**Postaround. me – technical spec**

JSON

**שדות**

**msgId** – integer number, message id. number to identity the message.

**Latitude** – float number, presents latitude of current message

**Longitude** - float number, presents longitude of current message

**Category** – text, presents message's category name

**Userid** – integer number, number to identity the user

**Name** – text, presents the user full name

**Title** – text, presents the title of the message

**Description** – text, presents the full body the message

**Image** – text, url to the image to be shown next to the message, usually user's avatar

**Date** – short datetime to determine the date the message was written

**Time** – 24hrs time. Determine the time the message was written

**Type** – integer number, distinguish between a commercial message, to a regular message

**Distance** – integer number, how far the message was written from me. In meters.

**CommentsNumber** – integer number, default 0, presents the number of comments the message got.

Example:

{ "msgId":"1" ,"latitude": "32.075446", "longitude":"34.774997", "category": "brro", "userid": "", "Name":"Yaniv Aharon", "title":"ladder", "description":"i need to replace my spots on my ceiling and so i need a ladder for 30 mins. thanx! ", "image":"images/186051\_527172722\_4512233\_q.jpg", "Date":"29 May 2011", "Time":"15:14", "type": "", "Distance":"100", "CommentsNumber":"31" }

{"msgId":1,"latitude":"lat","longitude":"lon","category":"Rent","catID":0,"userid":1,"Name":"Yaniv Aharon","title":"title","description":"desc","image":"images/186051\_527172722\_4512233\_q.jpg","FullDate":"\/Date(-62135596800000)\/","Date":"11/11/2011","Time":"16:30","type":3,"Distance":0,"CommentsNumber":2,"regionID":0}

מבנה תגובות:

שדות:

msgId – foreign key to presents to what message this comment belong

commentId – identity number of the message

date – the date the comment was written. Full date and time.

userId – the id of the user that wrote the comment

DATABASE

**Messages**

**msgId** – integer number, message id. number to identity the message.

**Latitude** – float number, presents latitude of current message

**Longitude** - float number, presents longitude of current message

**CategoryID** – int

**Userid** – integer number, number to identity the user

**Title** – text, presents the title of the message

**Description** – text, presents the full body the message

**Image** – text, url to the image to be shown next to the message, usually user's avatar

**Date** – full date time

**Type** – integer number, distinguish between a commercial message, to a regular message

**Region –** integer number represents the region number on the map. For improved implementation.

**Users**

**id**

**name**

**email**

**phone 1**

**phone 2**

**address**

CityID - ?

StateID - ?

CountryID - ?

longtitude

latidute

image

**LoginHistory**

userId

Date

**Categorys**

Id

Name

**Comments**

ID

MessageID

UserID

Body

Date

פונקציות DAL

**GetMessages(float Longtitude = null, float Latidute = null, int RegionID = null)**select messages from database.

If regionID != null: get messages by regionID and calculate their distance from given lat + lon.  
Else, get all messages and calculate their distance from given lat + lon.

The output result will be represented in a json array. As mentioned above.

**InsertMessage(Lat, Lon, CategoryID, userId, title, description, image, date, type, region)**insert single message into the database.

**GetMessagesByRange(int range, Lat, Lon)**

The function will get range, Lat Lon

Return: all the messages that were written up to this range, by Lat Lon

The distance may be calculated by two options:

1. Beta version : calculate the distance of each message from me.
2. Improved: Calculate the desired regions that we want to get the messages from. And select all the messages with that regionIDs.

**GetMessagesByCategories(int represetor)**

Find the index of the bits that is on with the 'representor' number. Then get all the messages with these index ID.

**GetMessagesByUserId( int userId)**

Get all the messages that was written by the given userId

**GetMessageByID(int msgID)**

Get message info by its id.

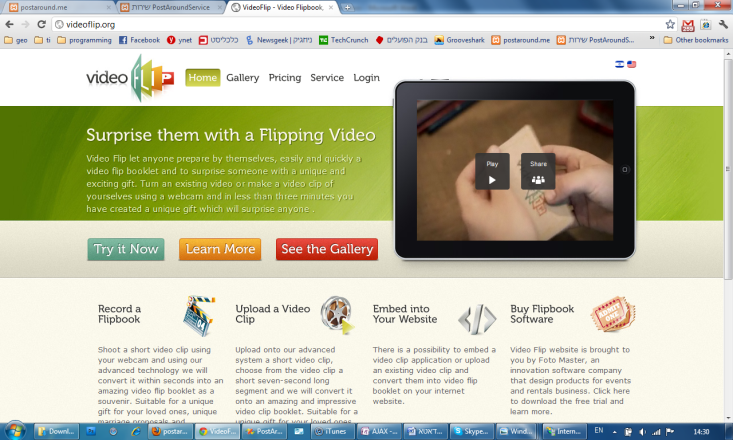
**UI**

**אפיון האתר**

**מסך ראשון:**

לוגו, הסבר קצר על האתר, קוביית לוגין עם פייסבוק

דוגמא:





**מסך שני (ראשון אחרי לוגין):**



**שורה ראשונה**

לוגו, שורת סטטוס, כפתור פוסט, לוגאין-לוגאאוט

**שורה שניה**

כפתורים:

1. מרחק – תמיד מסומן
2. לוקיישן – לחיצה תפתח חלון עם הצגת המיקום הנוכחי עם אפשרות לשינוי ידני
3. קטגוריה – לחיצה תפתח חלון עם הצגת הקטגוריות עם צ'ק בוקסים
4. קיוורד – לחיצה תפתח חלון שבו ניתן להזין מילת חיפוש לסנן על פיה
5. מיי פוסטס – לחיצה תציג רק את הפוסטים שלי

**שורה שלישית**

ההודעות. בשלב זה יוצגו 16 הודעות בעמוד (פיתוח – לפרוס את ההודעות על המסך כמה שנכנס) שלצידיהם כפתורי  
קדימה-אחורה.

**שורה רביעית**

קופירייט + לוגו + לינקים לפוטר

**הכנסת הודעה**

שמירת ההודעה בדאטאבייס רענון הבקשות מהדאטאבייס והצגה מחודשת של ההודעות על המסך.

**הצגת הודעה**

1. העמוד יקבל מערך של ג'ייסון של כל ההודעות באתר. ויציג אותם בתוך הקוביות תוך חלוקה לעמודים   
   (לממש פייג'ר).
2. כל פריט ג'ייסון יכיל מידע אודות המרחק מהמיקום הנוכחי.

**אפיון תהליכים:**

**תוכן**

1. יצירת הודעה חדשה
2. צפיה בהודעות קיימות
3. סינון תוצאות
4. דפדוף בין הודעות
5. הגדרות אישיות
6. לוגאאוט
7. **יצירת הודעה חדשה**

משתמש ממלא את פרטי ההודעה, לוחץ על POST, נפתח חלון חדש ובו ניתן להוסיף פרטים נוספים:

1. כותרת
2. פרטי ההודעה (מתוך מה שהזין בטקסט בוקס)
3. קטגוריה (מתוך דרופ דאון קטגוריות)
4. פרטי יצירת קשר עם צ'ק בוקס לידם:
   1. דואר אלקטרוני
   2. טלפון 1
   3. טלפון 2
   4. מיקום (רישום אוטומטי)
5. כפתור איקס לסגירה
6. כפתור אישור – שומר את המידע בדאטאבייס ומרענן את מסך ההודעות
7. כפתור ביטול – מוחק את כל המידע שהוזן וסוגר את החלון

**פעולת כפתור האישור**

מעבירה לסרבר json ומבצע שמירה בדאטאבייס של הפרטים הבאים:

**Latitude** – float number, presents latitude of current message

**Longitude** - float number, presents longitude of current message

**CategoryID** – מהדרופ דאון של הקטגוריות

**Userid** – משתנה מחלקה, עבור היוזר הנוכחי

**Title** – כותרת ההודעה

**Description** – תיאור ההודעה

**Date** – תאריך נוכחי. אוטומטי מהשרת

**Type** – ברירת המחדל - 1

**Region –** כרגע ריק. בעתיד קוד האיזור

לאחר השמירה, הצגה של ההודעה בראש ההודעות.

**צפיה בהודעות קיימות**

הצגת ההודעות יתבצע באמצעות HANDLER.

**GetMessagesByRange(int range, Lat, Lon)**

ה HANDLER הולך לדאטאבייס, ושולף את כל ההודעות בטווח.

BY DEFAULT – מוצגות כל ההודעות עד מרחק של 5KM.

ההודעות ממויינות לפי תאריך

**סינון הודעות**

סינון התוצאות מתבצע בכמה רבדים:

1. מיקום
2. קטגוריה
3. מילת מפתח
4. שלי
5. מרחק ממני

**דפדוף בהודעות**

כל עמוד מכיל 16 הודעות (עתידי, חישוב אוטומטי לפי רוחב המסך)

כאשר במידה ויהיו יותר מ16 הודעות, יהיה פייג'ר

לחיצה על החיצים תעביר ל16 הבאים בתור

**הגדרות משתמש**

1. 2 צ'קבוקסים בתחתית:
   1. מעוניין לקבל התראה במייל על מענה
   2. פרסם על הקיר שלי בפייסבוק