

הרשות להגנת הצרכן ולסחר הוגן

"אל תתקשר אלי"

מסמך ממשקי API לחברות הטלמרקטינג



Version 3.8



10 בנובמבר, 2022

תוכן

טבלת שינויים במסמך	3
1. כללי	4
1.1 מטרת המסמך	4
1.2 תכולת המסמך	4
1.3 הנחות עבודה והגדרות כלליות	4
2. כיווץ המידע	5
3. כללי API	9
3.1 API תהליך ההזדהות מול ה	9
3.2 מבנה תשובה כללי	9
3.3 קודי תשובה והודעות כלליות	9
4. רשימת השירותים	11
4.1 phone-query / שאילתה גדולה	11
4.2 Phone-list-verification	12
5. פעולות נוספות לביצוע דרך הפורטל	13
a. שאילתה קטנה (עד 10 רשומות)	13
b. רשומות 100K שאילתה בינונית (עד	14
c. גישה ללוגים	14
iv. API בדיקת תקינות קריאות	14
a. Postman לתוך collection ייבוא של ה	14

טבלת שינויים במסמך

גרסת מסמך	עורך	תאריך	שינויים
V1.0	ניסן לוי	28/3/2022	גרסה ראשונה
V2.0	בתאל דרורי	28/3/2022	עדכון סעיף 5 – פעולות לביצוע דרך הפורטל – גישה ללוגים
V2.1	בר נרקיס	04/04/2022	עדכון סעיף 4.1 וסעיף 1
V2.2	בר נרקיס	11/04/2022	עדכון סעיף 1 ו6
V3	בר נרקיס	14/04/2022	עדכון סעיף 6 ופרטי יצירת קשר
V3.1	יולי	20/05/2022	עדכון סעיף 1.3.3 - עדכון דומיין של סביבת טסט
V3.2	בר נרקיס	25/05/2022	עדכון פרטי יצירת קשר
V3.3	יולי	25/7/2022	עדכון סביבת טסט והאדרים בבקשה
V3.4	יולי	21/09/2022	עדכון כתובת לסביבת טסט
V3.5	ניקול	02/10/2022	עדכון כתובת לסביבת פרודקשן
V3.6	יולי	20/10/2022	הוספת פונקציה ללא כיווץ
V3.7	יולי	30/10/2022	הוספת טוקן קבוע לפרוד + הוספת הערה למיין בphone query
V3.8	ניקול	10/11/2022	שינוי כתובת PHONE LIST VERIFICATION

1.1 כללי

1.1.1 מטרת המסמך

מטרת פרויקט "אל תתקשר אלי" היא הפחתה משמעותית בכמות שיחות הטלמרקטינג לצרכנים שאינם מעוניינים לקבלם. הצרכנים יוכלו להירשם למאגר של מספרי טלפון. חברות טלמרקטינג יהיו מחויבות לבדוק אם מספר נמצא במאגר לפני שהן מתקשרות אל הצרכנים.

חברות הטלמרקטינג יעבירו בעזרת ממשקים המפורטים במסמך זה מספרי טלפון לבדיקה. לאחר הבדיקה ואישור המספר הרלוונטי, מוקדני הטלמרקטינג יוכלו ליצור קשר עם אותם אזרחים שמאושר להתקשר אליהם. מטרתו של מסמך זה היא לפרט את דרכי החיבור והשימוש של ה-API, להכרת הרכיבים שבו ואופן פעולתם. המסמך מיועד עבור חברות הטלמרקטינג, המבקשות להתחבר לממשק של פרויקט "אל תתקשר אלי".

1.2 תכולת המסמך

מסמך זה מכיל תיעוד מקיף של הפונקציות הזמינות לשימוש, תחקור וביצוע פעולות מול שירות "אל תתקשר אלי" (ה-API).

המסמך מכיל בתוכו הסברים מפורטים לגבי מבנה הבקשות והתשובות המקובלות על מנת לבצע תקשורת בין משתמש הקצה אל מול ה-API.

עבור כל שירות יופיע הסבר קצר על פעולת השירות, מבנה הבקשה והפרמטרים השונים, מבנה התשובה, דוגמאות עבור קלט ופלט תקינים והודעות שגיאה רלוונטיות.

כל פעולה מול ה-API מקבלת מספר טרנזקציה ייחודי המוחזרת למבצע הבקשה יחד עם תשובת השירות. מספר טרנזקציה זה מהווה מזהה ייחודי ועליו להיות מצורף יחד עם כל בקשה לטיפול בתקלה כל שהיא.

1.3 הנחות עבודה והגדרות כלליות

i. גודל בקשות ותשובות
ה-API מאפשר קבלה ושליחה של הודעות עד גודל של 6 מגה.
הודעות המגיעות אל ה-API או נשלחות ממנו החורגות ממגבלה זו יחסמו עם הודעת שגיאה אינפורמטיבית.

ii. מבנה טלפון תקין
מרבית ממשקי ה-API דורשים קבלה של מספר טלפון לביצוע הפעולה.
על מנת לפשט את פעולת ה-API על מבצע הבקשה לשלוח מספר תקין באורך של 9 (מספר ביתי) או 10 (מספר נייד) תווים בלבד.
על המספר להכיל מספרים בלבד ללא תווים מיוחדים או קידומות מיוחדות ככל שיהיו.

כל מספר טלפון החורג מהנחיה זו יוחזר עם שגיאה אינפורמטיבית מתאימה.

דוגמאות למספר טלפון תקין:

- 0501111111
- 0356488888

דוגמאות למספרי טלפון שאינם תקינים:

- +972501111111
- 0506-111111
- 03-5648888

iii. סביבות עבודה - api_base_url

:TEST

<https://dnc-dev.fta.gov.il/restricted/>

: PROD

[/https://dnc-api.fta.gov.il/restricted](https://dnc-api.fta.gov.il/restricted)

לטופס פניה בנוגע לשאלות במסמך זה יש להכנס ללינק הבא:

<https://govforms.gov.il/mw/forms/Telemarketing@Cpfta.gov.il>

2. כיווץ המידע

על מנת לבצע שאילתות "כבדות" אל מול ה API נדרש לכווץ את המידע לפני שליחתו אל ה API. כמו כן, על מנת להגדיר קבלה של תשובות מכווצות מצד ה API נדרש להוסיף Header מתאים לבקשה.

i. על מנת לקבל תשובות מכווצות מה API יש להוסיף את ה Header הבא לבקשה:

Accept-Encoding: gzip, deflate, br

ii. על מנת לייצר בקשות מכווצות יש לבצע את הפעולות הבאות:
JavaScript (using axios) ○

```
import axios from 'axios';
import pako from 'pako';

const api = axios.create({
  baseURL: "http://localhost:9000/api",
  withCredentials: true,
  transformRequest: axios.defaults.transformRequest.concat(
    function (data, headers) {
      // compress strings if over 1KB
      if (typeof data === 'string' && data.length > 1024) {
        headers['Content-Encoding'] = 'gzip';
        return pako.gzip(data);
      } else {
        // delete is slow apparently, faster to set to undefined
        headers['Content-Encoding'] = undefined;
        return data;
      }
    }
  )
});

export default api;
```

Javascript (using pako) ○

```
return new Promise((resolve, reject) => {
  gzip(JSON.stringify(payload)).then((gzippedBody) => {
    fetch(api_base + "/api/save-photo", {
      method: 'POST',
      mode: 'cors',
      headers: {
        'Content-Encoding': 'gzip',
        'Content-Type': 'application/json'
      },
      body: gzippedBody
    })
    .then((response) => {
      if (response.status === 404) {
        throw new Error('404 (Not Found)');
      } else {
        return response.json().then((json) => {
          console.log('save poster response: ', json);
          return json;
        });
      }
    });
  });
});
```

```
<script type="text/javascript" src="pako.js"></script>
```

then you can decompress data, such as a compressed JSON format, as such:

```
var data;
var request = new XMLHttpRequest();
request.responseType = 'arraybuffer';
request.onload = function() {
  data = JSON.parse(pako.inflate(request.response, { to: 'string' }));
};
request.open('GET', "data.gzip");
request.send();
```

[Java](#) ○

```
import java.io.InputStreamReader;
import java.io.Reader;
import java.net.HttpURLConnection;
import java.net.URL;

public class HttpConnect {
    public static void main(String[] args) throws Exception {
        URL url = new URL("http://www.rgagnon.com/howto.html");
        HttpURLConnection con = (HttpURLConnection) url.openConnection();
        con.setRequestProperty("Accept-Encoding", "gzip");
        System.out.println("Length : " + con.getContentLength());
        Reader reader = new InputStreamReader(con.getInputStream());
        while (true) {
            int ch = reader.read();
            if (ch==-1) {
                break;
            }
            System.out.print((char)ch);
        }
    }
}
```

[#C](#) ○

```
var client = new JsonServiceClient(baseUrl) {
    RequestCompressionType = CompressionTypes.GZip,
};

var client = new JsonHttpClient(baseUrl) {
    RequestCompressionType = CompressionTypes.Deflate,
};

var response = client.Post(new Request { ... });
```

Net Core. ○

```
using (var httpContent = new StringContent(stringPayload, Encoding.UTF8, "application/json"))
using (var compressedContent = new CompressedContent(httpContent, "gzip"))
using (HttpResponseMessage response = client.PostAsync("Controller/Action", compressedContent).Result)
{
    if (response.StatusCode != System.Net.HttpStatusCode.OK)
    {
        throw new Exception(string.Format("Invalid responsecode for http request response {0}: {1}", response.StatusCode, response.ReasonPhrase));
    }
}
```

By using a middleware ○

```
public class GzipRequestMiddleware
{
    private readonly RequestDelegate next;
    private const string ContentEncodingHeader = "Content-Encoding";
    private const string ContentEncodingGzip = "gzip";
    private const string ContentEncodingDeflate = "deflate";

    public GzipRequestMiddleware(RequestDelegate next)
    {
        this.next = next ?? throw new ArgumentNullException(nameof(next));
    }

    public async Task Invoke(HttpContext context)
    {
        if (context.Request.Headers.Keys.Contains(ContentEncodingHeader) && (context.Request.Headers[ContentEncodingHeader] == ContentEncodingGzip ||
            context.Request.Headers[ContentEncodingHeader] == ContentEncodingDeflate))
        {
            var contentEncoding = context.Request.Headers[ContentEncodingHeader];
            var decompressor = contentEncoding == ContentEncodingGzip ? (Stream)new GZipStream(context.Request.Body, CompressionMode.Decompress, true) :
                (Stream)new DeflateStream(context.Request.Body, CompressionMode.Decompress, true);
            context.Request.Body = decompressor;
        }
        await next(context);
    }
}
```


3. API - כללי

3.1 תהליך ההזדהות מול ה-API

שימוש בחלק מהשירותים של ה-API דורש שליחה של ACCESS TOKEN תקין כחלק מהבקשה עצמה. את ה-ACCESS TOKEN ושם tokenn יש לשלוח כ-AUTHORIZATION האדר לבקשה עצמה, לדוגמה:

"token": "a9b3a6f0-2701-40f1-99d6-4ea633dccb08",

"name": "123456_43844556",

את ה-ACCESS TOKEN ניתן לייצר בממשק הניהול באתר הפורטל תחת עמוד "הגדרות מנהל". ה-ACCESS TOKEN הנוצר יהיה זמין לתקופה בלתי מוגבלת ויהיה ניתן לבטלו גם כן תחת עמוד "הגדרות מנהל" באתר הפורטל. שימו לב- בסביבת הטסטים אין צורך לשלוח את הטוקן ואת שמו. יש לשלוח סטרינג ריק ב-HEADER-

- o "name": ""
- o "token": ""

3.2 מבנה תשובה כללי

i. להלן פירוט מבנה תשובה כללי עבור כל בקשה הנשלחת אל ה-API:

#	שם השדה	סוג השדה	פירוט
1	code	integer	קוד המתאר את תוצאת הבקשה
2	message	string	תיאור מילולי של תוצאה הבקשה
3	transactionId	string	מזהה ייחודי של הפעולה
4	data	object	מידע רלוונטי המצורף לתשובה (אופציונלי)

ii. דוגמה:

```
{
  "code": 200,
  "message": "Request completed successfully",
  "transactionId": "asdh-3454-jh65-a53g"
}
```

3.3 קודי תשובה והודעות כלליות

#	קוד תשובה	הודעה	תיאור
1	200	הפעולה הסתיימה בהצלחה	הצלחה
2	400	הבקשה אינה תקינה	מבנה הבקשה אינו תקין
3	401	אינך מורשה לגשת לשירות זה	כאשר הפונה לשירות אינו מאושר שימוש או שלא הועבר טוקן שימוש תקין

כאשר חלה תקלה ב API ולא ניתן לספק הסבר מדויק לסיבה שבגינה חלה התקלה	שגיאה פנימית - אנא פנה לתמיכה עם מזהה הבקשה.	500	4
---	--	-----	---

4. רשימת השירותים

4.1 שאילתה גדולה / phone-query

i. תיאור השירות:

באמצעות שירות זה ניתן לבדוק עד מליון מספרים שונים (ע"י כיווץ מידע) אל מול מאגר הנתונים הקיים של מספרי הטלפון הרשומים לשירות.

יש לשלוח את מספרי הטלפון ללא ספרת ה 0 ההתחלתית
יש למיין את המספרים לפני ביצוע הכיווץ.

תשובת ה API תכיל את המספרים אשר אינם רשומים במאגר ואת מספרי הטלפון בהם לא ניתן היה להחזיר תשובה תקינה עקב תקלה.

ii. כתובת: `api_base_url/restricted/phone-query`

iii. סוג: POST

iv. מבנה הבקשה

#	שם השדה	סוג השדה	פירוט
1	data	List <פריט לבדיקה>	רשימת מחרוזות של מספרים לבדיקה

v. מבנה התשובה

#	שם השדה	סוג השדה	פירוט
1	data	List <תוצאת בדיקה>	- מספרי טלפון שאינם רשומים למאגר – יוחזרו כמחרוזת - מספרי טלפון שלא נבדקו מול המאגר – יוחזרו כאובייקט

a. מבנה אובייקט עבור מספרים שלא נבדקו

#	שם השדה	סוג השדה	פירוט
1	phone	string	מספר טלפון תקין אשר לא רשום לשירות
	errorCode	integer	מזהה שגיאה (אופציונלי)

vi. קודי שגיאה

קוד שגיאה	סוג השדה
1	מספר הטלפון אינו במבנה תקין

vii. דוגמאות

a. בקשה

```
{
  "data": ["123", "516635487"]
}
```

b. תשובה

```
{
  "statusCode": 200,
  "message": "הפעולה הסתיימה בהצלחה",
  "transactionId": "e83de852-7d6b-4787-8c01-ade47c12775d",
  "data": [
    "0516635487",
    {
      "phone": "123",
      "errorCode": 1
    }
  ]
}
```

Phone-list-verification 4.2

i. תיאור השירות
 באמצעות שירות זה ניתן לבדוק עד 100K מספרי טלפון (ללא כיווץ) אל מול מאגר הנתונים הקיים של מספרי הטלפון הרשומים לשירות.
יש לשלוח את מספרי הטלפון ללא ספרת ה 0 ההתחלתית
 תשובת ה API תכיל את המספרים אשר אינם רשומים במאגר ואת מספרי הטלפון בהם לא ניתן היה להחזיר תשובה תקינה עקב תקלה.

ii. כתובת – BASE URL

Test – <https://pverification.fta.gov.il/restricted/phone-list-verification> ○

Prod - <https://pverification.fta.gov.il/restricted/phone-list-verification>

iii. סוג – POST
 iv. מבנה הבקשה

#	שם השדה	סוג השדה	פירוט
1	data	List <פריט לבדיקה>	רשימה של מחרוזות לבדיקה

v. מבנה התשובה

#	שם השדה	סוג השדה	פירוט
1	data	List <תוצאת בדיקה>	<ul style="list-style-type: none"> - מספרי טלפון שאינם רשומים למאגר – יוחזרו כמחרוזות - מספרי טלפון שלא נבדקו מול המאגר – יוחזר כאובייקט

מבנה אובייקט עבור מספרים שלא נבדקו

#	שם השדה	סוג השדה	פירוט
1	phone	string	מספר טלפון תקין אשר לא רשום לשירות
	errorCode	integer	מזהה שגיאה (אופציונלי)

.vi קודי שגיאה

קוד שגיאה	סוג השדה
1	מספר הטלפון אינו במבנה תקין

.vii דוגמאות

בקשה

```
{
  "data": ["123", "516635487"]
}
```

תשובה

```
{
  "statusCode": 200,
  "message": "הפעולה הסתיימה בהצלחה",
  "transactionId": "e83de852-7d6b-4787-8c01-ade47c12775d",
  "data": [
    "0516635487",
    {
      "phone": "123",
      "errorCode": 1
    }
  ]
}
```

5. פעולות נוספות לביצוע דרך הפורטל

בנוסף לשאילתה הגדולה שבה מבוצע התשאול ישירות מול הAPI, יהיו זמינות האפשרויות לביצוע שאילתה קטנה או שאילתה בינונית דרך פורטל DNC. בנוסף תהיה אפשרות לבצע תחקור של לוגים ישירות דרך הפורטל.

a. שאילתה קטנה (עד 10 רשומות)

- i. תבוצע באמצעות טופס בפורטל הWEB של "אל תתקשר אלי".
- ii. הטופס יוגבל לעד 10 מספרים.
- iii. לאחר בדיקת המספרים יוצגו אך ורק המספרים אליהם רשאי המשתמש להתקשר. כלומר המספרים שאינם רשומים למאגר "אל תתקשר אלי".

b. שאילתה בינונית (עד 100K רשומות)

- i. תבוצע באמצעות העלאת קובץ אקסל במבנה מוגדר מראש שיכיל את רשימת המספרים לבדיקה.
- ii. לאחר בדיקת המספרים יוצגו אך ורק המספרים אליהם רשאי המשתמש להתקשר. כלומר המספרים שאינם רשומים למאגר "אל תתקשר אלי".
- iii. המשתמש יקבל מידע לגבי הרשומות התקולות במידה ונמצאו כאלה.
- iv. פעולה זו תרשם ללוג ה-API.

c. גישה ללוגים

- i. בפורטל DNC יהיה קיים מסך ייעודי שבו תינתן אפשרות לחברות הטלמרקטינג לבצע תחקור של פניות עבר שבוצעו על ידה.
- ii. החיפוש יתאפשר ע"י הזנת הפרמטרים הבאים :
 - a. מזהה טרנזקציה (במידה ורוצים לקבל תוצאות עבור טרנזקציה ספציפית)
 - b. תאריך התחלה
 - c. תאריך סיום
 - d. מספר טלפון
- iii. תוצאות החיפוש יציגו למשתמש את הלוגים שנרשמו עבור אותם פרמטרים.

iv. בדיקת תקינות קריאות API

- i. Postman זוהי פלטפורמת לבניית API. ניתן לקרוא מידע נוסף אודות המערכת בלינק [הבא](#).
- ii. יש לבצע בדיקת תקינות בpostman ולוודא כי הקריאות מחזירות תשובות תקינות.
- iii. כדי לעבוד עם מערכת זו נדרש ליצור יוזר כניסה ע"י הלינק [הבא](#).

a. ייבוא של הcollection לתוך Postman

- i. יצרנו עבורכם collection מוכן עם הקריאות המפורטות במסמך.
- ii. ניתן להעזר במדריך [הבא](#) לצורך ייבוא הcollection המצורף בתת סעיף הבא.
- iii. מצורף הcollection בלינק [הבא](#).
- iv. שימו לב, במידה והcollection מיובא ללא הסביבות, יש להעתיק את הסביבות מסעיף 1.3