

Um es von der Logik richtig einzuordnen ( so Abschluss-Gedanke): Eine "normale" Web-App läuft im Browser und besitzt ein NodeJS-Backend, welches Daten speichert, SQL.-Abfragen ausführt und statische Inhalte ausliefert. Durch den integrierten webViewer werden die Inhalte vom Browser in die Anwendung verlagert, wodurch sich die Möglichkeit ergibt, in die Anwendung einzugreifen ...

Genau, Thomas – Dein Gedankengang ist absolut stimmig 👍. Lass uns das sauber auseinanderziehen, damit die Logik klar bleibt:

#### **Klassische Web-App**

- **Frontend:** Läuft im Browser, meist HTML/JS/CSS.
- **Backend (z. B. Node.js):**
  - Liefert statische Inhalte (HTML, JS, Bilder).
  - Führt Logik aus (REST-API, SQL-Abfragen, Business-Logik).
  - Speichert Daten in einer Datenbank.
- **Kommunikation:** Browser ↔ Backend über HTTP/HTTPS.
- Ergebnis: Der Benutzer interagiert mit einer Website, die Daten vom Server holt und darstellt.

#### **Office Add-in mit WebView2**

- **Frontend:** Dein Taskpane ist technisch ebenfalls eine Web-App (HTML/JS).
- **WebView2:**
  - Eingebetteter Browser innerhalb von Word/Excel/Outlook.
  - Lädt Deine Web-App direkt in die Office-Oberfläche.
- **Integration:**
  - Über Office.js kannst Du direkt ins Dokument eingreifen (Text einfügen, Inhalte lesen, speichern).
  - Das unterscheidet es von einer „normalen“ Web-App: Du hast Zugriff auf die **Office-Objektmodelle**.
- **Backend (Node.js):**
  - Läuft wie gewohnt, liefert Inhalte und verarbeitet Daten.
  - Für Office ist es egal, ob das Backend lokal oder in der Cloud steht – Hauptsache, es ist über HTTPS erreichbar.

#### **Logische Einordnung**

- Eine Office-Add-in-Taskpane ist **eine Web-App im WebView2**, die zusätzlich über Office.js Zugriff auf die Host-Anwendung (Word, Excel, Outlook) bekommt.
- Damit verschiebt sich die Grenze:
  - **Normale Web-App** → nur Browser + Server.
  - **Office Add-in** → Browser (WebView2) + Server + direkte Schnittstelle ins Dokument.

#### **Dein Abschluss-Gedanke**

Du hast es genau getroffen:

- Der WebViewer (WebView2) „verlagert“ die Web-App in die Anwendung.
- Dadurch entsteht die Möglichkeit, **in die Anwendung selbst einzugreifen** – Texte schreiben, Inhalte auslesen, Dokumente speichern.
- Das ist der eigentliche Mehrwert von Office Add-ins: Web-Technologien mit nativer Office-Integration.