if selected_course != "None": selected_lectures = st.sidebar.multiselect( "Select Lecture(s)", courses[selected_course], ) st.sidebar.header("📄 Uploaded Documents") selected_doc = st.sidebar.selectbox( "Select Uploaded Document", ["None", "Document 1", "Document 2"] # Placeholder list ) return selected_course, selected_lectures, selected_doc
```

---

## Home.py

```
import streamlit as st from ui_utils import sidebar_selectors st.set_page_config(page_title="Study Buddy", layout="wide") st.title("🤖 Study Buddy – Your Learning Companion") sidebar_selectors() st.write(""" Welcome to Study Buddy! Upload your course documents, generate summaries, practice questions, and track your learning progress – all in one place. """)
```

---

## pages/1\_Upload\_Documents.py

```
import streamlit as st from ui_utils import sidebar_selectors st.title("📄 Upload Documents") sidebar_selectors() uploaded = st.file_uploader( "Upload a PDF / TXT / DOCX file", type=["pdf", "txt", "docx"] ) if uploaded: st.success(f"Uploaded: {uploaded.name}")
```

---

## pages/2\_Generate\_Summary.py

```
import streamlit as st from ui_utils import sidebar_selectors st.title("📝\nGenerate Summary") selected_course, selected_lectures, selected_doc =\nsidebar_selectors() st.write("### Selected Source") st.write(f"• Course:\n**{selected_course}**") st.write(f"• Lectures: **{selected_lectures}**")\nst.write(f"• Document: **{selected_doc}**") st.button("Generate Summary (UI\nOnly)") st.info("Summary will appear here...")
```

---

## ❓ pages/3\_Practice\_Questions.py

```
import streamlit as st from ui_utils import sidebar_selectors st.title("❓\nPractice Questions") selected_course, selected_lectures, selected_doc =\nsidebar_selectors() q_type = st.selectbox( "Select Question Type", ["Multiple\nChoice", "True/False", "Short Answer", "Fill in the Blanks"] )\nst.button("Generate Questions (UI Only)") st.info("Questions will appear\nhere...")
```

---



## pages/4\_Notes\_and\_Statistics.py

```
import streamlit as st from ui_utils import sidebar_selectors st.title("📚\nNotes & Statistics") sidebar_selectors() st.subheader("📝 Notes")\nst.text_area("Write notes here...", height=200) st.subheader("📊 Statistics\n(Placeholder)") st.write("Your stats and learning reports will appear here.")
```

---



## Done — Pure Front-End MVP

You now have:

- ✓ Clean multi-page UI in Streamlit
- ✓ Shared sidebar across all pages
- ✓ Upload UI
- ✓ Summary UI

- ✓ Practice Questions UI with type dropdown
  - ✓ Notes + Stats UI
  - ✓ Fully front-end only — no backend logic
- 

If you want, I can add:

- ✨ Custom theme (colors, fonts)
- ✨ Navigation bar on top
- ✨ Better card-style layout for results
- ✨ Icons, spacing, padding improvements
- ✨ Placeholder demo text so slides look good

Want the upgraded sleek version?

**You said:**

I don't want separate paeges. Compile it into one streamlit\_app.py

**ChatGPT said:**

Perfect — here is a **single-file Streamlit front-end** (streamlit\_app.py) with:

- ✓ 5 pages using a **sidebar page selector**
- ✓ Shared dropdowns for **course** → **lecture** → **document**
- ✓ UI-only (no backend)
- ✓ Clean, minimal, easy to modify
- ✓ No extra folders required

You can run this directly with:

```
streamlit run streamlit_app.py
```

---

# ★ Final Single-File Streamlit Front-End

## (streamlit\_app.py)

```
import streamlit as st # -----
----- # PAGE SETUP #
----- st.set_page_config(page_title="Study Buddy", layout="wide") # -----
----- # SIDEBAR NAVIGATION # -----
----- st.sidebar.title("📚 Study Buddy") page = st.sidebar.radio( "Navigation", ["Home", "Upload Documents", "Generate Summary", "Practice Questions", "Notes & Statistics"] ) # -----
----- # SHARED
SIDEBAR SELECTORS (for all pages except Upload) # -----
----- def shared_selectors():
st.sidebar.markdown("---") st.sidebar.subheader("📌 Select Source") #
Preloaded course list courses = { "MLIP": ["Lecture 1", "Lecture 2", "Lecture 3"], "Intro to AI": ["Agents", "Search", "Expectimax"], "Quality Assurance": ["Testability", "Coverage", "Monitoring"] } selected_course =
st.sidebar.selectbox( "Course", ["None"] + list(courses.keys()), key="course" ) selected_lectures = [] if selected_course != "None": selected_lectures =
st.sidebar.multiselect( "Lecture(s)", courses[selected_course], key="lectures" ) st.sidebar.subheader("📄 Uploaded Documents") selected_doc =
st.sidebar.selectbox( "Document", ["None", "Document 1", "Document 2"], # Placeholder names key="doc" ) return selected_course, selected_lectures,
selected_doc # -----
----- # HOME PAGE #
----- if
page == "Home": shared_selectors() st.title("🤖 Study Buddy – Your Learning Companion") st.write(""" Welcome to the Study Buddy tool! This frontend prototype lets you explore how students can interact with uploaded course material or pre-loaded lectures across different learning modes. """) st.markdown("### What you can do:") st.markdown(" - 📁 Upload your course documents - 🖊 Generate clean and structured summaries - 🤔 Create different types of practice questions - 📄 Take notes and view simple statistics ") # -----
----- # UPLOAD
DOCUMENTS PAGE #
----- elif page == "Upload Documents": # Upload page does NOT require course/lecture selector st.title("📄 Upload Documents") uploaded =
st.file_uploader( "Upload a PDF / TXT / DOCX file", type=["pdf", "txt",
"docx"] ) if uploaded: st.success(f"Uploaded: {uploaded.name}") st.info("This is frontend only – backend storage will be added later.") # -----
----- # GENERATE SUMMARY PAGE #
----- elif page == "Generate Summary": selected_course, selected_lectures, selected_doc =
shared_selectors() st.title("📝 Generate Summary") st.write("### Selected Source") st.write(f"- **Course:** {selected_course}") st.write(f"- **Lectures:** {selected_lectures}") st.write(f"- **Document:** {selected_doc}") st.button("Generate Summary (UI Only)") st.info("The summary will appear here...") # -----
----- # PRACTICE QUESTIONS PAGE #
----- elif page == "Practice Questions": selected_course,
selected_lectures, selected_doc = shared_selectors() st.title("❓ Practice Questions") q_type = st.selectbox( "Select Question Type", ["Multiple Choice", "True/False", "Short Answer", "Fill in the Blanks"] )
st.button("Generate Questions (UI Only)") st.info(f"{q_type} questions will
```

```
appear here...") # -----  
# NOTES & STATISTICS PAGE #  
----- elif page == "Notes & Statistics": selected_course,  
selected_lectures, selected_doc = shared_selectors() st.title("📝 Notes &  
Statistics") st.subheader("📝 Notes") st.text_area("Write your notes here...",  
height=200) st.subheader("📊 Statistics (Placeholder)") st.write("Simple  
evaluation stats and progress reports will appear here.")
```

---



## Done!

This is a **complete single-file Streamlit UI** that includes:

- ✓ **Navigation (Home, Upload, Summary, Questions, Notes/Stats)**
  - ✓ **Shared dropdowns across pages**
  - ✓ **Interactions + buttons (UI only)**
  - ✓ **Very clean, readable structure**
  - ✓ **No backend logic needed**
- 

If you'd like, I can next help you:

- ✨ Add a clean theme (colors, padding, fonts)
- ✨ Add card layouts (like modern app UI)
- ✨ Add example generated outputs so it looks great for demos
- ✨ Add animation / icons / banners

Just tell me!

**You said:**

I want to add a logo on the left on nav bar, how do I do that?

**ChatGPT said:**

Streamlit's default **sidebar radio navigation** doesn't support adding a logo *inside the radio menu* —  
but you *can absolutely* put a logo **above the navigation** in the sidebar and make it look like a real nav bar with branding.

Here are **two clean ways** to add a logo on the left of the nav bar.

---

## ✓ Option 1 — Simple: Add Logo Above the Sidebar Navigation

Just place an image before the radio buttons.

**Add this at the top of your sidebar section:**

```
from PIL import Image logo = Image.open("assets/logo.png") # your logo file  
st.sidebar.image(logo, width=120) # adjust size
```

**Example in your existing sidebar code:**

```
st.sidebar.title(" ") logo = Image.open("assets/logo.png")  
st.sidebar.image(logo, width=120) page = st.sidebar.radio( "", ["Home",  
"Upload Documents", "Generate Summary", "Practice Questions", "Notes &  
Statistics"] )
```

This gives you:

```
[LOGO]  
Home  
Upload Documents  
Generate Summary  
...
```

## 🚀 Option 2 — Stylish Custom Header (Logo + Title on Same Line)

Streamlit doesn't support inline layout in the sidebar natively, but you can inject HTML/CSS safely.

### Put this before the `radio()`:

```
st.sidebar.markdown( """ <div style="display: flex; align-items: center;">

<h3 style="margin: 0;">Study Buddy</h3> </div> """ , unsafe_allow_html=True )
```

If your logo is local, convert to base64:

```
import base64 def load_logo(path): with open(path, "rb") as f: data = f.read() return base64.b64encode(data).decode() logo_base64 =
load_logo("assets/logo.png") st.sidebar.markdown( f""" <div style="display: flex; align-items: center;">  <h3 style="margin: 0;">Study Buddy</h3> </div> """ , unsafe_allow_html=True )
```

★ This gives a clean branded nav bar like:

[LOGO] Study Buddy

Home  
Upload Documents  
...



### Want a top navigation bar instead?

I can also convert your UI to a **horizontal nav bar at the top** with the logo on the left — very modern, app-like.

---

**Want me to integrate the logo code *directly into your full app file* so it's plug-and-play?**

You said:

add css as well for the front end and give the entire code.

ChatGPT said:

Perfect — here is your **complete single-file Streamlit front-end** with:

- Custom CSS**
- Sidebar logo + styled nav bar**
- 5-page navigation (Home, Upload, Summary, Questions, Notes & Stats)**
- Shared course/lecture/document selectors**
- Fully front-end only**
- Clean and modern UI look**

You can drop this into **streamlit\_app.py** and run immediately.

---

## ★ FINAL COMPLETE CODE — `streamlit_app.py`

 Place your logo at: `assets/logo.png`

Or replace with any image path.

```
import streamlit as st from PIL import Image import base64 # -----
# PAGE SETUP #

st.set_page_config(page_title="Study Buddy", layout="wide") # -----
# LOAD LOGO #
def
load_logo_base64(path="assets/logo.png"): try: with open(path, "rb") as f:
data = f.read() return base64.b64encode(data).decode() except: return None
logo_base64 = load_logo_base64() # -----
# CUSTOM CSS #
st.markdown( """<style> /* Make sidebar wider */ [data-
testid="stSidebar"] { background-color: #f7f7f7; padding-top: 20px; width:
260px !important; } /* Sidebar title spacing fix */ .sidebar .sidebar-content
{ padding-top: 0 !important; } /* Navigation radio buttons styling */
.stRadio > label { font-size: 16px !important; font-weight: 600 !important;
color: #333 !important; } /* Style for section headers */ .sidebar-subheader
{ font-size: 14px; font-weight: 700; color: #444; margin-top: 12px; } /* 
Notes box styling */ textarea { border-radius: 10px !important; } </style>
""", unsafe_allow_html=True, ) # -----
```

```

----- # SIDEBAR HEADER WITH LOGO + TITLE # -----
st.sidebar.markdown("<br>",
unsafe_allow_html=True) if logo_base64: st.sidebar.markdown( f"""\n<div\nstyle="display:flex; align-items:center; margin-bottom: 20px;"> <img\nsrc="data:image/png;base64,{logo_base64}" width="45" style="margin-\nright:12px; border-radius:5px;"> <h2 style="margin:0; font-size:22px;">Study\nBuddy</h2> """, unsafe_allow_html=True ) else: st.sidebar.write("Study\nBuddy") #
----- # SIDEBAR NAVIGATION #
page = st.sidebar.radio( "", ["Home", "Upload Documents", "Generate\nSummary", "Practice Questions", "Notes & Statistics"] ) #
----- # SHARED SELECTORS FOR ALL PAGES
(except Upload) #
def shared_selectors(): st.sidebar.markdown("<p class='sidebar-\nsubheader'>Course Material</p>", unsafe_allow_html=True) courses = { "MLIP":\n["Lecture 1", "Lecture 2", "Lecture 3"], "Intro to AI": ["Agents", "Search",\n"Expectimax"], "Quality Assurance": ["Testability", "Coverage", "Monitoring"]\n} selected_course = st.sidebar.selectbox("Course", ["None"] +\nlist(courses.keys())) selected_lectures = [] if selected_course != "None":\nselected_lectures = st.sidebar.multiselect("Lecture(s)",\ncourses[selected_course]) st.sidebar.markdown("<p class='sidebar-\nsubheader'>Uploaded Documents</p>", unsafe_allow_html=True) selected_doc =\nst.sidebar.selectbox("Document", ["None", "Document 1", "Document 2"]) return\nselected_course, selected_lectures, selected_doc #
----- # HOME PAGE #
if page == "Home": shared_selectors()
st.title("🤖 Study Buddy – Your Learning Companion") st.write(""" Welcome to\n**Study Buddy**! This frontend prototype demonstrates how a student interacts\nwith course material. """) st.markdown("## What you can do:")
st.markdown(" - 📄 Upload your course documents - 📝 Generate summaries - ?\nCreate different types of practice questions - 📚 Take notes and view\nstatistics ")
# UPLOAD DOCUMENTS PAGE #
elif page == "Upload Documents": st.title("📄 Upload Documents")
uploaded_file = st.file_uploader("Upload a PDF / TXT / DOCX", type=["pdf",\n"txt", "docx"]) if uploaded_file: st.success(f"Uploaded:\n{uploaded_file.name}") st.info("This UI is frontend-only. Backend storage\nwill be added later.") #
----- # GENERATE SUMMARY PAGE #
elif page == "Generate Summary": selected_course,\nselected_lectures, selected_doc = shared_selectors() st.title("📝 Generate\nSummary") st.write("## Selected Source") st.write(f"- **Course:**\n{selected_course}") st.write(f"- **Lectures:** {selected_lectures}")\nst.write(f"- **Document:** {selected_doc}") st.button("Generate Summary (UI\nOnly)") st.info("Summary results will appear here...") #
----- # PRACTICE QUESTIONS PAGE #
elif page == "Practice\nQuestions": selected_course, selected_lectures, selected_doc =\nshared_selectors() st.title("❓ Practice Questions") q_type = st.selectbox(\n"Select Question Type", ["Multiple Choice", "True/False", "Short Answer",\n"Fill in the Blanks"] ) st.button("Generate Questions (UI Only)")\nst.info(f"{q_type} questions will appear here...") #
----- # NOTES & STATISTICS PAGE #
elif page == "Notes &

```

```
Statistics": selected_course, selected_lectures, selected_doc =  
shared_selectors() st.title("📝 Notes & Statistics") st.subheader("📝 Notes")  
st.text_area("Write your notes here...", height=220) st.subheader("📊 Statistics (Placeholder)") st.write("Evaluation metrics and progress tracking  
will appear here.")
```

---



## Your frontend is now fully styled

This version includes:

- ✓ Custom sidebar layout
  - ✓ Logo + title header styling
  - ✓ CSS for padding, typography, radiobutton styling
  - ✓ Beautiful, consistent UI
  - ✓ Easy to extend
- 

If you want, I can also:

- ✨ Add **light/dark themes**
- ✨ Add **card-style layouts** for outputs
- ✨ Add **button animations**
- ✨ Make the app look like a polished **SaaS dashboard**

Just tell me!

ChatGPT can make mistakes. Check important info.