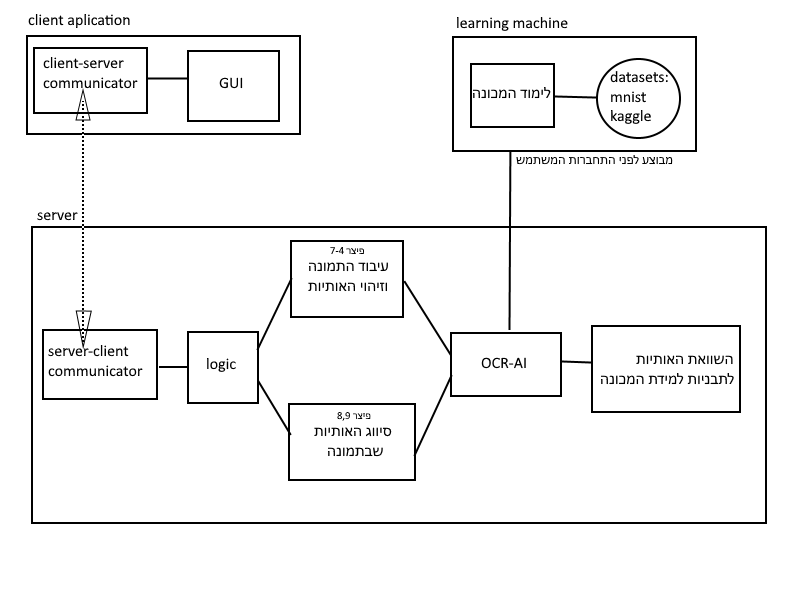
**ארכיטקטורת הפרויקט ImageToText:**

**מבט על:**



* **Client application:**
* רכיב האחראי על הצגת המידע למשתמש בצורה נוחה וקלה לשימוש.
* Client-server communicator – אחראי על התקשורת עם השרת.
* Application GUI – פלטפורמת האפליקציה שדרכה המשתמש ניגש למערכת. רכיב זה משמש למעשה כשכבת האפליקציה וייכתב בפייתון.
* **Server:**
* Server-client communicator – אחראי על ביצוע התקשורת מול הלקוח, כולל טיפול בתקלות תקשורת.
* רכיב זה ייכתב בpython.
* Logic רכיב לוגיקה – אחראי על עיבוד הנתונים והבקשות המתקבלים בתקשורת, אימות ושליפת המידע, והעברת הבקשות למענה המתאים.
* סיווג האותיות שבתמונה – אם הבקשה מתאימה לו, ישלח למערכת הבינה המלאכותית ויסווג את האות שבתמונה.
* עיבוד התמונה וזיהוי האותיות, אם הבקשה מתאימה לו, יעבד את התמונה ויערוך אותה בהתאם לבקשת הלקוח , ויזהה את האותיות בתמונה באמצעות הבינה המלאכותית.
* OCR AI- אחראי על זיהוי וסיווג האותיות בתמונה ועובדת עם learning machine ( ראה המשך learning machine).
* השוואת האותיות לתבניות למידת המכונה – לאחר שהמכונה לומדת באמצעות המאגר מידע ( ראה המשך learning machine).
* **Learning machine :**
* תהליך שמתבצע בהרצה הראשונית של השרת, שכביכול מלמדת את המכונה את התבניות של האותיות לפי ה MNIST וKaggle ופועל באופן ישיר עם מערכת הבינה המלאכותית OCRAI ולאחר מכן הוא נגמר.

**עיצוב הנתונים ויישויות המידע :**

* בקשות ומענות בין השרת ללקוח :
  + בקשת פתיחת התקשורת.
  + בקשת שליחת התמונה.
  + מענה על שליחת התמונה וסיווגה.
  + סיווג התקשורת.
* המידע המועבר בין GUI ללקוח הוא בהקשר לחיצת הכפתורים.
* בקשות ומענות בין הlogic ל OCR AI:
  + עיבוד התמונה ועריכתה בהתאם לבקשת הלקוח.
  + זיהוי האותיות מהתמונה וסיווגן.
* לימוד המכונה לבין המכונה עצמה באמצעות מאגרי המידע .

**טכנולוגיות עיקריות :**

* לצורך התקשורת בין השרת ללקוח נשתמש בsocket בשפת פייתון.
* לצורך עיבוד התמונה נשתמש בספריית opencv בשפת פייתון.
* לצורך לימוד המכונה נשתמש בtensorflow2 ובמאגרי המידע MNIST וKAGGLE.
* לצורך האפליקציה נשתמש בספריית Kivy Python Framework בשפת פייתון.

\*\* נשתמש בשפת פייתון מכיוון שאנחנו מאוד אוהבים אותה והיא מאוד נוחה ושימושית , הספריות בהן נשתמש נמצאות בה.

**התאמה לאפיון :**

|  |  |
| --- | --- |
| **פיצ'ר / תהליך:** | **רכיבים רלוונטיים** |
| קליטת התמונה מהמשתמש | לקוח (APP GUI) -> client-server communicator -> server-client communicator |
| עריכת התמונה שהתקבלה מהמשתמש | לקוח (APP GUI) -> client-server communicator -> server-client communicator |
| שליחת התמונה לשרת | לקוח (APP GUI) -> client-server communicator -> server-client communicator |
| דגימת התמונה כדי למצוא איזה גוון מבדיל בין הטקסט לרקע התמונה | server-client communicator-> Logic- > עיבוד התמונה -> ocr AI |
| (?) זיהוי שורות הטקסט ויישור השורה | server-client communicator-> Logic- > עיבוד התמונה -> ocr AI |
| (?) הפרדת השורות השלמות לאותיות ומילים על פי מרווחים | server-client communicator-> Logic- > עיבוד התמונה -> ocr AI |
| זיהוי האותיות בתמונה שהתקבלה | server-client communicator-> Logic- > זיהוי האותיות -> ocr AI |
| חיפוש בתוך מאגר המידע של האותיות | Learning machine-> ocr AI |
| סיווג האותיות בתמונה שהתקבלה. | communicator-> Logic- > סיווג האותיות -> ocr AI |
| (?) זיהוי מקום של האות בתוך התמונה | communicator-> Logic- > עיבוד התמונה -> ocr AI |
| קבלת לינק לקריאה בפונט קריא | Ocr AI-> logic - > server-client-> client-server – gui app |
| תרגום הטקסט שזוהה בתמונה | Ocr AI-> logic - > server-client-> client-server – gui app |
| הרצת קוד שזוהה בתמונה | Ocr AI-> logic - > server-client-> client-server – gui app |
| העברת קובץ הטקסט  ל PDF | Ocr AI-> logic - > server-client-> client-server – gui app |