**להגיש בקישור:** <https://drive.google.com/drive/shared-with-me>

**שלב ראשון של הפתרון לעזרה :** <https://drive.google.com/drive/folders/1w1wTk7eUSyUwIFltfgZYaEEWVWzQK9I8>

הrepo של הפתרון של המיני פרויקט**:**

[**https://github.com/JonathanZouari/Project\_Airbnb\_Sol?tab=readme-ov-file#notebooks**](https://github.com/JonathanZouari/Project_Airbnb_Sol?tab=readme-ov-file#notebooks)

**הסברים ונקודות לעצמי:**

**תהליך הפרויקט – עיקרי השלבים (לפי המסמך של התרגיל)**

1. **הגדרת מבנה הפרויקט** – יצירת תיקיות עבודה מסודרות.
2. **הקמת סביבת עבודה** – עדיף לעשות את ההקמה תחת ספרית הפרויקט. (לעבור לשם בטרמינל).
3. **ניקוי נתונים** – עיבוד קובץ ה־CSV: הורדת עמודות מיותרות, המרות, ניקוי ערכים חריגים וחסרים.
4. **הנדסת מאפיינים** (Feature Engineering) – המרת עמודות טקסט לקטגוריות, קידוד - לעמודות בינאריות (0/1) עבור כל ערך אפשרי.
5. **חלוקה ל־Train/Test** – הפרדת נתוני אימון ונתוני בדיקה.
6. **בניית מודל ML** – אימון מודל חיזוי מחיר, שמירתו, והערכתו. (אם איכות הנתונים לא מספיק טובה, אז צריך לשפר את המודל, לסנן אחרת או להיעזר במודלים נוספים, עד לקבלת תוצר איכותי).
7. **בניית ממשק Streamlit** – טופס קלט, תצוגה וגרפים.
8. **בוט טלגרם** – שליחת נכסים וחיזוי מחיר דרך בוט.
9. **שילוב LLM** – הסבר חכם של התוצאות.
10. **בדיקות קצה** – בדיקות לחריגות ושימושיות.
11. **תיעוד בקובץ קובץ requirements.txt** – עם פקודה בטרמינל
    * pip freeze > requirements.txt יעדכן אוטומטי.
12. **תיעוד** והעלאה ל־GitHub.
    * הפעלה מחדש של סביבת העבודה

.\venv\Scripts\activate

או להעתיק ביוק מה שבסוגרים ולהתאים את שמות המחיצות (.\venv\Scripts\activate)

**שלב ראשון: מבנה תיקיות**

* data לשמור שם את כל קבצי הנתונים, כולל cleaned\_data ו־featured\_data כולל קובץ המידע הראשוני בסיס הנתוניםlistings.csv
* scripts לכל שלבי העבודה בפייתון: ניקוי נתונים, הנדסת מאפיינים, אימון מודל וכו'.
* models שמירת המודל המאומן.
* utils פונקציות עזר (לא חובה, אבל מקובל).
* README.md תיעוד הפרויקט – יותר קל לאחרים ולי לזכור מה עשיתי ומה ההנחיות הבסיסיות להפעלת הפרויקט. רצוי לעדכן אותו בכל שלב משמעותי בפרויקט.
* requirements.txt רשימת חבילות להתקנה (נכין בהמשך).
* קובץ .gitignore
  + .gitignore הוא קובץ שמכניסים בתיקיית הפרויקט כאשר עובדים עם Git (מערכת גרסאות, הכי נפוץ בפרויקטים).
  + הקובץ אומר ל־Git איזה קבצים לא להעלות ל־GitHub או לכל מערכת גרסאות אחרת.
  + למה זה חשוב?: כדי לא להעלות בטעות מידע רגיש (כמו .env).. כדי לא להעלות קבצי מערכת, קבצים זמניים או תוצאות ריצה של קוד.

**דוגמה: רצף עבודה יומיומי**

1. פותח את Cursor ובוחר את תיקיית הפרויקט.
2. מוסיף קוד חדש/סקריפט חדש ב־scripts.
3. כל שינוי חשוב – שומר, מבצע commit.
4. בסיום – push ל־GitHub.
5. משתמש בהשלמה והסבר של Cursor ו־AI בכל שלב.
6. כותב שאלות או TODO ב־README (לעצמך או לשותף).
7. כל מידע רגיש — עובר ל־.env.

**איך רואים ומנהלים את ה־Git ישירות מתוך Cursor?**

1. **סרגל הצד (Side Bar) – Source Control**

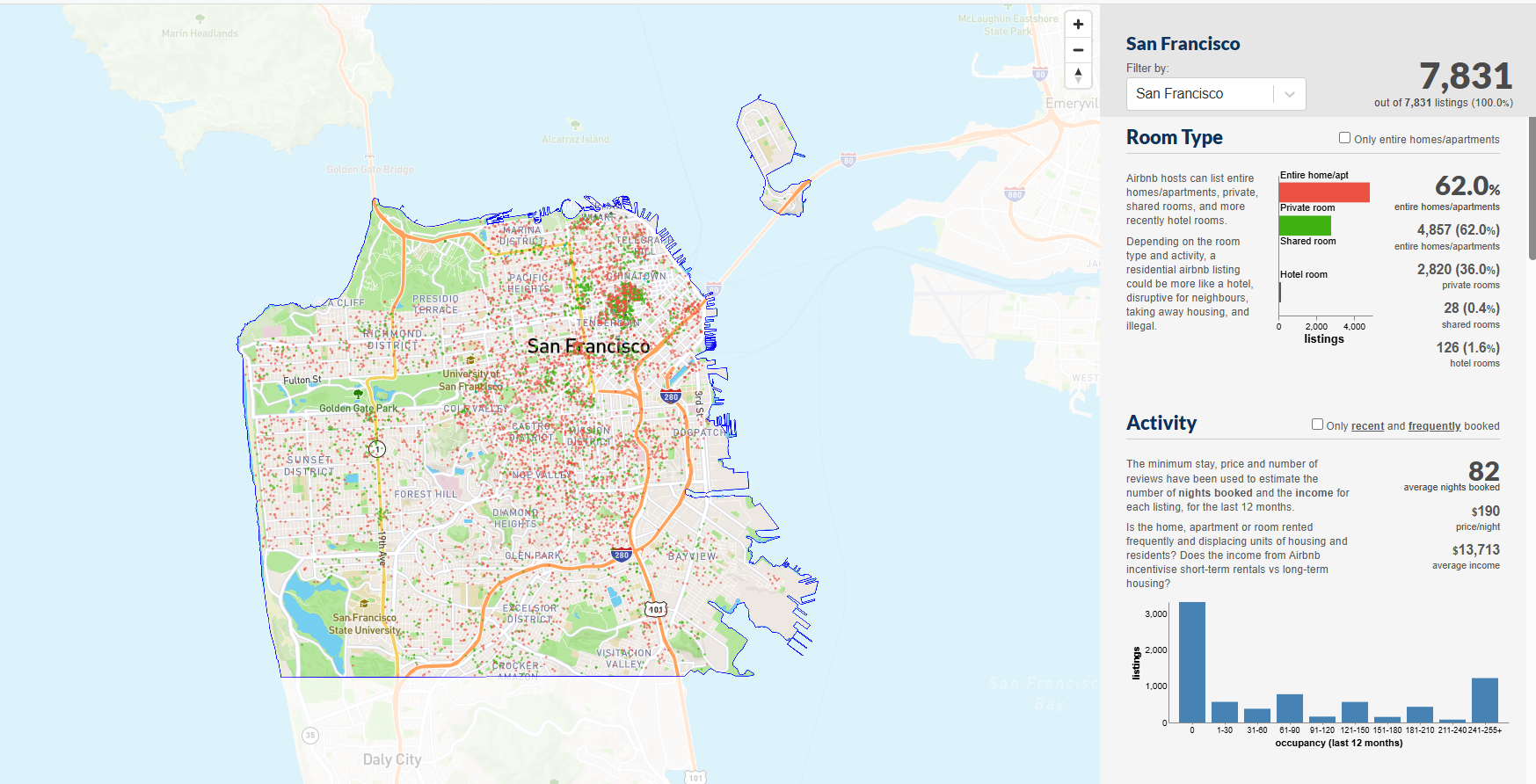
* ב־Cursor (כמו ב־VS Code) יש סרגל צד שמאלי, שבו מופיע אייקון של **ענף** (הוא נראה כמו שתי עיגולים עם קו ביניהם, או שלושה עיגולים מחוברים).
* תלחץ עליו – וזה עובר למצב Source Control.
* כאן תראה:
  + **שינויים שלא בוצע להם commit** ("Changes")
  + קבצים שנוספו/שונו
  + אפשרות להכניס הודעת commit, ולאשר (✓)
  + אפשרות ל־push / pull (אם מחובר ל־GitHub)

1. **פעולות עיקריות דרך הממשק**

* **Commit:**  
  כותבים הודעה בשדה, לוחצים ✓ – זה מבצע commit.
* **Stage / Unstage:**  
  אפשר להוסיף או להוריד קבצים מה־commit ע"י לחיצה ימנית.
* **Push:**  
  לוחצים על שלוש נקודות (⋮) למעלה או על החץ (⬆) ודוחפים את ה־commit ל־GitHub.
* **Pull:**  
  מושכים עדכונים מהשרת (⬇).
* **היסטוריה (History):**  
  לחיצה ימנית על קובץ > Show History – לראות כל הגרסאות הקודמות שלו.
* **הבדלים (Diff):**  
  לחיצה על קובץ ברשימת השינויים – רואים את ההבדלים (צבעוני, אדום ירוק).

**🏗️ מיני פרויקט: יועץ נדל"ן חכם בסן פרנסיסקו**

**שילוב של: Machine Learning + Streamlit + Telegram Bot + LLM**



**🎯 מטרה:**

בנו מערכת שלמה שמקבלת פרטים על דירה להשכרה בסן פרנסיסקו ומבצעת:

* חיזוי מחיר השכרה ללילה.
* הסבר בשפה פשוטה למה המחיר הזה.
* הצגת המידע בדשבורד אינטראקטיבי (Streamlit)
* יצירת בוט טלגרם (או פלטפורמה אחרת) שמבצע את כל הפעולות דרך הודעות.

**📁 שלב 1: הגדרת מבנה הפרויקט**

**יצירת תיקיות מסודרות:**

* data/ שם יהיה קובץ הCSV-
* scripts/ כל שלבי העבודה בפייתון.
* models/ המודל המאומן.
* utils/ פונקציות עזר (לא חייב).
* README.md ו־requirements.txt לשימוש עתידי.

את המילון שמפענח את המשתנים ניתן למצוא כאן:

<https://docs.google.com/spreadsheets/d/1iWCNJcSutYqpULSQHlNyGInUvHg2BoUGoNRIGa6Szc4/edit?gid=1322284596#gid=1322284596>

**🧹 שלב 2: ניקוי נתונים (Data Cleaning)**

**מטרות:**

* הסרת עמודות לא רלוונטיות.
* טיפול בערכים חסרים (NaN)
* המרה של עמודת מחיר מפורמט טקסט למספר.
* סינון קיצונים לא הגיוניים במחיר.

**משימות לביצוע:**

1. לטעון את קובץ הנתונים המקורי.
2. לבחור רק את העמודות החשובות למודל: מיקום, סוג חדר, מספר חדרים, אמבטיות, מחיר וכו'.
3. לבדוק האם בעמודת המחיר יש סימני $ ולהסיר אותם.
4. להמיר את המחיר לערך מספרי.
5. להסיר שורות עם ערכים חסרים בעמודות קריטיות.
6. לסנן שורות עם מחירים חריגים (למשל מעל 1000$ או מתחת ל־40$).
7. למלא ערכים חסרים בעמודות משניות עם ממוצע או חציון.
8. לשמור את הקובץ כ־cleaned\_data.csv

**🧠 שלב 3: הנדסת מאפיינים (Feature Engineering)**

**מטרות:**

* להפוך עמודות טקסט לייצוג מספרי.
* להכין את הנתונים לקראת מודל ML

**משימות לביצוע:**

1. לבדוק אילו עמודות הן קטגוריאליות (כמו סוג החדר).
2. לבצע קידוד One-Hot לאותן עמודות.
3. לוודא שאין בעיות של ערכים חסרים לאחר ההמרה.
4. לשמור את הקובץ כ־featured\_data.csv

**📊 שלב 4: חלוקה לאימון ובדיקה**

**מטרות:**

* לחלק את הנתונים ל־ Train ו־Test
* להכין את סט הנתונים למודל.

**משימות לביצוע:**

1. להפריד בין משתני הקלט (features) למשתנה המטרה (price)
2. להשתמש בפונקציית train\_test\_split ולבחור למשל 80% לאימון, 20% לבדיקה.
3. לשמור את הסטים אם צריך להערכה מאוחרת.

**🤖 שלב 5: בניית מודל ML**

**מטרות:**

* לאמן מודל לחיזוי מחיר.
* לבחור אלגוריתם ולכוון פרמטרים בסיסיים.

**משימות לביצוע:**

1. לבחור אחד מהמודלים הבאים: CatBoost / XGBoost / LightGBM.
2. לאמן את המודל על סט האימון.
3. לשמור את המודל המאומן כ־.pkl (לא חובה, מומלץ לקרוא על זה)
4. למדוד ביצועים על סט הבדיקה (לא שלכוח לעשות Cross Validation):
   * MAE
   * R²
5. לרשום הסבר על תוצאות ההערכה: איפה הוא טועה, האם המחיר מנופח מדי?

**🌐 שלב 6: בניית ממשק Streamlit**

**מטרות:**

* לאפשר למשתמש להזין פרטי נכס.
* להציג את המחיר הצפוי.

**משימות לביצוע:**

1. ליצור טופס קלט:
   * בהתאם למאפיינים שבחרתם במודל.
2. להמיר את הקלט לפורמט שהמודל מבין.
3. להשתמש במודל ולחזות את המחיר.
4. להציג את המחיר בצורה ברורה.
5. להוסיף גרפים או מפה להצגת התפלגות מחירים.
6. לבדוק מה קורה אם מכניסים ערכים לא תקינים.

**💬 שלב 7: בוט טלגרם**

**מטרות:**

* לאפשר שליחה של פרטי נכס כטקסט.
* להחזיר חיזוי מחיר + הסבר.

**משימות לביצוע:**

1. ליצור בוט טלגרם דרך BotFather
2. לכתוב קוד שמאזין להודעות.
3. לפענח את פרטי ההודעה: כמות חדרים, סוג החדר וכו’.
4. לבצע חיזוי מחיר עם המודל.
5. לשלוח תשובה עם המחיר.
6. לשלב מודל LLM כמו OpenAI כדי להסביר למה המחיר כזה.

**🧠 שלב 8: שילוב LLM להסברים**

**מטרות:**

* להוסיף רובד של ניתוח בשפה טבעית.

**משימות לביצוע:**

1. לשלוח בקשה ל־LLM עם המחיר והקלט המקורי.
2. לבקש הסבר פשוט על המחיר ("למה דווקא ככה?").
3. להחזיר את הטקסט הזה בטלגרם או בדשבורד.
4. בנו דאטהבייס שמשמר את ה-prompt והתשובה ל-prompt המתקבל מהמודל.

**🧪 שלב 9: בדיקות קצה**

**מה לבדוק:**

* מה קורה אם יש קלט שגוי?
* איך נראית תשובה לדירה יקרה מאוד או זולה מאוד?
* האם אפשר להשתמש באותו מודל גם לדירות בלי אמבטיה?

**🧾 שלב 10: תיעוד והעלאה ל-Github**

**משימות:**

1. לכתוב README שמסביר:
   * מה הפרויקט עושה.
   * איך להריץ כל סקריפט.
   * דוגמאות לקלט ופלט.
2. לכתוב מסמך קצר: אילו בעיות נתקלתם בהן? מה למדתם?