תמונה שמכילה טקסט, צמח, עלה, שרך

התיאור נוצר באופן אוטומטי

**מסמך עיצוב**

תמונה שמכילה טקסט, אוסף תמונות

התיאור נוצר באופן אוטומטי

צ'אט בוט חכם ללקיחת הזמנה מלקוח

עבור



מסעדת תאי צ'ין – קרית טבעון

**מנחה:** מר אבי זכאי

**צוות הפרויקט:** שלומי אפרגן, ארטיום ברוברניק ויהונתן דובובי

**פתח דבר**

זהו מסמך אפיון מערכת BoThaiChin . בהמשך למסמך הייזום בו הסברנו את הרעיון והמטרות של הפרויקט ובאופן כללי איך תעבוד המערכת כאן אנו "נצלול" מעט עמוק יותר ונסביר על אופי וצורת המערכת בצורה יותר טכנית כך שגם אם בית תוכנה כלשהו יקבל את המסמך הוא ידע לייצר את המערכת על פי המידע שכתוב בו. בעזרת המסמך נסביר מי הם המשתמשים של המערכת, למען מי היא נבנית, נבין בעצמינו דברים על הפרויקט שעלינו ליישם ונקבל החלטות שילוו אותנו למשך כל זמן יצירת המערכת. החלטות שיתקבלו בעת כתיבת המסמך הזה יחרצו את גורל המערכת ולכן אנו נפעל כדי לקבל אותן בשיקול דעת ולאחר מחקר מעמיק.

**תקציר מנהלים**

תאי צ'ין היא מסעדה אסיאתית הפועלת בקרית טבעון כבר מעל ל-20 שנה, המסעדה מציעה את שירותיה במתכונת של ישיבה במקום, הזמנת משלוח ואיסוף עצמי מהמקום והזמנת משלוח עד הבית.

מידי יום מספק מערך ה-take away של המסעדה עשרות רבות ואף מאות הזמנות שמתקבלות על ידי הלקוחות המרוצים. המסעדה מנוהלת על ידי אור בן-מלך אשר הקים את המסעדה כשהיה חייל משוחרר בתחילת שנות ה-20 לחיו ומאז ועד היום המסעדה לא מפסיקה להתרחב ולשגשג. כמו מרבית המסעדות בישראל גם בתאי צ'ין מנהלים את ההזמנות בעזרת מערכת המידע Tabit ואין אנו רוצים להחליפה אלא לעבוד במקביל אליה ויחד איתה. כיום דאטה רבה שמתקבלת מהזמנות מהמסעדה אינה באה לידי ביטוי בייעול ומיקסום הכנסות ואנו חותרים לשים לכך סוף. המערכת אותה אנו מפתחים תיעל את זמן לקיחת ההזמנות מלקוחות (טלפוניים או כאלו שמסתבכים עם ההזמנה האינטרנטית), תצמצם את מספר העובדים במשמרת, תגדיל את כמות המנות שמזמינים לקוחות מידי יום ותשפר את חווית הלקוחות בעת ביצוע ההזמנה. ביצוע מטרות אלו על ידי המערכת צפויה להיות שווה למסעדה מאות אלפי שקלים מידי שנה בזכות החיסכון במספר העובדים מצד אחד והגידול בהזמנות מצד שני. אין ברצוננו לגבות מהלקוח את עלויות הפיתוח של המערכת משום שאנו מתכוונים להחזיק בבלעדיות עליה על מנת שנוכל להטמיע אותה אצל בתי עסק נוספים ולשרת מספר לקוחות במקביל ואולי גם למכור את המערכת ביום מן הימים. ולכן יהיה על הלקוח לשלם באופן חד פעמי כ-7500 ₪ בעבור הדרכה מלאה בת 50 שעות על המערכת, מידי שנה יהיה עליו לשלם כ-50,000 ₪ בעבור ליווי ותמיכה טכנית, במידה והלקוח יחליט בשלב מסוים כי הוא אינו מעוניין יותר בתמיכתנו אך כן מעוניין להמשיך להשתמש במערכת בכוחות עצמו יהיה עליו לשלם לנו סכום חד פעמי של כ-100,000 ₪ ומכאן במידה והלקוח יחליט לצרוך את שירות התמיכה שלנו ב-5 השנים הראשונות העלות הכוללת של הדרכה וקבלת שירותי התמיכה תסתכם ב-257,500 ₪ אך כבר מהשנה השנייה לשימוש הלקוח במערכת היא צפויה להיות רווחית עבור הלקוח והחזר ההשקעה (ROI) ל-5 שנים אלו עומד על 248% שהוא החזר חיובי ואף גבוהה למדי. חשוב לציין שהמערכת אינה דורשת ציוד טכנולוגי מתקדם במיוחד כדי לפעול ומספיק לה מחשב ביתי רגיל ואמין אשר קיימים כמה כאלו בבית העסק של הלקוח שלנו.

תוכן עניינים

[טכנולוגיות 5](#_Toc102939383)

[תיאור התהליכים במערכת 5](#_Toc102939384)

[שחקני המערכת 5](#_Toc102939385)

[Use-case 6](#_Toc102939386)

[תיאור מודל הנתונים 7](#_Toc102939387)

[תיאור מקורות איסוף הנתונים 7](#_Toc102939388)

[תיאור משתנה המטרה והגדרת הצלחה של המערכת 7](#_Toc102939389)

[מודל מסד הנתונים 8](#_Toc102939390)

[תיאור הפיצ'רים הצפויים 9](#_Toc102939391)

[דרכי התמודדות עם בעיות במדגם 11](#_Toc102939392)

[ממשקי משתמש 11](#_Toc102939393)

[עקרונות וסטנדרטים 11](#_Toc102939394)

[פירוט המסכים המתוכננים : 12](#_Toc102939395)

[מודל ניתוח הנתונים 15](#_Toc102939396)

[תוכנית ניתוח סטטיסטי 15](#_Toc102939397)

[פירוט הדוחות של תוצאות הניתוח הסטטיסטי 15](#_Toc102939398)

[לוח זמנים מפורט לביצוע הפרויקט 15](#_Toc102939399)

[נקודות פתוחות ומהדורות עתידיות של המערכת 17](#_Toc102939400)

טכנולוגיות

המערכת שאותה אנו מפתחים יושבת על התשתית של פלטפורמת שליחת המסרים "Telegram" אשר מותקנת כאפליקציה בטלפונים הסלולריים של המשתמשים או לעיתים גם בשולחן העבודה במחשב. קוד המערכת כתוב בשפת python והוא מאוחסן באתר git hub ומשם מתחבר לשרת הענן של פלטפורמת Heroku כמו כן הdatabase שאספנו וממשיך להתעדכן כל עוד המערכת עובדת יושב גם הוא בענן של Heroku ומנוהל בעזרת Heroku postgres.

כפי שציינו מקודם הקוד כתוב בשפת Python ובסביבת העבודה PyCharm ומאוחסן בחשב של המפתחים, המערכת אינה פועלת על בסיס השרת של המחשב שבו שמורים קבצי הקוד אלא בעזרת הקבצים ששמורים בgit hub ומחוברים לheroku- . דרך קבצי הקוד ששמורים במחשב של המפתחים ניתן לבצע שינוים בקוד ולעדכנם ב-git hub במסגרת הפעולות שהחלטנו לבצע נעשה שימוש בספריות Constants, telegram, telegram.ext, Responses, numpy, Pandas השימוש בספריות אלו מאפשר לנו להתחבר לתשתית של telegram ולתפעל את הבוט דרכם, לעבוד עם נתונים שמגיעים דרך database שמאוחסן ב-postgresql כאשר הקוד רץ במצב לוקאלי ו-heroku postgres כאשר הקוד רץ בעזרת שרת הענן ולהתנהל איתם בעזרת הקוד ב-python, לתקשר עם משתמשי המערכת בצורה שתהיה להם נוחה ומובנת. אנו בוחנים גם שילוב של קבצי json אשר בהם נשמור תבניות שיחה מסוימות לשיפור מתמיד של איכות המוצר.

את הנתונים אספנו וריכזנו לקבצי CSV אשר אותם ייבאנו אל פלטפורמת ה-SQL postgresql ומאוחר יותר אל Heroku postgres כדי שמערכת תעבוד ללא הפסקה על גבי שרת הענן ש-Heroku מציעה.

את התשתית לבוט שלנו אני מקבלים משירות ה- BotFather שנגיש לכל משתמשי טלגרם.

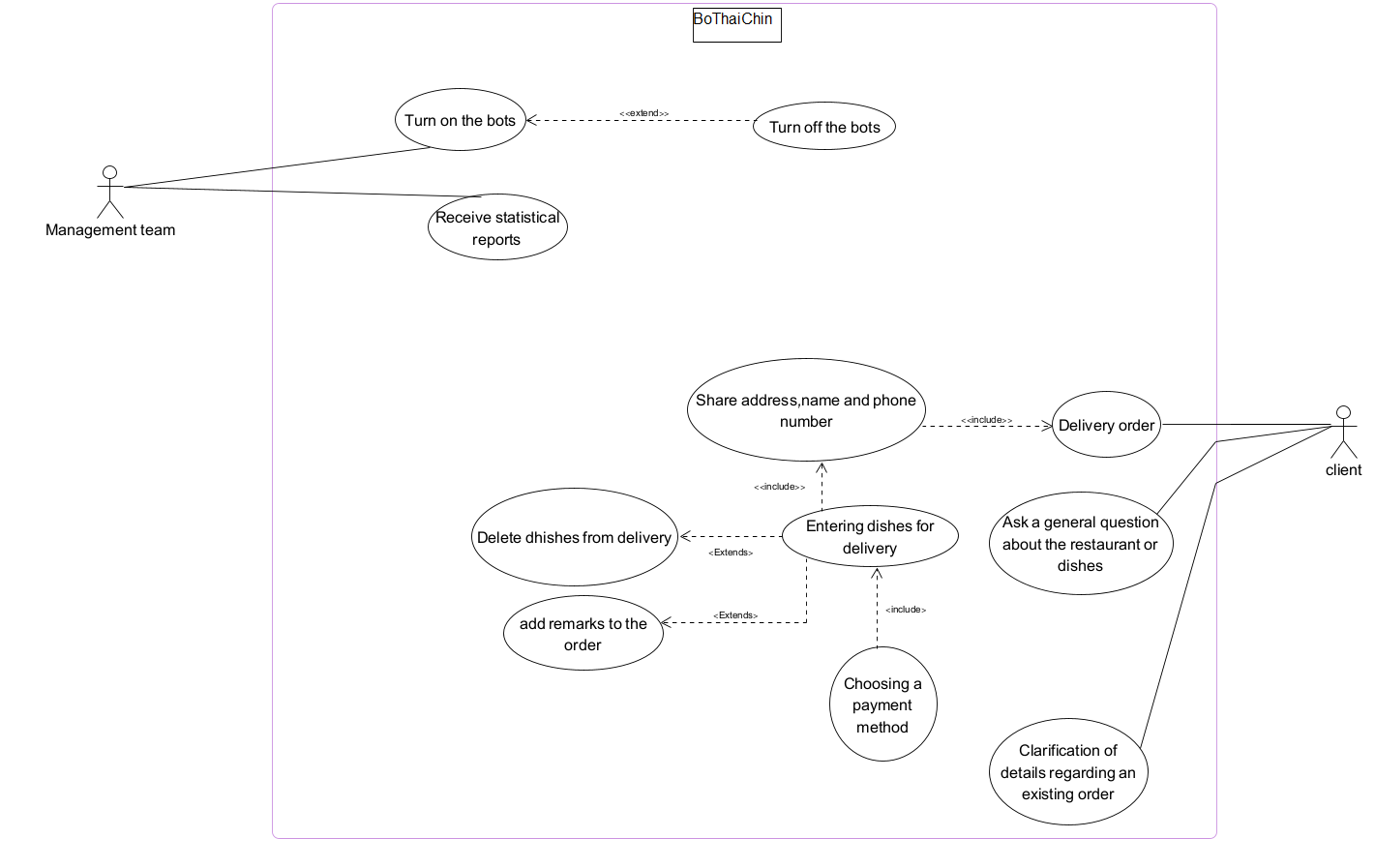
תיאור התהליכים במערכת

שחקני המערכת

מנהל המסעדה וצוות ההנהלה: המנהל והעובדים הבכירים שיקבלו ממנו את האישור יוכלו להזין סיסמא ולבצע פעולות נוספות. הם יוכלו לקבל אינדיקציה על ההזמנות והפעילות שמתבצעת דרך המערכת כגון סטטיסטיקות של הזמנות. להערכתנו מנהל המסעדה יתן את הגישה לשירות הנ"ל לעוד עובד או שניים בלבד ולכן כמות המשתמשים מסוג זה הוא 2-3.

לקוחות המסעדה: מבחינת היררכית הם הנמוכים ביותר בין שחקני המערכת אך תפקידם הוא החשוב ביותר והם באופן חד משמעי משתמשי המערכת העיקריים. הפעולות העיקריות שיוכלו לבצע באמצעות המערכת הן לבצע הזמנה של אוכל ולקבל מידע כלשהו אודות המסעדה. קשה לאמוד את כמות המשתמשים מסוג זה משום שחלקם גם ישתמשו במערכת באופן חד פעמי אבל נעריך שבשנה הראשונה מספרם יהיה בסביבות האלף משתמשים.

Use-case



|  |  |  |
| --- | --- | --- |
| שם פעולה | מספר סידורי | פירוט הפעולה |
| Enter to the management screen | 1 | המנהל מקיש קוד ייחודי ונכנס למסך ניהול |
| Receive statistical report | 1.2 | המנהל בוחר דוח רצוי |
| Turn on the bots | 2 | המנהל יכול להקיש על כפתור כיבוי בוט |
| Turn off the bots | 3 | המנהל יכול להקיש על כפתור הפעל בוט |
| Delivery order | 4 | הלקוח בוחר בכפתור הזמנת משלוח |
| Share phone number | 4.1 | הלקוח בוחר בשיתוף מספר טלפון |
| Share location | 4.2 | הלקוח בוחר בשיתוף מיקום |
| Entering dishes for delivery | 4.3 | הלקוח בוחר מנות שמובנות בבוט |
| Delete dishes from delivery | 4.4 | הלקוח יכול למחוק מנות שנבחרו |
| Add remarks to the order | 4.5 | הלקוח יכול להוסיף הערות להזמנה |
| Choosing payment method | 4.6 | הלקוח בוחר באפשרות התשלום הרצויה |
| Ask a general question | 5 | הלקוח יכול לשאול שאלות כלליות |
| Clarification of details order | 6 | הלקוח יכול לראות סטטוס הזמנה קיימת |

תיאור מודל הנתונים

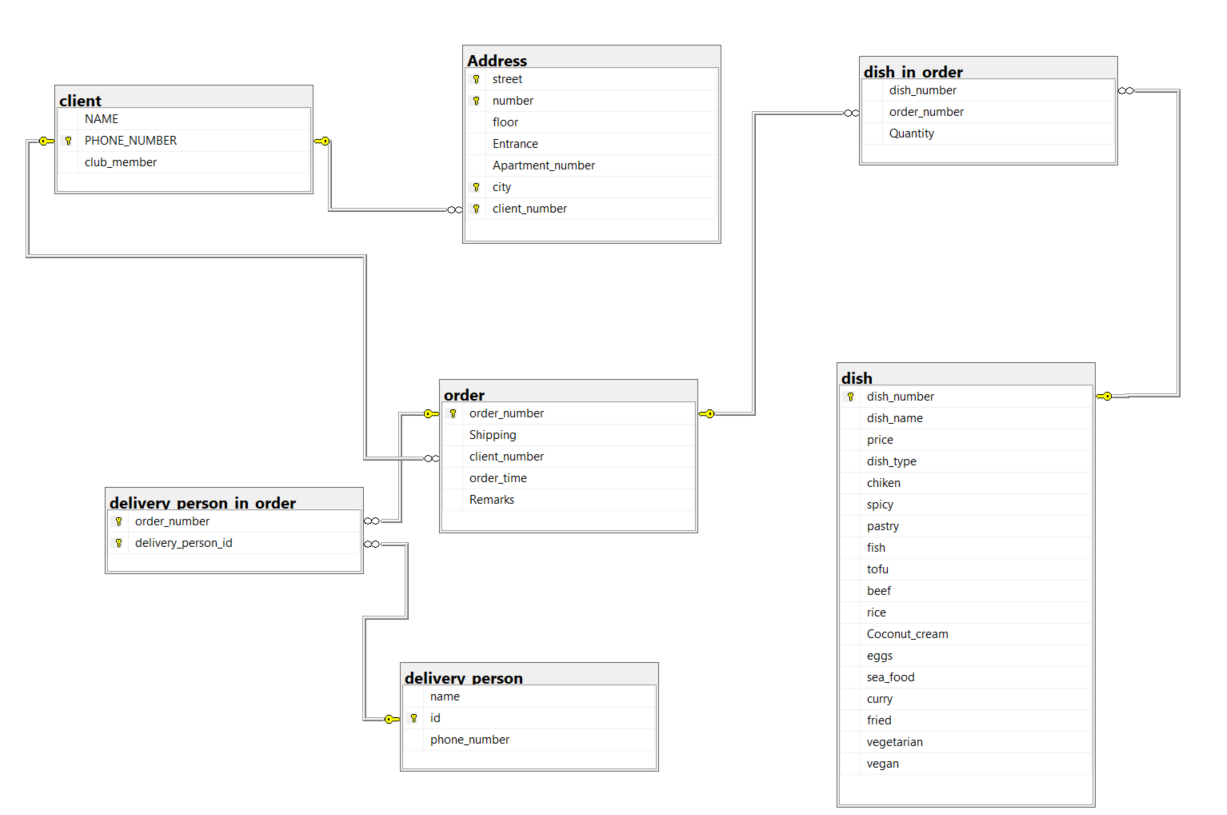
תיאור מקורות איסוף הנתונים

הנתונים נאספו מתפריט המסעדה ומקבלות של הזמנות שנאספו על ידי שליחים תקופה ארוכה ואותם אנו טוענים לתוך טבלאות SQL אותן יצרנו בעצמנו.

תיאור משתנה המטרה והגדרת הצלחה של המערכת

משתנה המטרה שלנו במערכת הוא מנה מתפריט המסעדה עליה נמליץ למשתמש בסיום תהליך ההזמנה שלו. משנה זה יכול לקבל כל מנה שקיימת בתפריט למעט אלו שהמשתמש כבר הוסיף בעצמו להזמנה. הצלחה של המערכת מוגדרת באופן בינארי כך ש-1 מסמן הצלחה של המערכת (מקרה בו המשתמש אכן הזמין את המנה עליה המערכת המליצה לו) ו-0 מסמן אי הצלחה.

מודל מסד הנתונים



create table client(

[NAME] VARCHAR(40) not null,

[PHONE\_NUMBER] varchar(10) primary key,

[club\_member] bit default '0'

);

create table Address\_(

[street] varchar(20) not null,

[number] varchar(3) not null,

[floor] varchar(2) null default '0',

[Entrance] varchar(2) null default '1',

[Apartment\_number] varchar(2) null default '1',

[city] varchar(20) not null,

[client\_number] varchar(10) references client(phone\_number),

primary key(client\_number, city, street, number)

);

create table order\_(

[order\_number] int primary key identity(00000,1),

[Shipping] bit not null,

[client\_number] varchar(10) references client(phone\_number),

[order\_time] date default getdate(),

[Remarks] varchar(100)

);

create table dish(

[dish\_number] int primary key identity(10001,1),

[dish\_name] varchar(50) not null,

[price] int not null,

[dish\_type] varchar(32),

[chiken] bit default '0',

[spicy] bit default '0',

[pastry] bit default '0',

[fish] bit default '0',

[tofu] bit default '0',

[beef] bit default '0',

[rice] bit default '0',

[Coconut\_cream] bit default '0',

[eggs] bit default '0',

[sea\_food] bit default '0',

[curry] bit default '0',

[fried] bit default '0',

[vegetarian] bit default '0',

[vegan] bit default '0'

);

create table dish\_in\_order(

[dish\_number] int references dish(dish\_number),

[order\_number] int references order\_(order\_number),

[Quantity] int

);

create table delivery\_person(

[name] varchar(20),

[id] int identity(10,1) primary key,

[phone\_number] varchar(10)

);

create table delivery\_person\_in\_order(

[order\_number] int references order\_(order\_number),

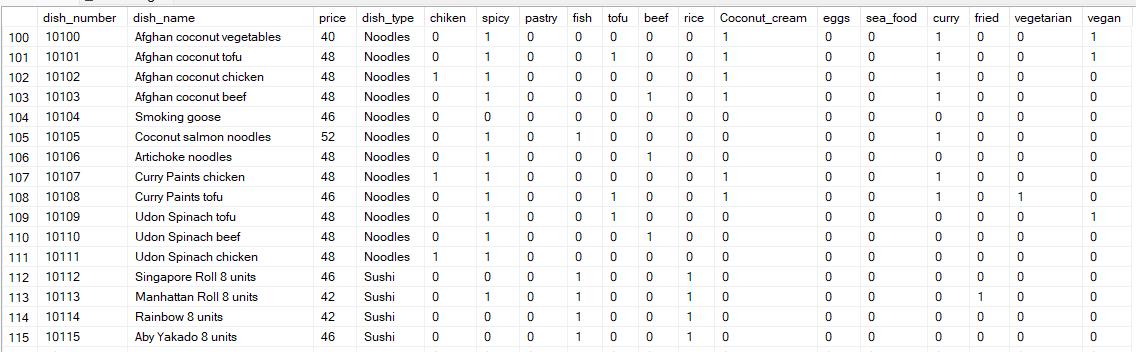
[delivery\_person\_id] int references delivery\_person(id),

primary key(order\_number, delivery\_person\_id))

תיאור הפיצ'רים הצפויים

הפיצ'רים הצפויים להיאסף הם מנות שהוזמנו בהזמנות עבר אותם אנו שומרים במערכת. הטבלה תיראה כך שכל שורה בה מייצגת משתמש וכל עמודה מייצגת מאפיינים מהזמנות עבר שלו בצורת פיצ'רים מסוג שם מנה, מחיר, סוג, האם קיים בה עוף/חריף/מאפה/דג/טופו/בקר/אורז/קרם קוקוס/ביצים/פירות ים/קארי/האם היא מטוגנת והאם היא צמחונית/טבעונית. בתפריט המסעדה קיימות קרוב ל-200 מנות ולכן בחרנו שלא להפיק את משתנה המטרה אך ורק על פי קיימות של מנות מסוימות בהזמנת המשתמש וגם בהזמנות עבר אלא לחפש את הקרבה של המנות אותן כבר בחר לאלו השמורות המערכת על פי הפיצ'רים של כל מנה. מתוך ההזמנה שתימצא הכי קרובה לזאת של המשתמש "החדש" נמצא את משתנה המטרה שלנו שהוא המנה שהכי מתאימה לו אך הוא לא הוסיף אותה מיוזמתו להזמנה ועליה נמליץ לו.

טבלת המנות עם הפיצ'רים הקיימים בה נראת כך:



תיאור הפיצ'רים בצורתם הסופית:

|  |  |  |
| --- | --- | --- |
| **שם הפיצ'ר** | **תיאור הפיצ'ר** | **סוג המשתנה** |
| price | המחיר הממוצע של המנות אותן הזמין המשתמש | כמותי |
| chicken | כמות הפעמים שהזמין מנות שכללו עוף | כמותי |
| spicy | כמות המנות שכללו חריף | כמותי |
| pastry | כמות המנות שכללו מאפה | כמותי |
| fish | כמות המנות שכללו דג | כמותי |
| tofu | כמות המנות שכללו טופו | כמותי |
| beef | כמות המנות שכללו בקר | כמותי |
| rice | כמות המנות שכללו אורז | כמותי |
| Coconut cream | כמות המנות שכללו קרם קוקוס | כמותי |
| eggs | כמות המנות שכללו ביצים | כמותי |
| Sea food | כמות המנות שכללו פירות ים | כמותי |
| curry | כמות המנות שכללו קארי | כמותי |
| fried | כמות מנות המטוגנות | כמותי |
| vegetarian | כמות המנות הצמחוניות | כמותי |
| vegan | כמות המנות הטבעוניות | כמותי |
| sushi | כמות הסושי | כמותי |
| Appetizer | כמות הראשונות | כמותי |
| Soups | כמות המרקים | כמותי |
| Salads | כמות הסלט | כמותי |
| Crispy Chicken mains | כמות העוף קריספי | כמותי |
| Special | כמות הספיישל | כמותי |
| Wok mains | כמות המנות ווק | כמותי |
| Pad Thai | כמות הפאד תאי | כמותי |
| Mains from sea | כמות העיקריות מהים | כמותי |
| side dish | כמות התוספות | כמותי |
| Noodles | כמות הנודלס | כמותי |
| desserts | כמות הקינוחים | כמותי |

דרכי התמודדות עם בעיות במדגם

המערכת שלנו בנויה כך שלא יכולים להיות בה ערכים חסרים משום שלקוח נשמר רק לאחר שביצע לפחות הזמנה אחת והמערכת לא תאפשר שערך חיוני כלשהו יוחסר. על מנת לטפל במקרים בהם לקוחות מסוימים מזמינים לעיתים תכופות יותר או בכמויות גדולות/קטנות מהממוצע באופן חריג החלטנו לנרמל את הפיצ'רים כך שהמשתנים הכמותיים לא יהוו את כמות הפיצ'ר x שהזמין לקוח בעבר אלא את הכמות של x מתוך מספר כל המנות הזמין אותו הלקוח בעבר x/n וכך לדוגמא לקוח ותיק שהזמין כבר 100 מנות ומתוכן 50 מנות כללו דג ולקוח חדש שהזמין רק 2 מנות ומתוכן מנה אחת כללה דג יקבלו בפיצ'ר "fish" את אותו הערך 1/2.  
בנוסף החלטנו לבצע נרמול נוסף, הפעם את הנירמול המוכר של כל העמודות כדי למנוע הבדלים משמעותיים בהשפעה על המודל בין הפיצ'רים.

ממשקי משתמש

עקרונות וסטנדרטים

ממשק המשתמש במערכת שלנו הוא הממשק הקיים של אפליקציית טלגרם, דרכה מתבצע החיבור בין המערכת שלנו למשתמש. אנו מתכננים את אופן השימוש במערכת כך שהמשתמש יכניס למערכת נתונים בעזרת לחיצה על כפתור מתוך מספר אופציות וניתן לו להקליד בעצמו כמה שפחות נתונים כדי לספק לו את החוויה הנוחה והמהירה ביותר.

פירוט המסכים המתוכננים :

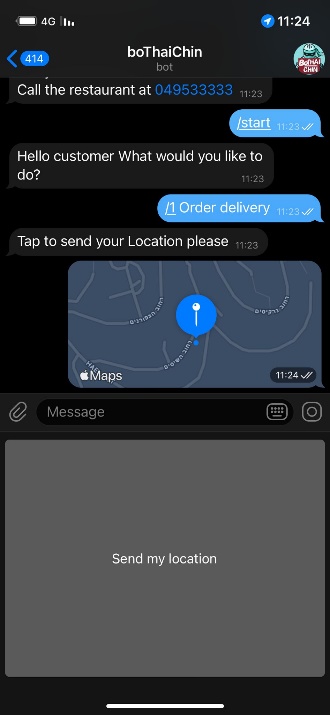
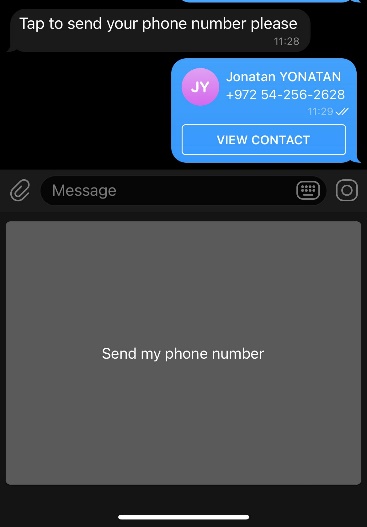
תמונה שמכילה טקסט, צג, צילום מסך

התיאור נוצר באופן אוטומטי

לאחר שהמשתמש ילחץ על הקישור שמוביל לצ'אט אפליקציית טלגרם תיפתח לו באופן אוטומטי ותישלח ההודעה /start שהיא בעצם בפקודה שמתחילה את תהליך ההזמנה:

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי  
  
המשתמש קיבל במסך הקודם אופציות לבחירה לשימוש במערכת והוא בחר להזמין משלוח:  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
המערכת מבקשת שיקיש על הכפתור שלוקח ממכשירו של המשתמש את מיקומו על מנת שהשליח ידע לאן מיועד המשלוח ומיד לאחר מכן המערכת מבקשת את מספר הטלפון של מבצע ההזמנה:

לאחר שהמערכת קיבלה מהמשתמש את המידע האישי שהיא זקוקה לו אנו מוכנים להתחיל בבחירת המוצרים להזמנה, המשתמש בוחר מתוך תפריט סוגי המנות שיש למסעדה להציע את המנות עצמן והוא רשאי ללכת קדימה ואחורה בין התפריטים:

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטיתמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי

תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי  
  
המשתמש סיים לבחור את המנות אותן הוא רוצה בהזמנה שלו וכעת המערכת ממליצה לו על מנה נוספת שלפי אלגוריתם למידת הלקוח שהופעל קיים סיכוי דיי סביר שהלקוח שלנו אכן יהיה מעוניין להוסיפה להזמנה:

מודל ניתוח הנתונים

תוכנית ניתוח סטטיסטי

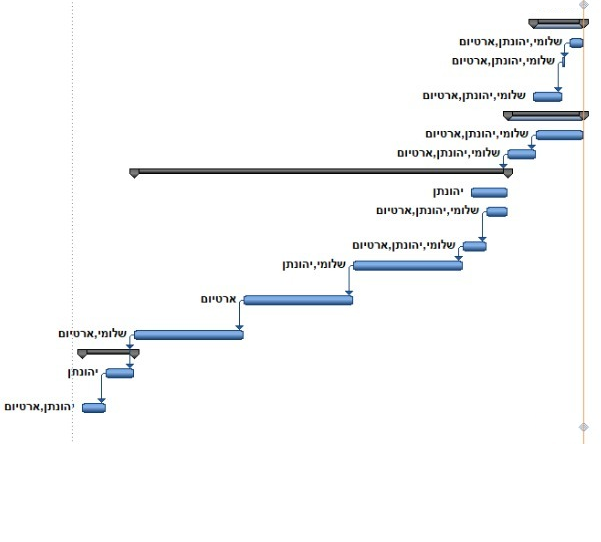
על מנת שנוכל לחזות בצורה המדויקת ביותר על איזו מנה נרצה להמליץ למשתמש שמבצע את ההזמנה נשתמש באלגוריתם ממשפחת מערכות ההמלצה (Recommended System) מבוססת תוכן אבל שמשווה בין משתמש למשתמש, מעין שילוב של user-user ו-item-item. נגדיר משתנה מטריציוני שיכיל את הדאטה סט שתיארנו בסעיף 4.4 ו-ווקטור שיכיל את אותם הפיצ'רים עבור המשתמש הנוכחי. כעת נחשב את המרחק בין המשתמש לבין שאר משתמשי המערכת לאורך ההיסטוריה, נמצא את ארבעת המשתמשים הקרובים ביותר  
אל מבצע ההזמנה ונשלוף חמשת המנות שאותם משתמשים "אהבו" ביותר, את הדירוג של המשתמשים בעבור המנות שפלנו גם כן בעזרת שאילתת SQL שמבצעת חישוב דומה לזה שבוצע כשיצרנו את טבלת הפיצ'רים רק שהפעם חילקנו את כמות הפעמים שמשתמש הזמין מנה מסוימת בכמות ההזמנות שאותו משתמש ביצע סה"כ, סכמנו את הדירוגים שהתקבלו מכל ארבעת המשתמשים הקרובים ושלפנו את חמשת המנות שקיבלו את הציונים הגבוהים ביותר. מרגע הפעלת המערכת נשמור בעבור כל המלצה שלנו בעבור כל הזמנה משתנה בינארי שיכיל את הערך 0 במידה והמשתמש בחר שלא להזמין את המנה המומלצת ו-1 במידה וכן, בעזרת בדאטה שנאגור נדע לאילו פיצ'רים יש השפעה רבה יותר/פחות ולדעת לייחס להם משקל בהתאם בהזמנות עתידיות וכך לשפר את האלגוריתם מפעם לפעם.

פירוט הדוחות של תוצאות הניתוח הסטטיסטי

פיצ'ר הדוחות יהיה זמין רק לדרג הניהולי מבין משתמשי המערכת והוא ייצר בעבורם תרשים Bar שיראה מנות ואת הפופלריות שלהן, את העלייה/הירידה במחיר הממוצע להזמנה בתרשים ציר זמן ועוד גרפים למיניהן שיציגו ללקוח (בעל המסעדה) נתונים סטטיסטיים בנוגע להזמנות שבוצעו דרך המערכת.

לוח זמנים מפורט לביצוע הפרויקט

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| מרכיב WBS | תיאור | משך העבודה | פעילויות קדם | משאבים |
| **BoThaiChin** |  | 171 ימים |  |  |
| 1. **תכנון מקדים** |  | 15 ימים |  |  |
| 1.1 הסכמה על רעיון | סיעור מוחות בין חברי הצוות וקבלת רעיון סופי | 5 ימים | - | כל חברי הצוות |
| 1.2 פגישה עם המנחה וקבלת אישור |  | יום אחד | 1.1 | כל חברי הצוות |
| 1.3 הכנת מסמך ייזום |  | 9 ימים | - | כל חברי הצוות |
| 1. **אפיון המערכת** |  | 21 | 1.3 |  |
| 2.1 פגישה של המפתחים | ביסוס אופי המערכת על ידי בניית דיאגרמות הכוללות דרישות, מחלקות ו-use case | 14 ימים | - | כל חברי הצוות |
| 2.2 הכנת מסמך אפיון |  | 7 ימים | 2.1 | כל חברי הצוות |
| 1. **פיתוח** |  | 113 ימים | 2.2 |  |
| 3.1 איסוף נתונים | קבלת בסיס הנתונים מהלקוח | 10 ימים | - | יהונתן |
| 3.2 בחירת שפת תוכנה עיקרית לפרויקט | חשיבה על צרכי המערכת ובחירת שפת תוכנה מתאימה. | 6 ימים | - | כל חברי הצוות |
| 3.3 למידה של הטכנולוגיה | השוואה בין מספר מודלים ובחירת המודל המתאים ביותר | 7 ימים | 3.2 | כל חברי הצוות |
| 3.4 Backend Development |  | 30 ימים | 3.3 | שלומי ויהונתן |
| 3.5 Front-End Development |  | 30 ימים | 3.4 | ארטיום |
| 3.6 UX Development | עיצוב המוצר בצורה קלה ונוחה לשימוש עבור הלקוח | 30 ימים | 3.5 | ארטיום ושלומי |
| 1. **בדיקות** |  | 14 ימים | 3.5 |  |
| 4.1 ביצוע בדיקות למערכת על ידי הדמיית לקוח | פיילוט ראשוני למערכת | 7 ימים | - | יהונתן |
| 4.2 תיקונים ושיפורים | ביצוע שינויים שהתגלה הצורך בהם בבדיקה | 7 ימים | 4.1 | יהונתן וארטיום |



נקודות פתוחות ומהדורות עתידיות של המערכת

מאחר ואנו עובדים תחת מגבלות זמן ותקציב מצומצם קיימים מספר פיצ'רים שלא יבוצעו במסגרת הפיתוח הראשוני ויתווספו בגרסאות מתקדמות יותר של המערכת:

* צ'אט ייעודי עבור שליחי המסעדה אשר יאפשר להם לקבל מידע אודות משלוחים ותקשורת ישירה אל מול הלקוחות המזמינים.
* פעילות הצ'אט בפלטפורמות נוספות כגון facebook messenger ו-WhatsApp .
* ניהול שיח בצורה מלאה באמצעות התכתבות בלבד בין המערכת והמשתמשים.

בנוגע לנושאים שטרם התקבלה ההחלטה הסופית לגבי הצורך בהם, בשלב זה אנו עדיין בוחנים את חבילת chatbotAI ב-python שמטרתה היא לשפר את השיח החופשי בין המערכת למשתמשים.