דוח מעבדה- תרגיל סוף (scripting project)

|  |  |  |
| --- | --- | --- |
| ייבוא ספריות ומבנה הקובץ |  | עם תחילת העבודה בחרנו לייבא מספר ספריות, לאחר התקנתן, שישמשו אותנו במהלכה. חלק מהספריות שיובאו הן: pandas, מספר הרחבות של ספריית selenium, requests, time ו- googletrans.  את קובץ הפייטון סדרנו לפי החלקים הבאים: ספריות שייבאנו, פונקציות שבנינו ובהן עשינו שימוש והקוד הראשי שמחולק לפתיחת דפדפן הפרלמנט של מדינת פינלנד, יצירת הגדרות ומשתנים לצורך קובץ האקסל אליו יועבר כל המידע והפעלת הפונקציה הראשית של קצירת המידע ע"י הזחלן ואחרי כל עמוד שמירת המידע בקובץ אקסל. |
| תחילת הקוד ופונקצייתmain crawler |  | עם תחילת הקוד הראשי, בעזרת ספריית selenium הגדרנו כל מה שדרוש על מנת להנגיש את הדפדפן לזחלן שלנו.  לאחר מכן הגדרנו פתיחת קובץ אקסל חדש ומתוך הסוויטה שלו קראנו לפונקציה הראשית של הזחלן כך שכל המידע שלו יועבר באופן ישיר לדאטה פריים שלנו ולקובץ האקסל הסופי שיסכם את כל המידע שנקצור.  האופן בו הזחלן בקוד שלנו עובד הוא בעזרת פונקציה שקוראת לו להתחיל את העבודה על הדפדפן שהגדרנו לו (def MainCrawler). בתוכה ישנן לולאות כפולות שמפעילות פונקציות פנימיות עיקריות שלכל אחת מהן ישנה מטרה אחרת בקצירת המידע.  המטרה של הלולאות היא לקצור מכל עמוד ועמוד באתר את כל המידע על חברי הפרלמנט הנמצאים בו בכך שהוא יוצר בתחילת העבודה רשימה של כל חברי הפרלמנט הנמצאים באותו עמוד ספציפי ורץ על כולם. בסוף הפעילות בעמוד הוא יעבור לעבוד הבא ויחזור על פעולתו.  אופן שמירת המידע על כל חבר פרלמנט מתבצע בעזרת רשימה אליה כל המידע על אותו חבר פרלמנט יוכנס ובסוף אותה רשימה תוכנס כשורה (תצפית) לדאטה פריים שלנו ולקובץ האקסל. |
| פונקציית Main Info Retrieve |  | המטרה של פונקציה זו היא לקצור את כל המידע הראשוני הנגיש על חבר הפרלמנט ולהכניסו לרשימה.  המידע שפונקציה זו קוצרת הוא שם חבר הפרלמנט, המדינה לה משתייך והקישור לדף הראשי של חבר הפרלמנט. כל פרטי המידע הללו קיימים עבור כל חברי הפרלמנט ולכן הם לא מוגדרים בתוך try ו-except.  מידע נוסף שנקצר בפונקציה זו הוא מספר טלפון, כתובת אימייל ודף בית חיצוני ראשי של חבר הפרלמנט אך בשל העובדה שמידע זה לא קיים עבור כל חברי הפרלמנט הבקשה לקצירת מידע זה הוכנסה בקוד לתוך try ובמידה ומידע זה לא קיים, במסגרת except יוכנס לרשימה במקום "המידע לא זמין". |
| פונקציית CV info retrieve |  | המטרה של פונקציה זו היא לקצור את המידע הקשור לקורות החיים של חבר הפרלמנט ולהכניסו לרשימה.  פונקציה זו פועלת בשלושה מישורים (כולם תחת try ו-except):   1. היא מחפשת מידע הקשור לקורות חייו של חבר הפרלמנט באזורים שונים של הדפדפן. חלק מהמידע קיים בחלקו העליון של הדפדפן וחלקו קיים תחת לשונית CV הדורשת הקלקה לצורך חשיפת המידע. בין היתר, המידע שהפונקציה מנסה לקצור הוא מקצוע, שנת ומקום לידה, השכלה, תפקידים מוניציפליים. 2. במידה והפונקציה מצאה מידע על קורות החיים של חבר הפרלמנט, היא מתרגמת אותו לאנגלית (מפינית) בעזרת קריאה לפונקציית EnglishAPI.   פונקציה זו מתרגמת את המידע באמצעות ספריית googletrans ומוסיפה לרשימה את התרגום של אותו פרק מידע בצמוד אליו.   1. לאחר שהפונקציה סיימה לקצור את כל המידע שיכלה מקורות החיים של חבר הפרלמנט, היא מפרידה וסוכמת את מספר המילים שנקצרו בכל אחד מהחלקים השונים (תוך כדי תנועה) ומכניסה את הסכום הסופי כערך נוסף לרשימה.   במסגרת קצירת המידע בפונקציה זו, יש 2 משתנים שהטקסט שלהם מצוי בתוך רשימה ולכן קצירה מלאה שלו מתג האב יוביל להימצאות "ירידות שורה" בטקסט (\n). על מנת לסדר את הטקסט בתוך משתנים אלו, טרם העברתו לתרגום, הפרדתי בין המילים באמצעות הצירוף "\n" ולאחר מכן החזרתי אותם חזרה להיות מחרוזת כך שמה שיפריד בין כל שורת מקור אחד לשנייה יהיה "; ". |
| פונקציית image retrieve |  | המטרה של פונקציה זו היא לקצור את תמונתו של חבר הפרלמנט, במידה וקיימת, ולהכניסה לרשימה (בעזרת try ו-except).  במידה ואין תמונה אז במקומה יוכנס "המידע לא זמין" לרשימה. |
| פונקציית gender api |  | המטרה של פונקציה זו היא לבחון לפי שמו של חבר הפרלמנט האם הוא זכר/נקבה ומה ההסתברות לכך.  את שתי המשימות הללו פונקציה זו מבצעת באמצעות API שזוהי ממש מטרתו.  עבור כל חבר כנסת חיפשנו את שמו המלא ומשם לקחנו רק את שמו הפרטי והכנסו אותו כפרמטר לאותו API.  לאחר מכן, לקחנו את הפסיקה של ה-API מהמשתנה שלו “gender” והכנסנו לרשימה עם כל המידע על חבר הפרלמנט. בנוסף לכך, לקחנו משתנה נוסף מאותו ה-API “probability” שמטרתו לבחון מה ההסתברות של אותה פסיקה וגם אותו הכנסנו לרשימת המידע. |
| פונקציית parliament membership |  | המטרה של פונקציה זו היא לקצור את רשימת השנים בהן היה חבר פרלמנט פעיל.  ביצענו זאת ע"י מציאת המיקום בדפדפן בו מפורטות שנות הכהונה בפרלמנט ומשיכת כלל הטקסט הקיים שם.  באמצעות split הפכנו את הטקסט לרשימה שבאופן רוחבי מיוצגת אצל כלל חברי הפרלמנט עם טקסט באיבר הראשון ואז כלל התאריכים בהם כיהן בחבר פרלמנט. בעקבות כך, חיברנו מחדש את הרשימה באמצעות join עם הפרדת "רווח" רק מהאיבר השני (ללא הטקסט האלפביתי) ועד הסוף וכך יצרנו מחרוזת ובה רק התאריכים הרלוונטיים עבורנו.  הסיבה שבחרנו לעשות join עם הפרדת "רווח" היא על מנת לאפשר קצירת של כלל התאריכים גם עבור פרטים שכיהנו כחברי פרלמנט מספר פעמים בטווחי שנים שונים. |
| אתגרים |  | במסגרת העבודה על הפרויקט נתקלנו באתגרים הבאים:   1. כבר משלב פתיחת הדפדפן והנגשת כל המידע הראשוני שבו לזחלן, בחלק מהריצות הראשונות לעיתים הקוד קרס כיוון שלא הצליח לפתוח את הדפדפן. על מנת להימנע ממצב כזה הגדרנו try ו-except כבר בשלב זה על מנת למנוע קריסה של הקוד. בנוסף, על מנת לתת לדפדפן מספיק זמן לטעון את המידע הראשוני שלו לאחר טעינת העמוד הגדרנו במספר מקומות sleep על מנת שיצליח להיטען במלואו לפני שהזחלן מתחיל את ביצוע הפעולות בו בפועל.   באופן כללי, לאחר כל פעולת טעינת מידע/ גלילת דף/ חיפוש מידע הגדרנו sleep על מנת לצמצם את אי הצלחתו של הזחלן במשימתו.   1. אתגר נוסף שנתקלנו בו כבר מההתחלה הוא השמירה על פעילות תקינה של הזחלן גם אם לא יצליח למצוא את המידע שאנו מבקשות ממנו לחפש. על מנת לוודא זאת, יישמנו את מנגנון try/except על כל פעולת חיפוש פריט בדפדפן שהיה לנו ספק אם קיים בכל העמודים של כלל חברי הכנסת. כל פעולה כזו שלא תצליח לא תקריס את הקוד וגם תסייע לנו באמצעות סוג exception ספציפי להכניס לרשימה "מידע לא זמין" (סוג ה-exception איתו עבדנו רוב הפעמים במקרים אלו הוא "NoSuchElementException"). 2. אתגר נוסף שעיכב אותנו בפרויקט היה מציאת ספריית תרגום טובה עבור הזחלן שלנו. חלק מהספריות וה-APIים שמצאנו בהתחלה היו מוגבלים במספר המילים שאנו יכולות להכניס לתרגום במסגרת הקוד ובשל העובדה שיש לנו באתר הפרלמנט יותר מ-2,600 חברים ידענו כי זה עלול להוות בעיה קשה עבורנו. אתגר נוסף בהקשר זה היה שלא מעט ספריות תרגום הינן בעלות מספר גרסאות שחלקן לא עובדות בכל גרסת פייטון וחלקן כבר לא רלוונטיות באופן כללי. אופן ההתמודדות שלנו עם שני אתגרים אלו היה פשוט לחפש ספרייה אחר ספרייה עד שמצאנו את זו שעובדת טוב ומתרגמת באופן יחסית טוב את המידע מהאתר ואכן הספרייה שמצאנו לנכון לעבוד איתה היא googletrans. 3. בהמשך לאתגר מס' 3 הקשור למציאת ספריית תרגום טובה עבור הקוד שלנו, לאורך הרצת הקוד מספר פעמים גילינו שגם ספריית googletrans בעלת מגבלות משלה. לאחר מספר איטרציות הקוד קרס והועלה לנו ה-exception הבא: " the read operation timed out" ומבדיקה אינטרנטית אודות exception זה עלה כי הוא מדבר על אי הספק של הספק (google) לתת מענה לשאילתת התרגום שהעברנו לו. מבחינתו של הנושא עלו מספר בעיות אפשריות סביב זה:    1. שלספק אין מספיק זמן לתרגם לנו את הטקסט בין שאילתה לשאילתה ולכן הקוד קורס.    2. שלספק יש מנגנון שאם הוא מזהה X שאילתות מאותו IP במסגרת זמן מסוימת הוא חוסם בקשות נוספות ממנו במסגרת אותה ריצה.   על מנת לנסות לפתור בעיה זו תחילה ניסינו לרווח את בקשות התרגום ששלחנו לספק באמצעות הכנסת sleep בפנים ואכן, זה הוביל לכך שמספר הבקשות תרגום עלה והקוד הצליח להתקדם לעוד חברי פרלמנט באיטרציות הבאות אך חזר וקרס בהמשך מאותה סיבה. ניסיון נוסף שלנו לפתור בעיה זו היה לאסוף ברשימה את כלל הטקסטים שאנו צריכות לשלוח לתרגום ולשלוח אותם יחד כמקשה אחת ולאחר קבלת התרגום להחזיר את הטקסט למצבו המתבקש והמתאים עבור המסד שאנו בונות. לצערנו גם ניסיון זה נתקל בבעיות מסיבה דומה, כאשר שלחנו רשימה ארוכה עם המון קובץ לתרגום, הפעולה לא צלחה.  דרך נוספת עימה בחרנו לנסות ולהאריך את זמן המענה לבקשת התרגום הוא הכנסת לולאת for לקוד כך שגם אם השרת של גוגל לא הצליח לתרגם בבקשה הראשונה, אנו חוזרות "להציק" לו עם בקשות חוזרות עד שיתפנה לתת לנו מענה. פעולה זו הובילה להעלאה משמעותית במספר חברי הפרלמנט שהזחלן שלנו מושך עבורם מידע אז עדיין זה לא פתר את הבעיה לחלוטין.  נקודה נוספת שגילינו על בעיה זו היא התזמון בו אנו שולחות את בקשת התרגום לשרת- אם אנחנו מריצות את הקוד באמצע שבוע במהלך שעות היום השרת עמוס הרבה יותר לעומת אם אנו מריצות את הקוד במהלך ימי החופש של ארה"ב (שבת-ראשון), אז הזחלן מצליח להפיק הרבה יותר מידע על יותר חברי פרלמנט.  לאחר ניסיונות רבים עצמאיים ומול חברינו לתואר התייעצנו בסוגייה זו עם המרצה אשר יעץ לנו לשנות את הלולאה בקוד ללולאת while. ניסיון זה לצערנו גם לא צלח לאחר שהמחשבים שלנו לא הצליחו להתמודד עם העבודה בתוך הלולאה וקרסו. נוסף על כך, ראינו כי במסגרת לולאה זו, הזחלן נתקע על בקשת תרגום אחת ופשוט לא הצליח להתקדם ממנה הלאה.   1. בהמשך לאתגר התרגום ובשל כמות חברי הפרלמנט הגדולה שקיימת באתרינו, חרגנו את מגבלת הבקשות שניתן להעביר ל-API של הניתוח המגדרי. על מנת להילחם במגבלה זו שלחנו מייל ליוצרי ה-API בו הסברנו מי אנחנו ומה מטלתנו וביקשנו אם ניתן יהיה לקבל מפתח אישי שלא יגביל את הבקשות שנוכל לשלוח. נענינו בחיוב וקיבלנו מפתח אשר יאפשר לנו להגיש 10 מיליון בקשות פר חודש ללא בעיות. 2. אתגר נוסף עימו התמודדנו צץ רק לאחר האיטרציה הראשונה של הזחלן (לאחר קצירת כל המידע אודות חבר הפרלמנט הראשון).   לאחר קצירת כל המידע אודות חבר הפרלמנט הראשון, הזחלן מתבקש לחזור דף אחד אחורנית לדף בו קיימת רשימת כלל חברי הפרלמנט ממנו נכנסים לעמודים הפרטניים של כל חבר. בעת הניסיון להיכנס לעמוד של חבר הפרלמנט השני, הקוד קרס והעלה לנו exception בשם " stale element reference: element is not attached to the page document " שמבדיקה ב-stackoverflow עלה כי מדובר על בעיה שאומרת שהאובייקט שאנו מחפשות כבר לא קיים בדפדפן בשל שינויים שבוצעו במהלך העבודה עליו ע"י הזחלן. הועלו מספר פתרונות להילחם עם exception זה שלצערי לא צלחו לנו בניסיונות מימוש שלהם.  בסוף פתרנו בעיה זו באמצעות הגדרת אובייקט רשימת חברי הפרלמנט מתוך הלולאה הפנימית של הזחלן ולא מחוצה לה. כך לאחר כל איטרציה על חבר פרלמנט ספציפי, הרשימה של העמוד מוגדרת מחדש ואז באיטרציה הבאה הזחלן מצליח למצוא את האובייקט הבא ברשימה באמצעות האינדקס שלו שרץ על טווח מספר חברי הפרלמנט הקיימים בעמוד בו אנו נמצאים. בעצם הגדרת אובייקט רשימת החברים כל פעם מחדש גורם לכך שבין חבר פרלמנט אחד לשני, לאחר חזרה אחורנית בדפדפן, המידע לא נעלם אלא מוגדר מחדש וכך ניתן להמשיך הלאה.   1. אתגר נוסף אחרון עימו התמודדנו הוא אופן שמירת המידע שקצרנו על כל חבר פרלמנט כשורה (תצפית) בקובץ האקסל שלנו.   ניסינו לבחון מספר דרכים שיהיו יעילות ונכונות לביצוע שמירת המידע (שמירת כל תצפית/ שמירת רשימת מידע כללית שכל פעם יכנס אליה מידע מחבר אחר וכו'..) אך בסוף הגענו למסקנה שהדרך הטובה ביותר עבורנו היא אכן שמירת המידע ברשימה עבור כל חבר פרלמנט באופן פרטני ובסוף קצירת כל המידע אודותיו, העברת הרשימה ל-dataframe ומשם לקובץ האקסל.  הסיבה שבחרנו לעשות את העברת ושמירת המידע באופן הזה בין ה-dataframe לאקסל היא ניסוי ותהייה. תחילה בדקנו העברת כל נתון בפני עצמו, לאחר קצירתו, ל-dataframe אך זה גם סרבל את הקוד וגם האט את זמן הריצה שלו.  לאחר מכן בחנו שמירת כל שורת חבר פרלמנט ב-dataframe וישר עדכון שלה בדוח האקסל אך גם פעולה זו האטה את הקוד בצורה משמעותית ולכן העדפנו לשמור כל שורה ב-dataframe ורק לאחר סיום המעבר על כך חברי הפרלמנט להעביר את הכל לקובץ האקסל בצורה מסודרת. |