



Relatiive

Lost Detection

פרויקט גמר 2022

213289754 | תהילה אברהמי

סמינר: גברא

מנחה: יעל עמר

תאריך הגשה: 30.05

תוכן עניינים

4.....	1. הצעת פרויקט	4.
7.....	2. מבוא / תקציר	7.
7.....	2.1. הרקע לפרויקט	7.
9.....	2.2. תהליך המחקר	9.
12.....	2.3. סקירת ספרות	12.
13.....	3. מטרות ויעדים	13.
14.....	4. אתגרים	14.
15.....	5. מדדי הצלחה	15.
15.....	6. תיאור המצב הקיים	15.
16.....	7. רקע תאורטי	16.
23.....	8. ניתוח חלופות מערכתי	23.
23.....	9. תיאור החלופה הנבחרת והנימוקים לבחירה	23.
24.....	10. אפיון המערכת	24.
24.....	10.1. ניתוח דרישות המערכת	24.
25.....	10.2. מודול המערכת	25.
25.....	10.3. אפיון פונקציונאלי	25.
27.....	10.4. ביצועים עיקריים	27.
27.....	10.5. אילוצים	27.
28.....	11. תיאור הארכיטקטורה	28.
28.....	11.1. הארכיטקטורה של הפתרון המוצע בפורמט של Design level Down-Top	28.
28.....	11.2. תיאור הרכיבים בפתרון	28.
30.....	11.3. ארכיטקטורת רשת (לא רלוונטי)	30.
31.....	11.4. תיאור פרטוקולי התקשורת	31.
31.....	11.5. שרת – לקוח	31.
31.....	11.6. תיאור הצפנות (לא רלוונטי)	31.
32.....	12. ניתוח ותרשים use case של המערכת המוצעת	32.
32.....	12.1. רשימת use case	32.
32.....	12.2. תיאור ה-use case העיקריים של המערכת	32.
37.....	12.3. מבני נתונים בהם משתמשים בפרויקט	37.
38.....	12.4. תרשים מחלקות	38.
40.....	12.5. תיאור המחלקות	40.

43.....	תיאור התוכנה.....	13.
44.....	אלגוריתמים מרכזיים.....	14.
46.....	קוד האלגוריתם.....	15.
49.....	תיאור מסד הנתונים.....	16.
50.....	פירוט הטבלאות ב- Data Base.....	16.1.
52.....	מדריך למשתמש.....	17.
52.....	תיאור המסכים.....	17.1.
53.....	מדריך למשתמש.....	17.2.
54.....	צילומי מסכים.....	17.3.
60.....	בדיקות והערכה.....	18.
60.....	ניתוח יעילות.....	19.
60.....	אבטחת מידע.....	20.
61.....	מסקנות.....	21.
63.....	פיתוח עתידי.....	22.
64.....	ביבליוגרפיה.....	23.

1. הצעת פרויקט

סמל מוסד:

שם מכללה: סמינר בית יעקב

שם הסטודנט: תהילה אברהמי

ת.ז הסטודנט: 213289754

שם הפרויקט: Realatiive

תיאור הפרויקט:

האפליקציה מזהה פנים אנושיות עפ"י תמונה קודמת שהוא מכניס למערכת או עפ"י תמונות שלו שאחרים מעלה לאפליקציה בזמן אמת כדי למוצאו.

משתמש חדש יכול להירשם לאפליקציה. לצורך כך עליו להכניס את פרטיו האישיים: שם, שם משפחה, ת.ז ומספרי טלפון וכן תמונת פנים. כל אלה ישמרו במאגר המשתמשים.

ישנה אופציה נוספת של העלאת תמונות בזמן אמת. במצב של היעדרות אדם ישנה אופציה שהוא נמצא בבית החולים ללא זיהוי במיוחד במקרים של אירוע המוני כמו תאונה או פיגוע וכו'. במצב כזה קרוביו של אותו אדם יכולים להעלות לאפליקציה תמונות עדכניות של הנעדר וכך כאשר בבית החולים ינסו לזהותו ע"י צילום פניו הוא ימצא במאגר וכך יתאפשר הזיהוי.

בבית החולים כאשר הגיע חולה לא מזוהה על הצוות להכניס את תמונתו ולבדוק אם הוא נמצא במאגר וע"י כך ליצור קשר עם משפחתו וקרוביו.

הגדרת הבעיה האלגוריתמית:

- זיהוי פנים אנושיות מתמונה.
 - חילוץ הפנים מתמונה.
 - השוואת הפנים לפנים מתמונה אחרת מהמאגר ובכך לזהות את האדם המבוקש.
- לצורך כך עלי ללמוד לעומק על כל הנושא של זיהוי פנים מתמונה ועל השוואת פנים מתמונות שונות.

בהשוואת הפנים ישנה בעיה שלעיתים ישנם פערים בין הפנים בתמונה המקורית לבין התמונה העכשווית. ישנו פער גילאים של האדם בין התמונות השונות כמו כן בצילום חולה יתכן שפניו יהיו שונות מהמקור – עיניו עצומות או פרצופו פצוע, לפעמים יש עליו מכשירים שונים וכד'. עלי לנסות להתגבר על בעיה זו כדי להגיע לתוצאה הטובה ביותר.

בכתיבת האלגוריתם אני אמורה להשתמש בספריה OpenCV וכן בספריות נוספות לשם כך עלי ללמוד על הספרייה כיצד להשתמש בה בצורה יעילה.

נוסף על כך אני צריכה לחפש אלגוריתמים בתחומים של זיהוי והשוואת פנים שיעזרו לי בכתיבת האלגוריתם וללמוד עליהם כיצד עובדים ואיך אני אוכל להיעזר בהם. מתוכם עלי למצוא את האלגוריתם היעיל ביותר לצורך ביצוע המשימה.

לצורך שמירת המידע במסד נתונים עלי להשתמש ב-mongoDB שהוא בסיס נתונים הנמצא בקטגוריית NoSQL ועובד במבנה של מסמכים (בשונה מבסיסי נתונים טבלאיים כמו SQL ועוד). לשם כך עלי ללמוד על בסיס נתונים זה כיצד עובד וכיצד אוכל להשתמש בו.

רקע תאורטי בתחום הפרויקט:

האפליקציה יעילה לבתי חולים. במקרה של חולה חסר הכרה שמגיע לבית החולים ללא זיהוי ניתן לצלם את תמונתו ואם הוא נמצא במאגר יהיה ניתן לזהותו בקלות.

יתר על כן ישנה אפשרות של זיהוי בזמן אמת. במקרים שבהם נעדר אדם ובני משפחתו חוששים אולי הוא נמצא בבית החולים ניתן להעלות לאפליקציה תמונות עדכניות שלו בתוספת פרטים מזהים וכך כשבבית החולים ירצו לזהותו זה יתאפשר בקלות רבה.

תהליכים עיקריים בפרויקט:

1. משתמש רוצה להירשם לאפליקציה. מזין את פרטיו האישיים ותמונת פרופיל שלו.
2. צוות בית החולים מעלה תמונות למאגר במקרה של מטופל אלמוני.
3. משפחה המחפשת נעדר מעלה את תמונתו כדי לבדוק האם הוא הגיע לבית החולים.
4. האפליקציה מזהה את תמונת האדם במקרה שמגיע לבית החולים ללא זיהוי.
5. האפליקציה מודיעה לבית החולים לגבי ההתאמה. באחריות בית החולים ליצור קשר עם משפחת החולה.

תיאור הטכנולוגיה: שרת REST, SPA / WINFORMS

צד שרת:

שפת תכנות בצד השרת: python

צד לקוח:

שפת תכנות בצד לקוח: ionic angular

מסד נתונים: mongoDB

פרוטוקלי תקשורת: אין.

לוחות זמנים:

1. חקר המצב הקיים – ספטמבר
2. הגדרת הדרישות – ספטמבר
3. אפיון המערכת – אוקטובר
4. אפיון בסיס הנתונים – נובמבר
5. עיצוב המערכת – דצמבר
6. בניית התוכנה – ינואר
7. בדיקות – מרץ
8. הכנת תיק פרויקט – אפריל
9. הטמעת המערכת – מאי
10. הגשת פרויקט סופי - מאי

חתימת הסטודנט: תהילה

חתימת רכז המגמה:

אישור משרד החינוך:

2. מבוא / תקציר

2.1. הרקע לפרויקט

כשהתחלנו לחשוב על רעיון לפרויקט רציתי רעיון שיהיה לי מעניין לעשות, לא רציתי לעשות פרויקט משעמם, העדפתי להשקיע בפרויקט שיכול להיות שיהיה קשה יותר אבל גם מהנה.

אחת ממטרות הפרויקט היא פיתוח מיומנויות חשיבה, להביא אותנו להתנסות עצמית. כשאני יושבת על פרויקט משלי שאני אמורה לעשות לבד מתחילתו ועד סופו אני אמורה ללמוד לבד חומרים שלא למדנו בכיתה, לנתח פרויקט ולהבין מה אני צריכה לבצע, להכיר קודים ואלגוריתמים שונים, לנסות ולהתנסות. וזה בעצם חלק ממטרות הפרויקט – להביא אותי בסופו של דבר למוכנות לעבודה כי הרי לימודים ועבודה זה עולמות שונים.

רציתי לנצל את הפרויקט כדי ללמוד עוד על תחומים שלא יצא לי להכיר ואכן זה מה שגם גרם לבחור דווקא את התחום של הזיהוי פנים – התחום נשמע לי מעניין, מרתק ושונה. מעולם לא יצא לי להתעסק בתחום הזה לפני כן וניצלתי את ההזדמנות ללמוד ולהכיר דברים חדשים. ואכן היה לי הרבה מה ללמוד. אני מרגישה שהפרויקט הרחיב לי אופקים התחום הצריך הרבה למידה עצמית, התנסות וכישלון וזה הביא אותי לרמה הרבה יותר גבוהה.

לאחר שהחלטתי על נושא לפרויקט וחקרתי אותו קצת הבנתי עד כמה הוא חשוב ונצרך. כל העניין הזה של איתור נעדרים הוא דבר שכמעט ולא קיים ויכול לתרום הרבה לאנושות ברמה שיכול ממש להציל חיי אדם במקרי היעדרות של מבוגרים או ילדים קטנים. החלטתי שאני מאוד רוצה לקחת דווקא את הנושא הזה כי אני רואה חשיבות הרבה ונחיצותו.

הדבר שהכי חיזק בי את ההחלטה לבחור ברעיון של איתור נעדרים כרעיון לפרויקט גמר הוא הסיפור הטרגי שארע במירון שנה שעברה. אני חושבת שאם הייתה קיימת כזאת אפליקציה הייתה נחסכת המון עוגמת נפש ולחץ של אימהות דואגות וקרובי משפחה שחיפשו את בני משפחתם ואפילו הגיע מצב של אימהות שהיו בטוחות שילדיהם בין ההרוגים ואפליקציה מעין זו יכולה להוריד במעט את הדאגה העצומה של אלפי אנשים.

גם ביום יום לאו דווקא באירועים המוניים וחריגים אפליקציה זו יכולה להציל חיים. סיפורים על מבוגרים חסרי ישע נעדרים וילדים קטנים שהולכים לאיבוד מתרחשים לצערנו בכל יום ולכן אפליקציה כזאת היא דבר נחוץ והכרחי. לאחר שתפוח אפליקציה כזאת מקרים רבים של היעדרות יפתרו במהירות ולא יצטרכו להמתין ימים שלמים עד שהמשטרה תעזור בעניין וימצא הנעדר.

האלגוריתם שבו בחרתי ליישום התוכנית הוא מתחום הלמידה עמוקה. האלגוריתם הוא Image similarity שהוא בעצם מודל העוסק בזיהוי פנים ומציאת דמיון בין תמונות. השם שבחרתי לפרויקט הינו Relatiive - השם מורכב מצירוף המילים relate שמשמעותה לייחס, לשייך ו-relative שמשמעותה קרוב משפחה. לסיכום אני מאוד מקווה שהפרויקט יועיל ויעזור לשיפור האנושות, אני חולמת שאנשים באמת יוכלו להשתמש באפליקציה והיא באמת תהיה יעילה ושימושית. בנוסף הקוד יהיה קוד פתוח ואנשים רבים יוכלו להיעזר בקוד ולהשתמש בו לצרכיהם שונים.

2.2 תהליך המחקר

כשהתחלתי לפתח את הפרויקט דבר ראשון הלכתי לחפש חומרים. קראתי הרבה מאמרים בנושאים שקשורים (יותר או פחות) לפרויקט. חקרתי והכרתי לעומק את כל הנושא של זיהוי והשוואת פנים – איך זה מתבצע, באיזה ספריות משתמשים, איזה אלגוריתמים קיימים בתחום וכו'. בהמשך כשידעתי מספיק על הנושא התחלתי לעבוד על הפרויקט בפועל וליישם את כל מה שקראתי ולמדתי.

את החומר הרב שקראתי מצאתי בגוגל באתרי למידה ומידע שונים כמו: Internet Israel, Stack Overflow וכד'. כמו כן ראיתי סרטונים רבים בנושאים של זיהוי פנים.

לאחר שחקרתי היטב את הנושא וקראתי חומר רב החלטתי להשתמש בתחום של למידת מכונה. למידת מכונה הינה תחום מחקר המאפשר למחשבים את היכולת ללמוד מבלי להיות מתוכנתים באופן ספציפי.

המטרה המרכזית של למידת המכונה היא טיפול ממוחשב בנתונים מן העולם האמיתי עבור בעיה מסוימת, כאשר לא ניתן לכתוב תוכנת מחשב עבורה למשל, בעיית זיהוי שמומחה אנושי מסוגל לפתור, אך לא מסוגל לכתוב את הכללים לזיהוי בצורה מפורשת, או שהם משתנים עם הזמן ולא ניתנים לכתיבה מראש.

תהליך הלמידה של המערכת נקרא "אימון". בשלב האימון, מציגים למערכת את הדוגמאות והמערכת לומדת היפותזה שמתארת בצורה הטובה ביותר את הדוגמאות שהיא ראתה.



בלמידה מונחית, כל דוגמה שהמערכת לומדת על פיה היא זוג המורכב מאובייקט קלט ומערך הפלט הרצוי בעבור אותו הקלט. לאוסף הדוגמאות קוראים בשם "סט האימון". סט האימון מכיל דגימות שמגיעות מהתפלגות משותפת של מרחב האלמנטים ומרחב התיוגים. אימון אלגוריתם למידה מונחית דורש בדרך כלל סט אימון שמכיל דוגמאות רבות. מטרת הלמידה היא ללמוד לנבא את התיוג של אלמנט חדש שנוצר מאותה ההתפלגות אם המודל שאנחנו מאמנים הוא מופלה, ואילו בעבור מודל גנרטיבי מטרת הלמידה תהיה ללמוד את ההתפלגות שממנה נוצרו הדוגמאות. לדוגמה, אם הבעיה שאנחנו רוצים ללמוד היא סיווג של תמונות לפי האובייקט המופיע בהן, למשל לקבל תמונה ולומר האם מופיע בה כלב או חתול, כל דוגמה בסט האימון תורכב מהקלט - התמונה, ולכל קלט יוצמד הפלט הרצוי - התיוג "כלב" או "חתול" לפי האובייקט המופיע באותה התמונה.



בשלב זה של הפרויקט התייעצתי עם מתכנת המתמחה בתחום של למידת מכונה ואימון מודלים. התחום של למידת מכונה הינו תחום מורכב הדורש למידה והכרת הנושא. באופן רגיל אין אפשרות ללמוד אותו בצורה עצמאית ומהירה. לכן המתכנת שאתו התייעצתי עזר לי בתחום, הוא נתן לי ידע רב שעזר לי להכיר את הנושא, הדריך אותי כיצד משתמשים, ואיך

לעבוד עם אימון מודל. בנוסף הוא נתן לי עצות כיצד להתקדם בפיתוח הפרויקט בצורה יעילה ומקצועית.

2.3 סקירת ספרות

האתרים בהם השתמשתי לביצוע הפרויקט הם:

חקר הפרויקט –

- ויקיפדיה
- Microsoft
- Internet Israel
- OpenCV
- DataCamp

אלגוריתם –

- GeeksForGeeks
- Stack OverFlow
- GitHub

שפות –

- MongoDB
- Medium
- Dwh.co.il

עיצוב –

- Bootstrap
- W3School

3. מטרות ויעדים

כ-250 בני אדם נעדרים בכל יום בישראל, בין אם אלו קשישים עם דמנציה, אנשים הסובלים מאוטיזם ונחשבים בתפקוד נמוך או ילדים שהלכו לאיבוד.

למטרה זו נועדה Relatiive - לעזור לאנשים שנראים חסרי ישע או איבדו את דרכם וכן ילדים קטנים המשוטטים לבדם, כדי לנסות לחבר אותם למשפחה המודאגת.

בבניית מערכת ובמיוחד מערכת כזו שמטרתה איתור נעדרים - אחד הדברים החשובים הוא להקפיד על מערכת ברורה ונוחה למשתמש. לרוב האנשים אין את הזמן והכוח לנסות להבין איך המערכת עובדת וכיצד להשתמש בה ולכן על המתכנת לבנות את המערכת בצורה שיהיה למשתמש קל ונוח להתעסק בה כמובן שחשוב שהמערכת תעבוד בצורה יעילה ונכונה שהרי היא נועדה לזיהוי נכון של נעדרים.

לשם הקמת המערכת וכדי שהיא תתפקד בצורה המיטבית הצבתי לעצמי כמה מטרות ובשביל כך כמה יעדים:

מטרות המערכת:

1. אפשרות לאיתור נעדרים בצורה מהירה ויעילה.
2. נוחות וידידותיות למשתמש.
3. לחסוך התערבות משטרתית בכל אירוע של היעדרות.
4. לאפשר לבתי חולים לזהות חולים לא מזוהים המגיעים לבית החולים חסרי הכרה.

יעדי המערכת:

1. בניית מערכת חכמה לזיהוי פנים אנושיות.
2. הוספת אפשרות ליצירת חשבון אישי ובו העלאת תמונה שתאפשר הגנה על המשתמש במקרה היעדרות.
3. יצירת המערכת כאפליקציה המפותחת בצורה נוחה וקלה למשתמש.
4. ניתן להעלות תמונה של אדם הנראה אבוד גם ללא התחברות למערכת וכך לגלות את זהותו – אם קיים במערכת.

4. אתגרים

כשניגשתי להתחיל את הפרויקט ידעתי שהוא לא הולך להיות קל אבל לקחתי את זה בחשבון. ידעתי שאם אני רוצה להצליח אני חייבת להתאמץ ולהשקיע אפילו שיהיה קשה.

בתחילת תכנון הפרויקט ושלבי העבודה לא ידעתי בכלל מאיפה להתחיל, הפרויקט היה נראה לי אתגר ענק ובלתי אפשרי, היו המון נושאים שלא הכרתי ומעולם התעסקתי בהם. עכשיו לאחר בניית הפרויקט אני מבינה שהכול אפשרי וגם דברים שנראים בלתי אפשריים בסוף עבירים.

לצורך שמירת התמונות הייתי צריכה להשתמש במסד נתונים השונה מ SQL – שאותו למדנו בכיתה ולכן הייתי צריכה ללמוד על MongoDB.

MongoDB הינו בסיס הנתונים נשען על מבנה של מסמך בניגוד לבסיסי נתונים טבלאיים (כמו Oracle, SQL Server ו MySQL) העובדים מעל טבלאות מקושרות. מבנה המסמכים עובד מעל מימוש של JSON הנקרא על ידי MongoDB - BSON.

למידת הנושא בצורה עצמאית היוותה לי מעט קושי מכיוון שהיה עלי לחפש חומרים שיעזרו לי ללמוד על הנושא, להכיר אותו ולהצליח בסופו של דבר לעבוד עם זה ולשמור בו את הנתונים.

כשהתחלתי בכתיבת האלגוריתם לזיהוי פנים מצאתי קוד שהיה אמור לעזור לי בכך אך לאחר עבודה קשה ומאומצת התברר שהקוד לא הועיל במאום והיה עלי לחפש קוד חלופי אחר.

לאחר שכתבתי ומחקתי פעמים רבות החלטתי לשנות כיוון ולעבור לתחום של למידת מכונה. בשלב זה התייעצתי עם מתכנת שעזר לי לעבור ללמידת מכונה ולהכיר את התחום. גם שלב זה לא היה לי קל, היה עלי ללמוד תחום חדש שמעולם לא התעסקתי בו. תחום הלמידה עמוקה הינו תחום מרתק ומעניין אך מצריך למידה מוקדמת ומקצועית. את כל הלמידה הזאת ערכתי תוך פשוט משבוע בשל לחץ הזמן והדחק. היה עלי להגביר הילוך ולהתחיל להתקדם בצורה מהירה יותר.

גם כשהתחלתי להתעסק עם המודל ולאמן אותו הייתי זקוקה ללמידה והדרכה מתמדת כל הזמן. החומר היה זר לי ולא מוכר והיה עלי להשקיע בו יותר מחשבה משום ששם נכתב הרעיון האלגוריתמי.

כך, בכל מהלך הפרויקט נתקלתי בקשיים ובאתגרים חלקים אף לא צפויים אך למרות זאת אני שמחה באתגרים הללו כי הם בסופו של דבר חישלו אותי והכירו לי את התחום לעומק ובצורה רחבה יותר.

5. מדדי הצלחה

- הוספת משתמש חדש למערכת ושמירתו בData base.
- כניסה לאזור האישי בצורה מאובטחת – באמצעות מספר זהות וסיסמה.
- זיהוי נעדר באמצעות תמונה.

6. תיאור המצב הקיים

כ-250 בני אדם נעדרים בכל יום בישראל, בין אם אלו קשישים עם דמנציה, אנשים הסובלים מאוטיזם ונחשבים בתפקוד נמוך או ילדים שהלכו לאיבוד. המצב הזה לצערנו הינו מצב יומיומי ורגיל, למשטרה אין הרבה יכולות למציאת נעדרים ולעיתים ישנם מצבים בהם הנעדר אינו נמצא למשך ימים ארוכים. גם במקרים שבהם ילדים קטנים נעלמים אפילו למספר שעות אפליקציה מעיין זו יכולה להציל חיים כי ילד קטן חשוף לסכנות בכל רגע לבדו. נוסף על כך הדאגה העצומה של אימהות לילדים ושל קרובי משפחה לנעדר יכולה להוציא אנשים ומדעתם והאפליקציה הינה רעיון אדיר למצבים כאלו.

לאחר בירור וחיפוש הגעתי למסקנה שאין מענה לבעיה זו, אנשים רבים נעדרים ואורך זמן רב למציאת קצה חוט. ולכן אני חושבת שאפליקציה כזו תשנה את פני האנושות, אנשים רבים ינצלו בזכותה ויותר מכך אנשים רבים יזכו להציל חיים בלי מאמץ וקושי רב. שהרי "כל המציל נפש אחת מישראל כאילו קיים עולם מלא" ואין זכות גדולה מזו.

אני מאמינה שאנשים רבים משוועים לאפליקציה כזו שתעזור להם למצוא את קרוביהם במהירות ועוד לפני שיספיקו לדאוג.

7. רקע תאורטי

לצורך פיתוח המערכת השתמשתי באלגוריתם Image Similarity. אלגוריתם זה עובד באמצעות למידה עמוקה. באופן כללי ניתן להשתמש בלמידת מכונה כדי להחזיר תמונות דומות, טקסט או אודיו בצורה פשוטה ומהירה.

נתאר לעצמינו את המאמץ התכנותי הדרוש כדי ליישם אלגוריתם להשוואה ויזואלית של תמונות פנים שונות כדי למצוא מספר תמונות של אותו בן אדם. יישום אלגוריתם מעיין זה הוא מאתגר אך בעזרת למידה עמוקה נוכל לבצע זאת בקלות, למספר בלתי מוגבל של מקרי שימוש שונים. בחרתי לעשות זאת באמצעות הייצוג החזותי הנלמד של מודל Deep Learning.

כעת אפרט דרך פשטה ומהירה ליישום האלגוריתם Image Similarity.

מצבי שימוש:

אתחיל בכמה מצבי שימוש באלגוריתם כדי לתת קצת השראה לפני שאעמיק בפרטים הטכניים. מקרי השימוש הם אינסופיים וניתן להשתמש ב - Image Similarity בתחומים רבים ושונים כמו:

מסחר אלקטרוני – למסחר אלקטרוני יש שימושים רבים ב - Image Similarity . העיקרי ביותר, מציאת מוצרים דומים על סמך תמונת מוצר בפועל. ישנם גם פחות שימושיים כמו שימרה על קטלוג המוצרים נקי על ידי מניעת העלאת תמונות מרובות של אותו מוצר.

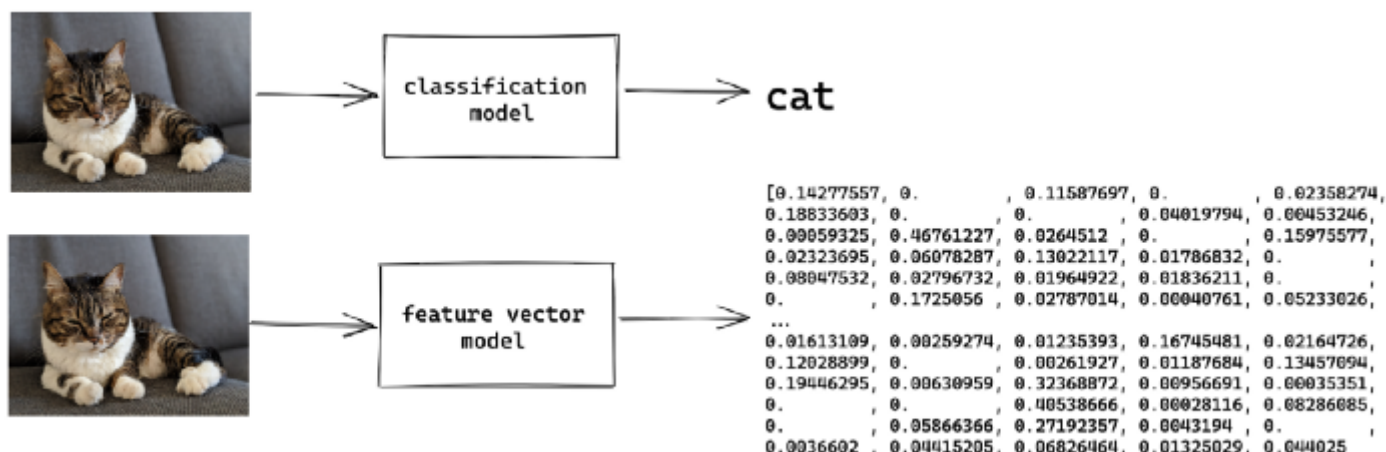
ארכיון מסמכים דיגיטלי – נדמיין ארכיון דיגיטלי עם מיליוני מסמכים סרוקים, גדולים ולא מובנים. הצבת התוויות הנכונות כדי למצוא מסמכים דומים גוזלת זמן רב. שימוש בגישת למידה עמוקה מסיר את הצורך לשים תוויות על התמונות.

ייצוג חזותי של תמונה

כאשר משתמשים במודל למידה עמוקה משתמשים לרוב בשכבה האחרונה של המודל, שכבת הפלט. שכבה זו נותנת למשל את המחלקה של התמונה . האם בתמונה זה חתול, מכונת, שולחן, כלב או עכבר ?

ניתן להסיר את שכבת הסיווג הזו ולקבל את הייצוג החזותי של המודל המיוצר עבור תמונת הקלט.

הייצוג הוא וקטור של מספרים בין 0 ל-1.



נסה להכליל את הגישה ולהשוות אותה לאופן שבו אנו כבני אדם תופסים את העולם. למדנו בילדותנו פרטים ייחודיים כמו מכונית היא ברובה מרובעת, ויש לה צמיגים, אורות ודלתות. המספרים של הווקטור עשויים לייצג פרטים דומים. מודל הלמידה העמוקה למד את המידע הזה במהלך ההדרכה. מה שהדגם למד הוא "בעיקר" קופסה שחורה. מאמצים אחרונים במחקר עוזרים לנו להבין זאת טוב יותר.

מה שעלי למצוא כעת זה מודל מאומן מראש כדי לקבל וקטור כזה שנקרא גם דגם מיומן מראש. במצב כזה ניתן את אחד ממודלים הרבים של למידה עמוקה שהוכשרו מראש, הם מכילים מספיק כדי להתאים לרוב מקרי השימוש. למרות זאת החלטתי לא לכוון את הדגם המאומן מראש עבור מקרה השימוש הספציפי שלי.

TensorFlow מקל על שימוש חוזר בתכונות תמונה שכבר הוכשרו מראש ובמודלים וקטוריים. כעת מאמנים את המודל באמצעות TensorFlow Keras. צורת הקלט מגדירה את גודל התמונה שעליו הוכשר המודל. זה אומר שהמודל למד למצוא דפוסים בגדלים מסוימים, עושים זאת בצורה נכונה באמצעות מעקב אחר גודל התמונה המומלץ.

```

1 import tensorflow as tf
2 import tensorflow_hub as hub
3
4 model_url = "https://tfhub.dev/tensorflow/efficientnet/lite0/feature-vector/2"
5
6 IMAGE_SHAPE = (224, 224)
7
8 layer = hub.KerasLayer(model_url, input_shape=IMAGE_SHAPE+(3,))
9 model = tf.keras.Sequential([layer])

```

Embedding

כדי לקבל את ההטמעה מזינים תמונה ונותנים למודל לבצע את החיזוי שלו. התמונה עצמה זקוקה למעט עיבוד מקדים.

1. שינוי גודל התמונה לגודל שעליו הוכשר המודל.
2. המרה התמונה לייצוג צבע עבור כל פיקסל.
3. נרמול הערכים בין 0 ל-1.

```

1 file = Image.open(file).convert('L').resize(IMAGE_SHAPE) #1
2 file = np.stack((file,)*3, axis=-1) #2
3 file = np.array(file)/255.0 #3
4
5 embedding = model.predict(file[np.newaxis, ...])
6 embedding_np = np.array(embedding)
7 flattened_feature = embedding_np.flatten()
8
9 print(flattened_feature)

```

בעזרת שורות הקוד הבודדות הללו, אוכל לקבל את הייצוג החזותי, וקטור של מספרים צפים בין 0 ל-1. אורך הווקטור שונה בהתאם לדגם. בדוגמה כאן, למודל המשומש יש אורך וקטור של 1280 מספרים.

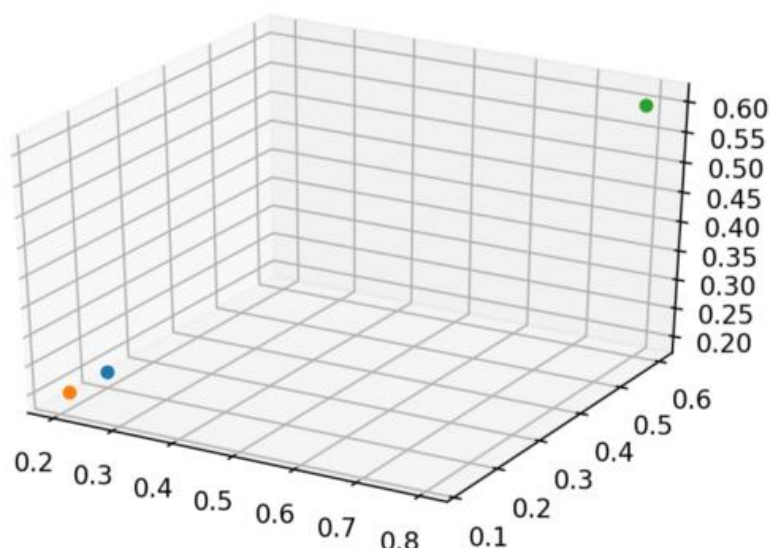
אנחנו יכולים להשתמש בייצוג החזותי הזה כדי לחשב כמה תמונות דומות.

כיצד למדוד דמיון תמונה

כדי לחשב את הדמיון בתמונה אנחנו צריכים מדד.

נדמין מערכת קואורדינטות עם 3 צירים x , y ו- z . נניח שלוקטור התכונה שלנו יש אורך של 3 אלמנטים במקום 1280. נדמין גם שיש לנו את וקטורי התכונה הבאים עבור 3 תמונות המיוצגות באותה מערכת קואורדינטות.

- חתול – $[0.2, 0.1, 0.1]$ (נקודה כחולה)
- חתול – $[0.2, 0.12, 0.2]$ (נקודה כתומה)
- שולחן – $[0.6, 0.6, 0.8]$ (נקודה ירוקה)

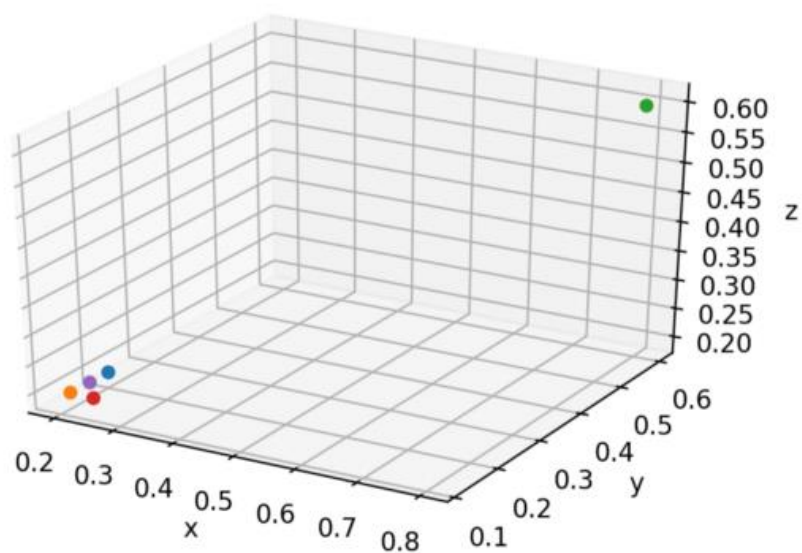


המרחק האוקלידי, כפי שהשם מרמז מודד את המרחק בין נקודות. אנו יכולים לראות בבירור שהמרחק של החתולים (כתום וכחול) זה מזה קטן יותר בהשוואה לשולחן (ירוק). למרות שזה עובד עבור המקרה הספציפי הזה, יתכן שהוא לא יתאים כל הזמן. לכן נוסיף שני וקטורים נוספים.

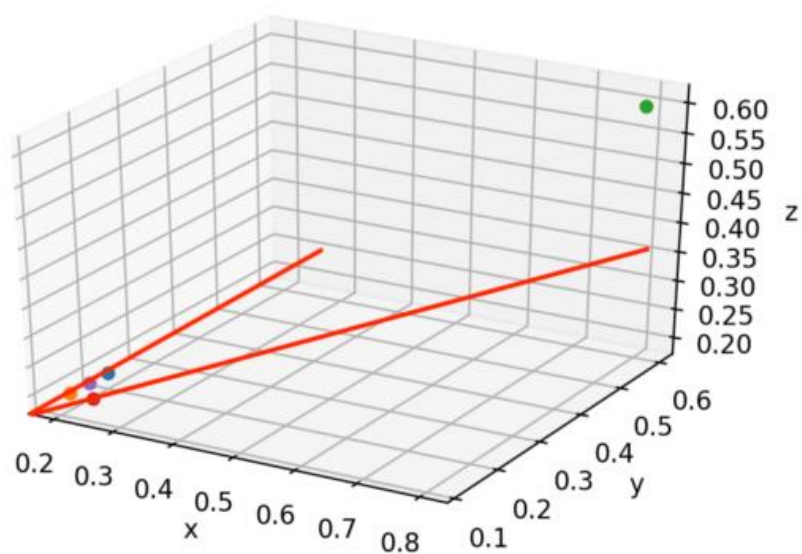
- עכבר – $[0.6, 0.6, 0.8]$ (נקודה אדומה)
- חתול – $[0.2, 0.16, 0.2]$ (נקודה סגולה)

אנחנו יכולים לראות שהעכבר (אדום) קרוב לחתול (סגול) ואם מודדים את המרחק האוקלידי למעשה הוא אפילו יותר קרוב מהחתול השני (כחול). אנחנו צריכים גורם

בנוסף כדי למדוד את הדמיון במדויק. אנחנו יכולים להשתמש בכיוון של הנקודות בתוך מערכת הקואורדינטות.



הדבר נעשה ברור יותר אם נצייר כמה קווים הממחישים את הכיוון, כך נוכל לראות תמונות דומות חולקות את אותו הכיוון. מדד הנקודה בהשוואה לאוקלידי משתמש במרחק ובנוסף בכיוון כדי לחשב את הדמיון.



המדד האחרון הוא הדמיון הקוסינוס הוא רק לוקח בחשבון את הכיוון ללא המרחק בין שתי הנקודות. הבחירה ביניהם תלויה במקרה השימוש הספציפי.







הדוגמה שסיקרנו זה עתה היא פשוט, וקטורי התכונה שאנו מקבלים מהמודלים שלנו הם בעלי ממדים גבוהים בהרבה.

כדי לחשב את ערכי הדמיון איננו צריכים ליישם את המתמטיקה בעצמנו. ישנם ספריות רבות המספקות את השיטה הזאת עבורנו, לשימוש.

עבור דוגמא זו משתמשים ב- SciPy אך ניתן גם להשתמש ב- Sklearn.

```
1 from scipy.spatial import distance
2 metric = 'cosine'
3
4 cosineDistance = distance.cdist([cat], [rocket], metric)[0]
5 print(cosineDistance)
6 print("the distance between cat and the rocket is {}".format(cosineDistance))
```

ככל שהערך קטן יותר, התמונות אכן דומות יותר. ניתן לראות שהחתולים קרובים מאוד זה לזה עם ערך של 0.45. בעוד ההבדל בין החתולים לטיל עם ערך גבוה יותר של 0.731 ו- 0.690, כלומר דמיון נמוך יותר.

			
	0	0.731	0.345
	0.731	0	0.690
	0.345	0.690	0

כיצד להתאים את המודל למיליוני תמונות

איטרציה של יותר ממיליוני תמונות כדי למצוא את התמונות הדומות ביותר לא משתנה מאוד אם אנחנו צריכים לעבד את כל התמונות בכל פעם שאנחנו רוצים למצוא תמונה דומה. כדי לייעל את התהליך היה עלי לבנות איזשהו אינדקס ולמצוא דרך לחזור עליו בצורה יעילה יותר.

8. ניתוח חלופות מערכתי

ישנן אפשרויות שונות לפתרון הבעיה האלגוריתמית. אחת מהן היא זיהוי פנים בצורה ידנית. האפשרות היא לכתוב אלגוריתם לזיהוי הפנים עצמם בתוך התמונה, לאחר מכן יש לחלץ את הפנים לשמור מהם את האיברים המייחדים את הפרצוף (כמו גבות, אף, פה) באמצעות וקטור. עושים פונקציה של דיסטנס בין המרחקים שנשמרו בווקטור וכך משווים בין תוצאות של תמונות.

9. תיאור החלופה הנבחרת והנימוקים לבחירה

המערכת שפיתחתי מתבססת על מודל למידה עמוקה. המודל הזה היוו אלגוריתם הנקרא Image Similarity. המודל לומד למדוד מרחק בין נקודות נתונים. בדרך כלל היו בוחרים במרחק סטנדרטי תוך שימוש בידע הפריורי בתחום. עם זאת קשה לעצב נתונים למדדים למשימה הספציפית של העניין.

למידה מטריית שואפת לבנות באופן אוטומטי מדדי מרחק ספציפיים למשימה מנתונים בפיתוח באופן למידת מכונה. לאחר מכן ניתן להשתמש במדד המרחק הנלמד כדי לבצע משימות שונות כמו זו של אחזור מידע באמצעות תמונה.

לעומת זאת האפשרות הקודמת יכולה להיות אפשרות נכונה ולהביא תוצאות אך הזיהוי לא יעשה בצורה מושלמת ולכן בחרתי באפשרות של אימון מודל. האפשרות הזאת מניבה תוצאות מדויקות יותר והרבה יותר נכונות. העדפתי לעבוד קצת יותר קשה וללמוד חומר שהיה חדש לי וקצת יותר מורכב מהמוכר והידוע אך הפרויקט אכן יצא טוב יותר ונעשה בצורה הרבה יותר מקצועית.

10. אפיון המערכת

סביבת פיתוח:

חומרה: מעבד RAM 16.0 GB

Intel(R) Core(TM) i7-7700 CPU @ 3.60GHz 3.60 GHz

עמדת פיתוח: מחשב DALL

מערכת הפעלה: windows 10

שפות תוכנה: C#, Phyton, תוך שימוש בטכנולוגית WebApi, Angular.

כלי תוכנה לפיתוח המערכת: vs code, Microsoft Visual Studio 2019, Jupyter, notebook, mongoDB compass.

מסד נתונים: MongoDB

עמדת משתמש מינימאלית:

- חומרה: מעבד RAM GB4 i5
- מערכת הפעלה: Windows 7 ומעלה.
- חיבור לרשת: נדרש.
- Chrom: תוכנות.

10.1. ניתוח דרישות המערכת

דרישות בהן המערכת צריכה לעמוד:

- כתיבה בסטנדרטים מקצועיים.
- מחשוב השרות ללקוח.
- כתיבת הקוד בסיבוכיות היעילה ביותר.
- ממשק נוח וידידותי למשתמש.
- תגובה מהירה ככל שניתן למשתמש.

10.2. מודול המערכת

- הלקוח מעלה תמונת נעדר למערכת.
- העברת התמונה מצד לקוח לצד שרת.
- הפעלת פונקציה לזיהוי פנים.
- הפעלת פונקציה להשוואת פנים הבודקת האם הנעדר קיים במאגר – מוגן ע"י האפליקציה.
- החזרת תגובה למשתמש: פרטי יצירת קשר במקרה של מציאת התאמה ואין התאמה במצב שלא קיים.

10.3. אפיון פונקציונאלי

פונקציות בשלב עיבוד הנתונים:

Create – פונקציה זו יוצרת את נתוני האימון - עריכת עיבוד מקדים לתמונות הנקרא pre processing. כמו כן הכנסת התמונות המעובדות לתוך מערך לצורך האימון.

```

26
27 def create_training_data():
28     for category in CATEGORIES :
29         path = os.path.join(DATADIR, category)
30         class_num = CATEGORIES.index(category)
31         for img in os.listdir(path):
32             img_array = cv2.imread(os.path.join(path, img))
33             image = cv2.resize(img_array, (IMG_SIZE, IMG_SIZE)).flatten()
34             training_data.append([image, class_num])
35

```

Reshape – פונקציה זו הינה פונקציית ספריה ומטרתה להכין את ה Data במבנה המתאים בווקטורים.

```

48
49 X = np.array(X).reshape(-1, IMG_SIZE, IMG_SIZE, 3)
50

```

פונקציית יצירת המודל:

```

51 # Creating the files containing all the information about your model
52 pickle_out = open("X.pickle", "wb")
53 pickle.dump(X, pickle_out)
54 pickle_out.close()
55
56 pickle_out = open("y.pickle", "wb")
57 pickle.dump(y, pickle_out)
58 pickle_out.close()
59
60 pickle_in = open("X.pickle", "rb")

```

פונקציית אימון המודל:

```

# Training the model, with 40 iterations
# validation_split corresponds to the percentage of images used for the validation phase compared to all the ima
history = model.fit(X, y, batch_size=32, epochs=40, validation_split=0.1)

```

```

# Compiling the model using some basic parameters
model.compile(loss="sparse_categorical_crossentropy",
              optimizer="adam",
              metrics=["accuracy"])

```

10.4. ביצועים עיקריים

משתמש יכול ליצור לעצמו חשבון ובו הוא יוצר הגנה על עצמו במקרה של היעדרות חלילה.

המשתמש מזין את פרטיו האישיים כנדרש, יוצר טופס יצירת קשר ומעלה תמונה עדכנית שלו למקרה הצורך.

כאשר נמצא ילד קטן לבדו, אדם מבוגר ללא זהות או חולה בבית החולים חסר הכרה מוטלת אחריות על מי שמוצא אותו לצלמו ולהעלות את התמונה ל - Relatiive וכך קיים סיכוי שאם הוא אכן מוגן ע"י האפליקציה תמצא התאמה ותתברר זהותו וכך יהיה למי לפנות ליצירת קשר.

10.5. אילוצים

המערכת מבצעת את ההתאמה במקרה שהתמונה עדכנית וברורה. לא תתבצע התאמה במקרים שבהם ישנו שוני בפנים ולא ניתן לזהותו כמו פצוע שמגיע לבית החולים והוא חבוש או פצוע בפנים. האפליקציה לא באה במטרה לגבור על היכולות האנושיות ולכן במקרה שאפילו אדם לא יכול לזהות כמובן שהאפליקציה לא תוכל לבצע את הזיהוי.

11. תיאור הארכיטקטורה

11.1. הארכיטקטורה של הפתרון המוצע בפורמט של Design level Down-Top

צד השרת - server side פותח במודל 3 השכבות ומתחלק ל-4 פרויקטים

החלוקה לשכבות נועדה להפריד באופן מוחלט בין הלוגיקה של הפרויקט לבין הנתונים עצמם. הפרדה זו מאפשרת לבצע שינויים בכל אחת מהשכבות בלי תלות ובלי זעזועים בשכבות האחרות.

API – שכבת ה Controller – חיבור בין צד השרת והלקוח.

BL – ניהול המידע ב – Data base.

DAL – מכיל את הפונקציונאליות הנדרשת לכל התקשורת עם ה - Data Base.

Models – מכילה מחלקות המתארות את הנתונים ובמבנה זה מעבירים את הנתונים בין השכבות.

מטרת שכבה זו היא למנוע תלות של שכבת ה - BL במבנה בסיס הנתונים. שכבת ה - BL מכילה פונקציות המרה מטיפוס הנתונים של בסיס הנתונים לטיפוס הנתונים של שכבת ה - Models ולהיפך, וכך מיוצגים הנתונים בכל הפרויקט.

11.2. תיאור הרכיבים בפתרון

הפרויקט מחולק ל-2 חלקים:

- צד שרת - הנכתב בשפת C# ובטכנולוגיית WebApi.
- צד לקוח - נכתב בשפת Angular ובטכנולוגיית TypeScript, Html.

בחרתי לכתוב צד לקוח ב- אנגולר שהינה שפה מתקדמת ועדכנית בעלת מאפייני Angular8 חדשניים ופונקציונאלית ביותר.

אנגולר הינה סביבת עבודה שפותחה על ידי גוגל. מאפשרת לפתח אפליקציות Framework אינטרנט בקלות ומהירות. במקור היא באה לתת מענה

לבניית Applications Page Single בצורה מושלמת ומהירה. מהיתרונות הבולטים והעיקריים של אנגולר אפשר למנות: חיסכון במשאבים, מהירות ביצוע, קוד קצר יותר, רוב העבודה מתבצעת בצד הלקוח ופחות בשרת ויכולת התמודדות טובה סיון מהיר ופשוט לביצוע של תוכן המתקבל מהשרת לפי מספר רב של פרמטרים.

צד שרת בחרתי לכתוב ב-C#. C# היא שפת תכנות עילית מרובת-פרדיגמות, מונחית עצמים בעיקרה המשלבת רעיונות כמו טיפוסיות חזקה, אימפרטיביות, הצהריות, פונקציונאליות פרוצדוראליות וגנריות.

C# היא שפה מעניינת, נוחה ומלאה פונקציונאליות למתכנת. שימוש בשפה זו נפוץ כיום, וכתוצאה מכך, ניתן היה למצוא בה קודים שונים שנדרשו לפיתוח.

ה - DataBase שנכתב בשפת MongoDB. למסדר נתונים של MongoDB יש כלים נרחבים לגיבוי כל המידע של המערכת, כולל מערכת ההפעלה, חשבונות המשתמשים והרשאותיהם, הגדרות ההתקנים, תוכניות וכן של שאר הרכיבים המסופקים עם השרת ואובייקטי המשתמש.

דוגמא לזרימת מידע במערכת

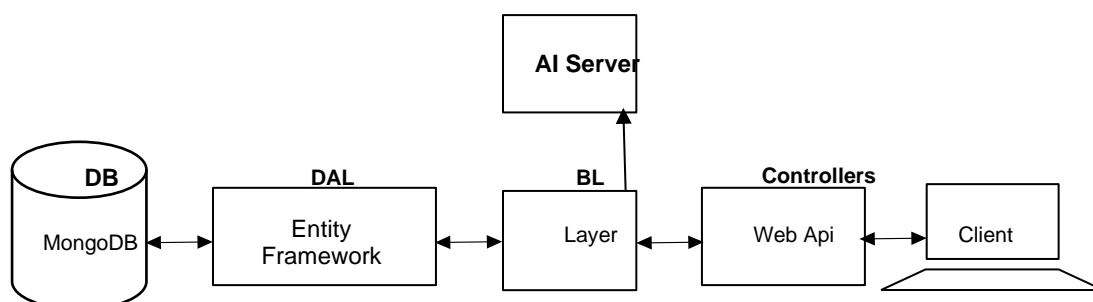
שליפת כל המשתמשים:

ברצוננו לקבל את כל המשתמשים הקיימים במערכת מה DB ולכן יתבצע השלבים הנ"ל:

- בכניסה בתור מנהל תהיה אפשרות לראות את כל המשתמשים הקיימים במערכת. הוא ילחץ על כפתור הצג משתמשים בתצוגה (ב-html) ובקשתו תפנה ל-TypeScript.
- תתבצע קריאה לפונקציה ShowUsers ב-TypeScript אשר תפנה לשרת url ותתבצע בקשת services.
- השרת מקבל את הבקשה ומנווט ל Controller שנמצא ב-API.
- ה Controller יזמן את הפונקציה GetUsers שנמצאת ב- UserManager ב-BL הוא מעוניין לקבל נתונים מה DB ולכן הוא פונה ל-DAL דרך framework Entity.
- ה-DAL שואב את הנתונים הרצויים ממסד הנתונים וכעת מתבצע שלב החזרה.
- ה-DAL מחזיר את רשימת המשתמשים לשכבת ה-BL בה מתבצעת פונקציית הסיון של הבאת המשתמשים הקיימים במערכת.
- הפונקצייה GetUsers מחזירה את הנתונים מה-controller ל-BL.

- הנתונים מוחזרים ל- controller מה- services.
- מה service חוזרת הרשימה ל- TypeScript.
- הרשימה מוצגת ב-HTML.

איור:



1. מסד הנתונים הבנוי ממסמכים מבנה המסמכים עובד מעל מימוש של JSON הנקרא על ידי MongoDB – BSON

2. שכבת הישיות.

3. שכבת ה- BL בה פונקציות המנהלות את המידע ב- Data base.

4. Web Api פרוטוקול התקשורת בין צד הלקוח וצד השרת.

5. TypeScript, angular - צד לקוח.

11.3. ארכיטקטורת רשת

לא רלוונטי

11.4. תיאור פרוטוקולי התקשורת

http – ראשי תיבות של המילים - Hypertext Transfer Protocol
 הוא פרוטוקול תקשורת שנועד להעברת דפי HTML ואובייקטים ברשת
 האינטרנט וברשתות האינטראנט. הפרוטוקול פועל בשכבת היישום של מודל
 ה - OSI ובשכבת היישום של מודל TCP/IP.

11.5. שרת – לקוח

צד השרת נכתב בטכנולוגיית WebApi ובשפת C#.

צד הלקוח נכתב בשפות:

- Angular
- HTML
- CSS
- TypeScript

11.6. תיאור הצפנות

לא רלוונטי

12. ניתוח ותרשים use case של המערכת

המוצעת

12.1. רשימת use case

:User

- הרשמה
- התחברות
- מילוי טופס יצירת קשר
- עדכון טופס יצירת קשר
- העלאת תמונה

:Seeker

- העלאת תמונת אדם הנחשד כנאבד
- צפייה בטופס יצירת קשר – במידה וקיימת התאמה.

12.2. תיאור use case

UC1

- **UC1 :Identifier**
- **Name:** הרשמה לאפליקציה.
- **Description:** משתמש מזין את פרטיו האישיים.
- **Actors:** משתמש שלא קיים במערכת.
- **Frequency:** בכניסה לאפליקציה.
- **Pre - Condition:** לא קיים במערכת.
- **Post – Condition:** נתוני המשתמש נקלטים במערכת. המערכת מוסיפה אותם למאגר (ב – Data base).
- **Extended use cases:** בהרשמה לאפליקציה.
- **Assumptions:** הנתונים שהוזנו נכונים ותקינים.
- **Basic course of action:** המערכת שומרת את פרטי המשתמש. כעת המשתמש יכול להיכנס לאזור האישי שלו.
- **Altemate course of action:** אם מספר הזהות כבר קיים במערכת תוצג שגיאה וכן אם המשתמש לא מילא את כל השדות.
- **Change history:** גרסה ראשונה, תהילה אברהמי.
- **Issues:** הכנסת פרטי משתמש חדש
- **Decisions:** אנו מסתמכים על נכונות הפרטים.

UC2

- **UC2 :Identifier**
- **Name:** התחברות לאפליקציה.
- **Description:** משתמש מכניס מספר זהות וסיסמה.
- **Actors:** משתמש הקיים במערכת.
- **Frequency:** בכניסה לאפליקציה.
- **Pre - Condition:** פרטי המשתמש קיימים במערכת.
- **Post – Condition:** הפרטים נבדקים במידה ואכן קיימים במערכת מתבצעת התחברות – ניתן לגשת לאזור האישי.
- **Extended use cases:** בהתחברות לאפליקציה.
- **Basic course of action:** המערכת קולטת את פרטי המשתמש. על מנת להתחבר המשתמש חייב להיות קיים במערכת.

- **Altemate course of action**: אם המשתמש לא זוהה תוצג הודעה "משתמש לא קיים" ואפשרות לנסות שוב.
- **Change history**: גרסה ראשונה, תהילה אברהמי.
- **Issues**:
 - הכנסת מספר זהות.
 - הכנסת סיסמה.
- **Decisions**: אנו מסתמכים על נכונות הפרטים.

UC3

- **UC3 :Identifier**
- **Name**: מילוי טופס יצירת קשר והעלאת תמונה.
- **Description**: משתמש מזין פרטים ליצירת קשר במקרה של היעדרות וכן מעלה תמונת פניו.
- **Actors**: משתמש.
- **Frequency**: כשמשתמש מחובר למערכת.
- **Pre - Condition**: התחברות/ הרשמה.
- **Post – Condition**: טופס יצירת קשר נקלט במערכת, המערכת מוסיפה אותם למאגר (ב – Data base). ישנה אפשרות של עדכון טופס.
- **Assumptions**: הנתונים שהוזנו תקינים, התמונה הועלתה בהצלחה.
- **Basic course of action**: המערכת שומרת את פרטי הטופס והתמונה. כעת הטופס ניתן לעדכון.
- **Altemate course of action**: אם התמונה לא הועלתה כמו שצריך תוצג הודעה מתאימה.
- **Change history**: גרסה ראשונה, תהילה אברהמי.
- **Issues**: מילוי טופס יצירת קשה והעלאת תמונה.
- **Decisions**: אנו מסתמכים על נכונות הפרטים.

UC4

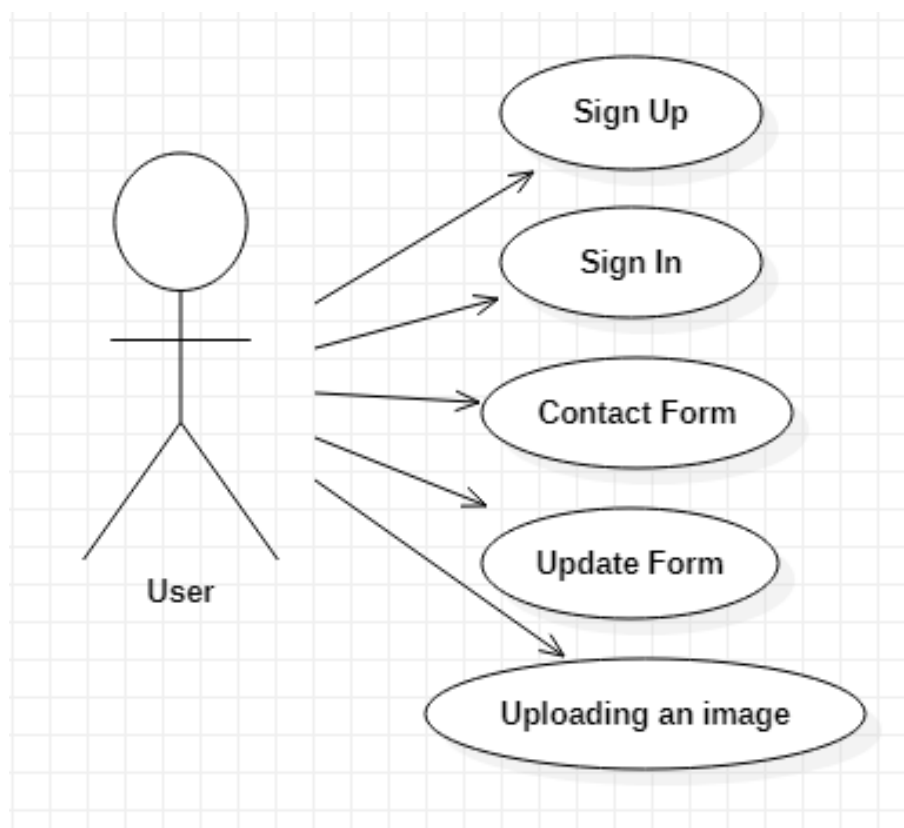
- **UC4 :Identifier**

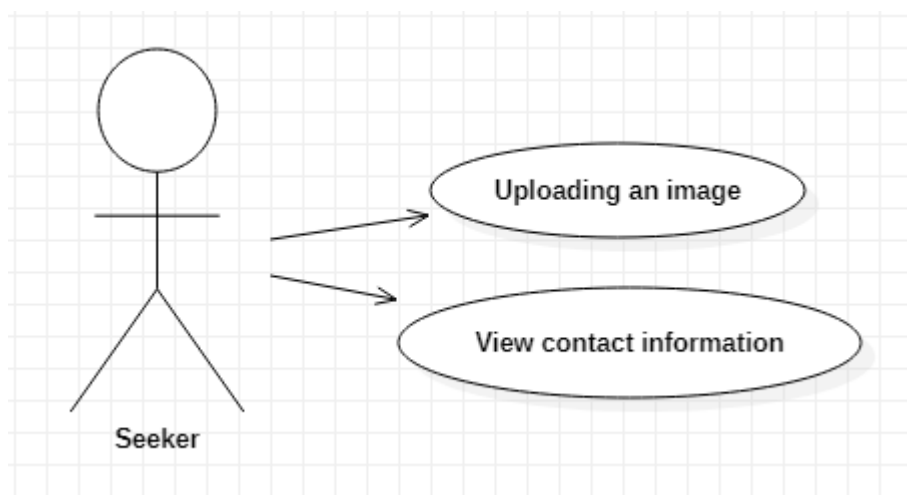
- **Name:** העלאת תמונת אדם הנחשד כנאבד.
- **Description:** משתמש מעלה תמונה של אדם שמצא.
- **Actors:** משתמש אנונימי.
- **Frequency:** בכניסה לאפליקציה.
- **Pre - Condition:** אין צורך בהרשמה/ התחברות
- **Post – Condition:** התמונה נקלטת במערכת ועוברת לעיבוד וניסיון התאמה.
- **Assumptions:** התמונה הועלתה בהצלחה.
- **Basic course of action:** המערכת קולטת את התמונה. מתבצע ניסיון התאמה.
- **Altemate course of action:** אם התמונה לא הועלתה בהצלחה תוצג הודעת שגיאה ואפשרות להעלות שוב.
- **Change history:** גרסה ראשונה, תהילה אברהמי.
- **Issues:** העלאת תמונה.

UC5

- **UC5 :Identifier**
- **Name:** צפיה בטופס יצירת קשר.
- **Description:** משתמש מחכה להתאמה. לאחר ההתאמה יוחזר טופס יצירת קשר.
- **Actors:** משתמש אנונימי.
- **Frequency:** לאחר העלאת תמונה.
- **Pre - Condition:** האדם המצולם זוהה במערכת.
- **Post – Condition:** המשתמש יוצר קשר עם המשפחה באמצעות הפרטים שקיבל.
- **Assumptions:** האדם המצולם זוהה במערכת – נמצאה התאמה.
- **Altemate course of action:** אם לא נמצאה התאמה תוצג הודעה על כך.
- **Change history:** גרסה ראשונה, תהילה אברהמי.

Use case Diagram



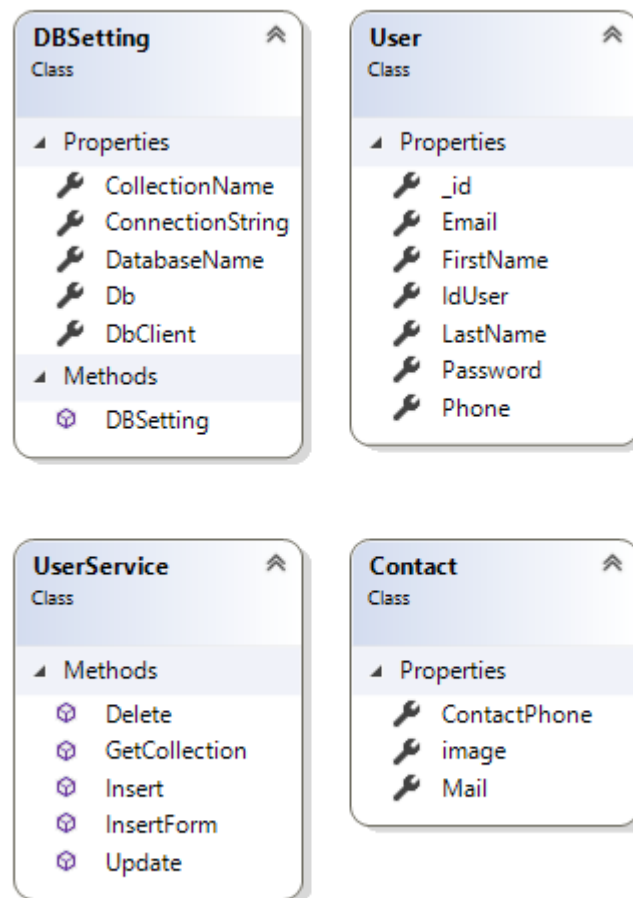


12.3. מבני נתונים בהם משתמשים בפרויקט

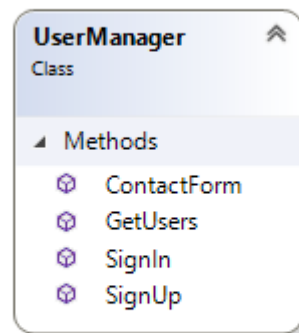
- List - במהלך הפרויקט השתמשתי במבנה הנתונים List המייצג רשימה. השימוש בו היה לשליפת משתמשים ושמירתם בתוך רשימה.
- וקטור – משמש את המודל ללמוד מתוך נתונים שאעמיד לרשותו כאשר הלמידה המשמעותית והנכונה תהיה באמצעות Data Set מקביל שימש כ- tester כדי לבדוק את נכונות הלמידה.
- מערך - מבני נוסף ששימוש אותי הוא מערך. המערך שימש לצורך הכנסת התמונות המעובדות לתוך מערך לשם האימון.

12.4. תרשים מחלקות

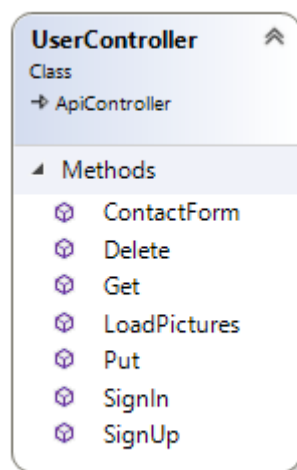
שכבת ה-DAL:



שכבת ה-BL:



שכבת ה-API:



12.5. תיאור המחלקות

הפרויקט מחולק ל- 3 שכבות. כל שכבה אחראית על תחום מסוים בפרויקט.

DAL – השכבה הנמוכה ביותר. שכבה זו אחראית על התקשורת עם Data Base. בשכבה זו פונקציות שונות המפעילות את ההתקשרות.

המחלקות הקיימות בשכבה:

- **DBSetting** – מחלקה זו משמשת להתקשרות בפועל ל – Data Base. המחלקה מכילה את הניתוב ל – Data Base, שם ה – Data Base, ושם ה – Collection.

- **User** – מחלקה זו מייצגת משתמש במערכת. מכילה את תכונות המשתמש: שם פרטי, שם משפחה, מספר זהות, מייל, פלאפון וסיסמה – כולם מסוג String.

- **Contact** – במחלקה קיימות תכונות המשמשות כפרטים ליצירת קשר והם: מייל, פלאפון והעלאת תמונה לשמירה במאגר.

- **UserService** – במחלקה זו קיימות פונקציות האחראיות על פעולות המשתמש:

GetCollection() – החזרת רשימת כל המשתמשים.

Insert() – הוספת משתמש חדש למערכת.

Update() – עדכון נתוני משתמש.

Delete() – הסרת משתמש מהמערכת.

IsertForm() – הוספת טופס יצירת קשר למשתמש.

BL – השכבה שמעל ה – DAL היא מקשרת בין ה – DAL ל – API. בשכבה קיימות פונקציות שתפקידם היא ניהול המידע ב – Data base.

המחלקות הקיימות בשכבה:

- **userManager** – במחלקה קיימות פונקציות שמשמשות לניהול המשתמשים במערכת:

GetUsers() – הפונקציה מחזירה משתמש מסוג List.

SignUp() – פונקציית הרשמה למערכת.

SignIn() – פונקציית התחברות למערכת. הפונקציה מוודאת שהמספר זהות והסיסמה אכן תקינים.
 ContactForm() – פונקציית הוספת פרטי יצירת קשר.

API – שכבה זו אחראית על החיבור בין צד השרת והלקוח. בשכבה זו קיימים קבצי מערכת רבים, קבצי התקנות, סקריפטים וכו'. בנוסף בשכבה זו קיימים Controlers – בקרים האחראים על ניתוב התקשורת בין השרת והלקוח:

- **UserController** – במחלקה זו קיימות הפעולות: GET, POST, הרשמה, התחברות, הוספה, מחיקה, הוספת פרטי יצירת קשר, עדכון פרטים ו - LoadPictures() – מטרתה לטעון תמונה המתקבלת מהלקוח.

13. תיאור התוכנה

• סביבת עבודה:

- Visual Studio ○
- Visual Studio Code ○
- MongoDB compass ○
- Jupyter notebook ○

• שפות תכנות:

צד השרת נכתב בטכנולוגיית WebApi ובשפת C#.

שרת למידה עמוקה נכתב בשפת Python.

צד הלקוח נכתב בשפות: HTML, CSS, TypeScript בטכנולוגיית Angular.

14. אלגוריתמים מרכזיים

הפעולות העיקריות בפרויקט:

14.1 הגדרת הבעיה

הבעיה הנתונה הינה שליפת נתון ממסד הנתונים כאשר המזהה היחיד העומד לרשותי הוא מזהה ויזואלי – תמונת פניו של הנמצא. דרך תמונה זאת ניתן להגיע למטרה – רשומה ייעודית ב-Data base.

היות והמזהה היחיד העומד לרשותי הוא מזהה ויזואלי עלי להבטיח את מירב האיכות והדיוק בזיהוי. לאחר ביצוע פעולת החיזוי ושליפת הנתונים ננסה להעמיד להעמיד לרשותי מאמתיים נוספים שיהוו עבורי מבדק אמינות בזמן אמת.

14.2 מציאת פתרון

מודל היודע לבצע פעולת Image Similarity – ביצוע פעולת השוואה ויזואלית על וקטורים.

14.3 תהליך היישום

בתהליך היישום אספתי נתוני הדרכה האיכות ובכמות גבוהה. נתוני ההדרכה מחולקים לתמונת היעד ומכלול תמונות של בן אדם מסוים.

14.4 יצירת Data Set

- איסוף – איסוף (כרייה/ ליקוט מתוך מאגר קיים) נתוני אימון.
- ארגון ותוויות – ארגון נתונים במבנה תיקיות עם שמות (תוויות הנתונים).
- עיבוד מקדים – ביצוע פעולות טרנספורמציה (עיבוד מקדים) לנתונים:
 1. שינוי גודל תמונה לגודל אחיד שהמודל ידע לקלוט.
 2. שינוי עומק תמונה.
- בניה – וקטור הנתונים – בווקטור קיימים כל נתוני העיבוד – אורך ה-List, אורך תמונה, רוחב ועומק.

14.5 בניה ואימון המודל

בנית המודל נעשית באמצעות הפונקציה Create.

פונקציית האימון בנויה מ – input, process, output.

במהלך האימון טוענים את ה – Data.

מכריזים על המודל כמודל סדרתי – מזרימים לו נתונים אחד אחרי השני. הזרמת המידע באמצעות input ו – output המיוצגים כ – x ו – y דרכם המודל מבצע את תהליך הלמידה.

המודל בנוי משכבות קונבולוציוניות – אלו שכבות הלומדות את התמונה ומבצעות ייצוג של התמונה במרחב הוקטורי. תהליך זה מבוצע באמצעות כעין פילטרים המייצגים צורות ויזואליות קטנות ושונות ומיקומם בתמונה הנלמדת.

השכבות הבאות הם שכבות נסתרות. תפקידם לדעת להפיק את התחזית מהייצוג שנלמד בשלב הקודם.

Activation – שכבת ההפעלה של תהליך הלמידה.

לאחר ביצוע פעולת הלמידה שומרים את נתוני הלמידה שישמשו לשלב החיזוי.

14.6 חיזוי

בשלב החיזוי נכנס קלט (x בלי y). המודל חוזה מהו הפלט (y) כתוצאה מהלמידה.

14.6 Deploy – פריסה על השרת

בניית אפליקציית שרת שעליו יושב המודל.

השרת יקבל request – פעולת חיזוי ויחזיר תוצאה בחזרה.

15. קוד האלגוריתם

המערכת מאתחלת את נתוני הלמידה: שמירת הקטגוריות ב-List, קביעת גודל תמונה קבוע – 32.

לאחר מכן מעבר בלולאה על כל קטגוריות ה-Data Set. עוברים על כל תמונה בקטגוריה.

הפונקציה create – יצירת נתוני האימון - עריכת עיבוד מקדים לתמונות הנקרא pre processing. כמו כן יצירת נתוני האימון - עריכת עיבוד מקדים לתמונות הנקרא pre processing. כמו כן הכנסת התמונות המעובדות לתוך מערך לצורך האימון.

```

1 import numpy as np
2 import os
3 from matplotlib import pyplot as plt
4 import cv2
5 import random
6 import pickle
7
8
9 class_list = []
10
11 DATADIR = "./dataset"
12
13 CATEGORIES = ['Anthony_Mackie', 'Adriana_Lima', 'Alex_Lawther', 'Alexandra_Daddario', 'Alvaro_Morte', 'alycia_dabn',
14              'Amanda_Crew', 'amber_heard', 'Andy_Samberg', 'Anne_Hathaway']
15
16 # The size of the images that your neural network will use
17 IMG_SIZE = 32
18
19 # Checking or all images in the data folder
20 for category in CATEGORIES :
21     path = os.path.join(DATADIR, category)
22     for img in os.listdir(path):
23         img_array = cv2.imread(os.path.join(path, img), 1)
24
25 training_data = []
26
27 def create_training_data():
28     for category in CATEGORIES :
29         path = os.path.join(DATADIR, category)
30         class_num = CATEGORIES.index(category)
31         for img in os.listdir(path):
32             img_array = cv2.imread(os.path.join(path, img))
33             image = cv2.resize(img_array, (IMG_SIZE, IMG_SIZE)).flatten()
34             training_data.append([image, class_num])
35
36
37 create_training_data()

```

המודל מכין את נתוני האימון בפלט ובקלט (y, x).

המודל אמור ללמוד ולבחון את ההתאמה. באמצעות בחינת הפיצ'רים ידע בעתיד לחזות נתונים שלא למד עליהם.

```
42 X = [] #features
43 y = [] #labels
44
45 for features, label in training_data:
46     X.append(features)
47     y.append(label)
48
49 X = np.array(X).reshape(-1, IMG_SIZE, IMG_SIZE, 3)
```

המודל מציג מיקום של ה-Data ב- x ו- y ומכין את ה-Data במבנה המתאים בוקטורים - Reshape.

```
51 # Creating the files containing all the information about your model
52 pickle_out = open("X.pickle", "wb")
53 pickle.dump(X, pickle_out)
54 pickle_out.close()
55
56 pickle_out = open("y.pickle", "wb")
57 pickle.dump(y, pickle_out)
58 pickle_out.close()
59
60 pickle_in = open("X.pickle", "rb")
```

ייבוא ספריות למידת מכונה למודל.

הגדרת משתנים X ו- Y (input, output).

בניית אובייקט מודל.

```

1 import tensorflow as tf
2 from tensorflow.keras.models import Sequential
3 from tensorflow.keras.layers import Dense, Dropout, Activation, Flatten, Conv2D, MaxPooling2D
4 import pickle
5 from keras.models import model_from_json
6 from keras.models import load_model
7 import matplotlib.pyplot as plt
8
9 # Opening the files about data
10 X = pickle.load(open("X.pickle", "rb"))
11 y = pickle.load(open("y.pickle", "rb"))
12

```

הגדרת שלושת שכבות המודל (קונבולוציה).

הגדרת 2 שכבות חבויות האחראיות על פעולת המודל.

```

13 y = np.array(y)
14
15 # normalizing data (a pixel goes from 0 to 255)
16 X = X/255.0
17
18
19 # Building the model
20 model = Sequential()
21
22 # 3 convolutional layers
23 model.add(Conv2D(32, (3, 3), input_shape = X.shape[1:]))
24 model.add(Activation("relu"))
25 model.add(MaxPooling2D(pool_size=(2,2)))
26
27 model.add(Conv2D(64, (3, 3)))
28 model.add(Activation("relu"))
29 model.add(MaxPooling2D(pool_size=(2,2)))
30
31 model.add(Conv2D(64, (3, 3)))
32 model.add(Activation("relu"))
33 model.add(MaxPooling2D(pool_size=(2,2)))
34 model.add(Dropout(0.25))
35
36 # 2 hidden layers
37 model.add(Flatten())
38 model.add(Dense(128))
39 model.add(Activation("relu"))

```


נעשית פעולת קימפול על המודל.

Model_fit – פעולת אימון המודל.

```

41 model.add(Dense(11))
42 model.add(Activation("relu"))
43
44 # The output layer with 11 neurons for 11 classes
45 model.add(Dense(11))
46 model.add(Activation("softmax"))
47
48 # Compiling the model using some basic parameters
49 model.compile(loss="sparse_categorical_crossentropy",
50               optimizer="adam",
51               metrics=["accuracy"])
52
53 # Training the model, with 40 iterations
54 # validation_split corresponds to the percentage of images used for the validation phase compared to all the ima
55 history = model.fit(X, y, batch_size=32, epochs=40, validation_split=0.1)
56
57 # Saving the model
58 model_json = model.to_json()
59 with open("model.json", "w") as json_file :
60     json_file.write(model_json)
61
62 model.save_weights("model.h5")
63 print("Saved model to disk")
64
65 model.save('CNN.model')
66
67 # Printing a graph showing the accuracy changes during the training phase
68 print(history.history.keys())
69 plt.figure(1)
70 plt.plot(history.history['accuracy'])
71 plt.plot(history.history['val_accuracy'])
72 plt.title('model accuracy')
73 plt.ylabel('accuracy')
74 plt.xlabel('epoch')
75 plt.legend(['train', 'validation'], loc='upper left')

```

16. תיאור מסד הנתונים.

16.1. פירוט הטבלאות ב- Data Base

Email String	Password String	sub_contact Object	password String
"tehila@gmail.com"	"1234"	{ } 2 fields	No field
"tmoalem@gmail.com"	"1212"	{ } 2 fields	No field
"sb@gmail.com"	"4444"	{ } 2 fields	No field
"tuito@gmail.com"	No field	{ } 2 fields	"9090"
"rinat@gmail.com"	"5555"	{ } 2 fields	No field
"t09@gmail.com"	"4321"	{ } 2 fields	No field
"shulamit2182@gmail.com"	"123"	No field	No field
"shulamit2182@gmail.com"	"111"	No field	No field
"sara.m.b@gmail.com"	"smb"	No field	No field
"mk0527660151@gmail.com"	"mk1997"	No field	No field
"chani2484@gmail.com"	"chani2484"	No field	No field
"tamar.b@gmail.com"	"tlch"	No field	No field
"st.b@gmail.com"	"stb"	No field	No field

_id Mixed	FirstName String	LastName String	IdUser Mixed	Phone String
1 61f8f95518204ed2a3cfed3	"Tehila"	"Avrahami"	"213289754"	"0548526325"
2 61f8fb5218204ed2a3cfed5	"Tchelet"	"Moalem"	"212558547"	"0556985858"
3 61f8fc0618204ed2a3cfed6	"Shira"	"Bura"	"213285467"	"0542874691"
4 61f8fc5218204ed2a3cfed7	"Tehila"	"Tuito"	"215427495"	"0547148544"
5 61f8fca118204ed2a3cfed8	"Rinat"	"Avital"	"215475284"	"0547481240"
6 61f8fce618204ed2a3cfed9	"Tahel"	"Levi"	"214685247"	"0491274510"
7 "Tehila"	No field	"Avrahami"	null	"*972548526325"
8 "sara"	No field	"Avrahami"	null	"*972548526325"
9 "Sara Miriam"	No field	"Barzilay"	"628cd95e713174d2338866de"	"058392661"
10 "Michal"	No field	"Karawani"	"628cdb08713174d2338866df"	"0527660151"
11 "Chani"	No field	"Washdi"	"212066512"	"0548452484"
12 "212048795"	"Tamar"	"Barzilay"	No field	"0583292661"
13 null	"Esty"	"Barzilay"	"330145684"	"058329661"

User



Contact form



Image



```

_id: ObjectId("61f8f95518204ed2a3cfefd3")
FirstName: "Tehila"
LastName: "Avrahami"
IdUser: "213289754"
Phone: "0548526325"
Email: "tehila@gmail.com"
Password: "1234"
> sub_contact: Object

```

```

_id: ObjectId("61f8fb5218204ed2a3cfefd5")
FirstName: "Tchelet"
LastName: "Moalem"
IdUser: "212558547"
Phone: "0556985858"
Email: "tmoalem@gmail.com"
Password: "1212"
> sub_contact: Object

```

```

_id: ObjectId("61f8fc0618204ed2a3cfefd6")
FirstName: "Shira"
LastName: "Bura"
IdUser: "213285467"
Phone: "0542874691"
Email: "sb@gmail.com"
Password: "4444"
> sub_contact: Object

```

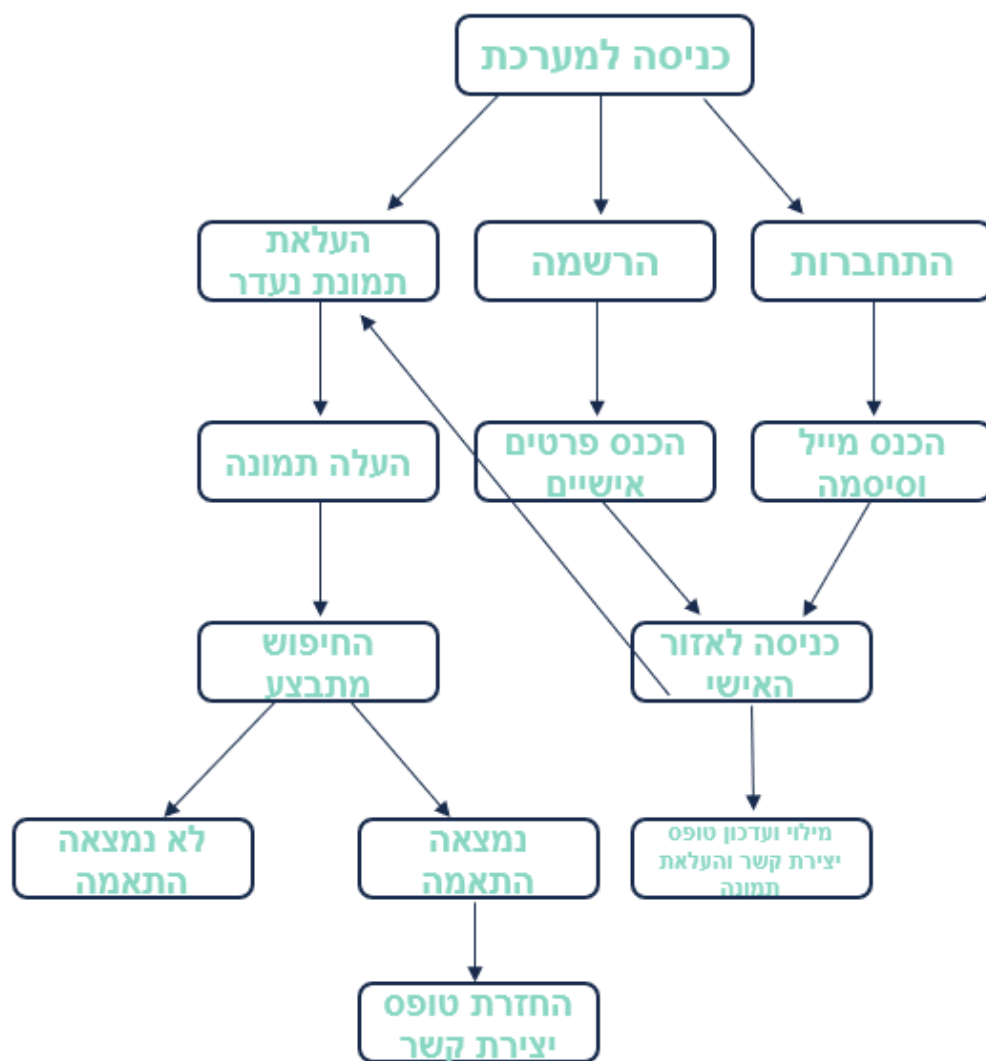
```

_id: ObjectId("61f8fc5218204ed2a3cfefd7")
FirstName: "Tehila"
LastName: "Tuito"
IdUser: "215427495"
Phone: "0547148544"
Email: "tuito@gmail.com"
password: "9090"
> sub_contact: Object

```

17. מדריך למשתמש

17.1. תיאור המסכים



17.2. מדריך למשתמש

שלום לך משתמש יקר.

אנו שמחים להכיר בפניך את Relatiive – האפליקציה שתשנה את פני האנושות.

תחילה, בכניסתך למערכת עליך לוודא שהינך מחובר. אם אתה משתמש חדש עליך להירשם למערכת לחץ על Sign Up – ומלא את פרטיך האישיים שם, משפחה, מספר זהות וכו'.

אם הינך משתמש קיים עליך להתחבר למערכת באמצעות Sign In – הזן מייל וסיסמה והתחבר.

כעת הינך יכול למלא טופס יצירת קשר למקרה של היעדרות חלילה תהיה אפשרות ליצור קשר עם קרוב משפחה. מלא מייל ופלאפון ליצירת קשר והעלה תמונת פנים שלך. על התמונה להיות עדכנית וברורה. בכל שלב תוכל לעדכן את הפרטים והתמונה. כעת הינך מוגן ע"י Relatiive ונוכל לזהות אותך במקרה היעדרות.

במקרה שנמצא אדם מבוגר משוטט לבדו ונראה במצב לא תקין או ילד קטן תוכל לנסות לאתר את קרובי משפחתו של הנמצא ע"י העלאת תמונת הנמצא באפשרות של Upload a missing image לאחר העלאת התמונה המערכת תחפש התאמה ותחזיר טופס יצירת קשר במקרה שאכן נמצאה התאמה.

17.3. צילומי מסכים

Home – דף הבית



כ-250 בני אדם נעדרים בכל יום בישראל. בין אם אלו קשישים עם דמנציה, אנשים הסובלים מאוטזם נמוך ונחשבים בתפקוד נמוך או ילדים שהלכו לאיבוד. אז כן, הרוב המוחלט חוזר הביתה בסופו של דבר - אבל לא כולם. Relatiive רוצה לקצר את הזמנים שבהם נמצאים אותם אנשים, לרוב חסרי ישע בעזרת שימוש במכשיר הפאלפון.

[קרא עוד](#)

About – אודות



כ-250 בני אדם נעדרים בכל יום בישראל. בין אם אלו קשישים עם דמנציה, אנשים הסובלים מאוטיזם נמוך ונחשבים בתפקוד נמוך או ילדים שהלכו לאיבוד. אז כן, הרוב המוחלט חוזר הביתה בסופו של דבר - אבל לא כולם. Relatiive רוצה לקצר את הזמנים שבהם נמצאים אותם אנשים, לרוב חסרי ישע בעזרת שימוש במכשיר הפאלפון.

מצלמים אדם או ילד שנראה שהלך לאיבוד והמערכת תעשה את השאר

Realatiive הינה אפליקציה שתאפשר לכם לצלם אדם הנראה אבוד או חסר ישע. התמונה עוברת עיבוד במערכת ואם קרובי משפחתו של הנעדר מילאו את פרטיו - היא תחבר ישירות בין המצלם לבין המשפחה.

Sign in – התחברות

Sign in to Relatiive

Email address

Password

Sign in

הרשמה – Sign Up

Join Relative

Create your account

First name

Last name

Identity

Email address

Phones

Password

Sign in

Contact us form – טופס יצירת קשר

Contact us form

Email address

Contact phone

No file chosen

✗ No image uploaded

לאחר העלאת תמונה

Contact us form

Email address

Contact phone

 IMG_8382.png

✔ Image uploaded successfully

Upload a missing image – העלאת תמונת נעדר

Please upload a missing image

Choose File logo2.jpg

✓ Image uploaded successfully

Search

18. בדיקות והערכה

לאחר אימון המודל לזיהוי פנים נבדקו ונבחנו תוצאות המודל. הזיהוי לא היה מקסימלי ולכן לצורך שיפור הזיהוי ערכתי שיפור מקדים של המודל בנוגע לעיבוד תמונה כמו גודל תמונה וכו'. לאחר מכן הזיהוי עבד בצורה טובה יותר ומיטבית. יתר על כן המודל אומן על מספר אנשים שונים ועל תמונות שונות ואכן המודל זיהה את האנשים השונים וידע לשייך תמונות שונות לאדם מסוים.

19. ניתוח יעילות

יעילות האלגוריתם הינה חשובה מאוד לביצוע. חשיבותה רבה משום שהאפליקציה עובדת בזמן אמת – בזמן מציאת נאבד וכאשר יאריך זמן רב בחיפוש ההתאמה ובזיהוי זה עלול להיות קריטי. לכן השתדלתי מאוד בכל התהליך שהאלגוריתם יהיה יעיל ובד בבד בעל סיבוכיות נמוכה ככל האפשר.

20. אבטחת מידע

הכניסה לחשבון האישי חייבת להתבצע באמצעות הזנת שם משתמש וסיסמה. כך משתמש יכול להיות רגוע ובטוח שהמידע אודותיו שמור במערכת, מוצפן וחסוי. באופן כללי כניסה למערכת לא מחייבת התחברות באמצעות סיסמה אך כל פעילות במערכת של העלאת תמונה או עדכון פרטים אישיים מחייבת התחברות או הרשמה.

21. מסקנות

הנה כעת לאחר עבודה רבה ומאומצת אני מגיעה כמעט לחתימת הספר בו כתבתי והרחבתי על Realatiive – כיצד נולד הרעיון לפרויקט זה, איך נרקם, איך התקדם והכל שלב אחר שלב.

בשלב ההתחלתי עוד בחשיבת על רעיון העליתי ופסלתי המון רעיונות חלקם טובים יותר וחלקם פחות חיפשתי רעיון שיהיה לי מעניין לבצע, שתהיה בו הנאה מעבר ללמידה העמוקה, חיפשתי להתנסות בדבר חדש שלא הכרתי, עניין אותי לחקור נושאים חדשים שמעולם לא נגעתי בהם.

לאחר שהחלטתי סופית על רעיון לפרויקט – זיהוי פנים והשוואה התחלתי לפרק את העניין מה עלי לעשות, מהיכן להתחיל, מה ללמוד ומה לחקור בשלב זה העבודה הייתה עדיין נחמדה. נהייתי מאוד לקרוא חומרים, להבין קצת איך עובדות הרבה אפליקציות הקשורות לזיהוי פנים זה היה ממש מרתק.

לאחר שצברתי מספיק ידע התחלתי לחשוב איך אני משתמשת בו, איך לפתח את האפליקציה בצורה נכונה ויעילה. בשלב זה העניינים התחילו להראות קשים יותר ורחוקים מביצוע. לא היה לי מושג איך להתחיל, באיזה צורה לעבוד ובמה להשתמש. כמות הנושאים החדשים שהיה עלי ללמוד הייתה רבה. מלכתחילה לא חשבתי שהעבודה תהיה כה קשה ואינטנסיבית. ניסיתי את מזלי בהרבה קודים, כתבתי המון ומחקתי עוד יותר. המצב היה מאוד מייאש, הרגשתי שאין לי סיכוי וכמעט חשבתי להרים ידיים, היו פעמים שישבתי שעות על גבי שעות ולא התקדמתי כמעט במאומה. המצב הזה הלחיץ אותי והרגשתי שאולי אין לי סיכוי ועלי לוותר. אך בסופו של דבר החלטתי שאסור לי ככה להתייאש עלי לקום ולנסות כיוון אחר, משהו שונה ואכן לאחר שהתמקדתי בניסיונות כושלים לכתוב אלגוריתם שיהיה מדויק ויעיל – הדבר לא עלה בידי ולכן עברתי לתחום של למידת מכונה שמטרתה המרכזית היא טיפול ממוחשב בנתונים מן העולם האמיתי עבור בעיה מסוימת, כאשר לא ניתן לכתוב

תוכנת מחשב עבודה. לצורך תהליך הלמידה הצרכתי לאמן את המערכת – להכניס למערכת דאטה סט ענק של פרצופים וכך המערכת לומדת היפותזה שמתארת בצורה הטובה ביותר את הדוגמאות שהיא ראתה. כל העניין הזה של למידה מכונה בכלל ואימון מודלים בפרט לא היה מוכר לי עד הפרויקט. מבחינתי הוא היה נושא מאוד רחוק וגדול ממני אך עכשיו לאחר סיום הפרויקט אני שמחה שהגעתי לתחום הזה, הוא הוסיף לי המון ידע וניסיון שלא הייתי מקבלת לולי הפרויקט.

דבר נוסף שהשגתי מעשיית הפרויקט הוא יכולת להבין קוד שונה שלא תמיד אני כתבתי ומכירה. במהלך הפרויקט יצא לנו לעזור הרבה אחת לשנייה בין אם זה בכתיבת האלגוריתם, בעיצוב וצד לקוח או בתכנון ובניה. הדבר מאוד תרם לנו לידע הכללי וגם לפרויקט האישי בפועל.

כעת לאחר סיום האפליקציה אני ממש גאה בעצמי ומאושרת שלא וויתרתי והמשכתי להתקדם ולנסות והנה יש לי ביד אפליקציה מוגמרת מתחילה ועד הסוף הדבר הוסיף לי המון ידע וניסיון בתחומים שלא הייתי מגיעה אליהם מעצמי.

הפרויקט מבחינתי היה הדבר שהכי קידם אותי מכל לימודי התכנות אני חושבת שלווותר על עשיית הפרויקט זה בלתי אפשרי, ההבדל בין המצב שבו היינו לפני הפרויקט ואחריו הינו הבדל משמעותי של שמים וארץ. התנסינו המון וצברנו ידע וכולי תקווה שאכן Realatiive תראה אור, תשמש אנשים ותציל חיים.

22. פיתוח עתידי

פיתוח הפרויקט היה ארוך משחשבתי. בתחילה לא לקחתי בחשבון המון עבודה שלכאורה הייתה נראית יחסית קלה וקצרה אך ארכה לי זמן וכוח רב. השקעתי בפרויקט את כל כוחי ומרצי העדפתי שיתבצע בצורה מקצועית וטובה ואפילו אם לא אספיק לבצע את כל רצונותיי ותוכניותי מסיבה זו הרבה דברים שתכננתי לעשות בפרויקט לא התבצעו בסופו של דבר מפאת חוסר הזמן והיכולת.

בהמשך הייתי מאוד רוצה לשכלל את הפרויקט ולהוסיף לו תוספים ושינויים שלא הספקתי לבצע כמו למשל לשכלל את המודל לזיהוי פנים שיהיה יותר מדויק וזיהוי הפנים יתבצע בצורה מדויקת יותר ומרבית. הייתי רוצה גם להשקיע גם בתחום של השוואת הפנים - לשכלל אותו יותר שההתאמה תהיה מלאה במאת האחוזים והמערכת תעבוד באופן כללי בצורה יותר מדויקת ומהירה.

דבר נוסף בשלב התשאול – כאשר משתמש ממלא את פרטיו האישיים בהרשמה אוסיף שדה שבו המשתמש יוסיף מזהים המייחדים אותו כגון: צבע שיער, מבנה גוף, גובה, וכו'. כך בזמן אמת כאשר תתבצע ההתאמה יוכל המוצא לבדוק את המזהים האם הם אכן תואמים לנמצא מולו במציאות. כך יתבצע אימות אנושי בנוסף למערכת.

בהמשך כאשר אשכלל את המערכת אבנה מודל שידע לפענח בעצמו מתוך תצלום של בן אדם את המזהים הייחודיים שלו. וזאת כדי לבצע רמת אמינות נוספת על התוצאה.

החלום שלי הוא להפוך את הפרויקט שהשקעתי בו כה רבות למציאות. שהוא באמת יוכל לשמש אנשים בצורה יומיומית ולהשיג את המטרה שלשמה הוא נועד. מסיבה זו אני זקוקה לפרויקט שעובד בצורה מלאה, בלי תקלות ושיבושים ולכן אני מוכנה ורוצה להשקיע בו גם לאחר ההגשה וסיום הפרויקט.

23. ביבליוגרפיה

- <https://github.com/>
- <https://stackoverflow.com/>
- <https://getbootstrap.com/>
- <https://internet-israel.com/>
- <https://zetcode.com/>
- MongoDB
- <https://www.w3schools.com/>
- <https://www.geeksforgeeks.org/>
- <https://he.wikipedia.org/wiki/>
- Medium