# שם הפרויקט :

חסד במחיר מציאה

# רקע

## **תיאור ורקע כללי**

מערכת מקוונת המיועדת בעיקר לרשת פנים ארגונית, ולקהל יעד ממוקד.

היא נועדה לשמש ככלי לניהול החזרת אבדות לבעליהם, באופן נגיש ומסודר.

המערכת דינאמית ומאפשרת התאמות פנים ארגוניות על פי צרכיו של כל לקוח אופציונלי.

## **מטרות המערכת**

## לספק כלי לפרסום מידי של מציאות במוסד לימודי או בכל מוסד ארגוני אחר, במטרה להקל על ניהולו השוטף של המוסד.

## ליצור מערכת אמינה זמינה ופשוטה לשימוש שתאפשר מציאה לפריטים אישיים של יחידים במוסד ותהווה כתובת בלעדית לפרסום כל מציאה.

# סקירת מצב קיים בשוק

כיום, מטפלים ארגונים ומוסדות רבים באבידות בדרך של פרסומים ספציפיים במקומות מרכזיים, או שלא מפרסמים כלל. במוסדות רבים אין ממש למי לפנות כדי להעזר במציאת ובפרסום אבדות,  
מערכת האבדות מהווה במוסדות רבים מטרד לא קטן, הן לצוות הניהול והן לבעלי אבדות, או למוצאים.  
הנהלת המוסד נדרשת לטפל בעשרות אם לא במאות ויותר מזה חפצים שאין לדעת אם ימצא להם דורש מתי והיכן, כל ההתנהלות מורכבת, לא מסודרת ויוצרת מצב לא אידיאלי בעליל.  
כשחבר במוסד מוצא חפץ יקר ככל שיהיה, אין לו דרך מסודרת ואחראית לידע את **כל** חברי המוסד האחרים במציאה במטרה למצוא את המאבד,  
קל וחומר שכשאדם מבין שאבד לו דבר מה, אין לו אפילו קצה חוט למציאת אבדתו, היא עשויה להתגלגל מיד ליד במשך זמן רב וקלוש הסיכוי למציאתו.

# מה הפרויקט אמור לחדש או לשפר

הפרויקט מציע מערכת לניהול אבדות ומציאות, ידידותית ונגישה, המאפשרת פרסום של כלל המציאות במוסד, דפדוף מהיר וחיפוש יעיל שינגיש למאבד את אבדתו ויחסוך עשרות העברות ידיים וחיפושים עצמאיים מורכבים.

המערכת תהיה נגישה לכל חבר במוסד ותהיה גמישה להתאמות פנים ארגוניות ע"י הנהלת הארגון.

מערכת מקוונת זו עתידה להעלות משמעותית את אחוזי מציאת האבדות, הודות למאגר האחיד המוחלט והידיעה הברורה שכשמשהו אבד, המקום הטבעי שלו להיות – זה במאגר הזה.

כמו כן המערכת תקל משמעותית על ניהול מוסדות בכך שתסיר מעליהם את עול הטיפול באבדות בצורה מוחלטת.

# דרישות מערכת ופונקציונאליות

## **דרישות מערכת** מסד נתונים ריאלציוני שיושתת ברשת הEthrnet של אותו המוסד חיבור וסינכרון בין המחשבים, מערכת הפעלה Windows התומכת בשפות .net

## **פונקציונאליות**

מנהל:

* + בוחר סיסמת מנהל
  + מזין כתובת מייל רשמית למוסד
  + מזין רשימת מיקומים על פי חלוקה מסודרת של המוסד
  + מזין קטגוריות ראשיות ומשנה כפי הנצרך במוסדו

מוצא:

* + מזין פרטי אבידה מלאים כולל בחירת קטגוריה.
  + מזין פרטים אישיים לפניה ישירה של המאבד.

מאבד:

* + מזין פרטי אבידה במערכת החיפוש.
  + צופה בנתוני מוצא.

# בעיות צפויות במהלך הפיתוח

## הבעיות:

### בעיה 1: התאמת המוצר לצרכים שונים של ארגונים, כמו מיקומים אפשריים למציאת אבידות (חלוקה על פי ערים / מדינות / קווי אוטובוס / אזורים בתוך מבנה וכדו), חלוקה לקטגוריות שונות ועוד.

### בעיה 2: קושי בהטמעת המערכת בארגונים ומוסדות שמרניים שיקשו לקחת חלק במהלך הדיגיטציה

## פתרונות אפשריים:

### לבעיה 1:

### פתרון 1: הכלת אפשרויות המתאימות לכל ארגון אפשרי.

* + פתרון 2 : יצירת חבילות מוכנות של נתונים לפי צרכים שונים ונתינת אפשרות בחירה בינהם.
  + פתרון 3: נתינת אפשרות עריכה אישית של מסד לכל לקוח מנהל.

### לבעיה 2:

### פתרון 1: ליווי בהטמעת המערכת בשלביה הראשונים.

### פתרון 2: תמיכה טכנית, סיוע וזמינות לטווח הרחוק.

### פתרון 3: חישוב תועלת ההשקעה שתופק ממערכת כזו לעומת הקושי הראשוני בשימושה.

## הפתרון הנבחר עבור כל אחת מהבעיות:

### לבעיה 1: פתרון 3, כל ארגון ישתמש בחשבון המנהל בכדי לגשת בעצמו למסד הנונים ולערוך אותו, את הקטגוריות של האבידות, את רשימת המיקומים האפשריים וכו.

# פתרון טכנולוגי נבחר

## **טופולוגית הפתרון**

## 

דיאגרמה

## **טכנולוגיות בשימוש:**

מערכת זו מבוססת על עיקרון MVC, מחולקת לMODELS – מבנה לוגי של ישויות מסד הנתונים, VIEWS – הגדרת מראה המסכים שיוצגו למשתמש ו CONTROLLERS – שכבה שיוצרת את החיבור בין הנתונים למסך המשתמש, ומכילה את הפונקציונליות המתאימה לכל בקשה שמגיעה מהמשתמש.

## **שפות הפיתוח:**

### בצד השרת: שפת Solidity של איטריום

### בצד הלקוח: שפות javaScript, html5, css3

## **תיאור הארכיטקטורה הנבחרת**

השימוש בטכנולוגית הבלוקצ'יין מבטיח את אמינות הנתונים, ומונע באופן מושלם זיופים וחבלות.

שפתSolidity מאפשרת כתיבת חוזים חכמים בשפה נוחה וקריאה לאדם.

השימוש בספרית React.js המאפשרת פיתוח אפליקציות מתקדמות,והיא טכנולוגיה פופולרית ואהובה בשוק.

ספרית Web3.js היא הספריה הנדרשת לצורך האיטראקציה בין צד הלקוח לרשת איטריום.

## **חלוקה לתכניות ומודולים:**

### **בצד השרת:**

לאיטריום ישנו EVM (Etherium Virtual Machine) - זוהי מכונה וירטואלית אשר מקבלת כקלט את קוד הSolidity , מקמפלת אותו ומפיצה ל Ethereum Networ.

כל החוזים אשר רצים על גבי Ethereum Network מבודדים לחלוטים מהעולם החיצון.

החוזים החכמים מממשים את הפונקציונאליות של המערכת וכל טרנזקציה מריצה את החוזה החכם ונשמרת בשרשרת הבלוקים.

### **בצד הלקוח:**

חלוקה לוגית וארגונית לקומפוננטות כמקובל בספרית React ,

כמו כן, ספרית React עובדת על Virtual DOM , הדום הוירטואלי של React הוא שכל קוד ה HTML נכתב מתוך הJavascript , אין יותר הפרדה בין HTML ל CSS הכל משתלב לקובץ Javascript אחד או לרבים כמותו.

# סביבת השרת

## **ממשק המשתמש** יתקבל כ - HTML ויהיה מוצג באמצעות תוסף לדפדפן אינטרנט

(MetaMask)

## **ממשקים למערכות אחרות:**

## API משרד הפנים – לצורך שליפת נתוני האזרח.

## שימוש בהתקן לסריקת טביעת אצבע לזיהוי מהימן של האזרח .

## **שימוש בחבילות תוכנה:**

שימוש בספרית React.

שימוש בספרית Drizzle של איטריום.

שימוש בספרית Web3 של איטריום.

# מבני נתונים וארגון קבצים

## **שיטת האחסון**

הנתונים נשמרים על רשת הבלוקצ'יין של איטריום (במערכת הנוכחית הרשת היא הסימולטור גנאש).

## **מבני הנתונים**

הקונספט של טבלאות לא קיים, הנתונים נשמרים באוביקטים של סולידיטי לצורך ההפשטה נתאר את המודולים באופן טבלאי, למרות שלמעשה בלוקצ'יין לא תומך במבנה נתונים, כי אם באוביקטים מסוג Json:

* טבלת מפלגות והסכמי עודפים
* טבלת ישובים
* טבלת הצבעות ע"פ ישוב ומפלגה
* טבלת מצביעים (hash table)

## **מנגנוני התאוששות מנפילה/ קריסה/ תמיכה בטראנזקציות**

הרעיון של בלוקצ'יין הוא למנוע סכנות מסוג זה בעצם הוויתו, מתבסס על פונקציונאליות זהירה ביותר, והנתונים מוכפלים לכמות נרחבת של Lids

# תרשימי מערכת מרכזיים

## Use Case





## Sequence diagram





# תיאור המרכיב האלגוריתמי – חישובי

## **חלוקת המושבים בכנסת בין הרשימות המתמודדות**

## חישוב אחוז החסימה מתוך סך הקולות.

## חישוב המודד – שווי מושב בודד בכנסת (מספר הקולות למנדט).

## קביעת מספר המנדטים השלמים לכל מפלגה.

## חחלוקת הקולות העודפים למנדטים על פי נתוני ההסכמים.

# תיאור/התייחסות לנושאי אבטחת מידע

מערכת הבלוקצ'יין מאובטחת ומטפלת בשמירת הנתונים בצורה המהימנת ביותר.

# משאבים הנדרשים לפרויקט:

## מספר שעות המוקדש לפרויקט: 720

## ציוד נדרש: מחשב, חיבור לאינטרנט, סורק טביעת אצבע.

## תוכנות נדרשות: visual Studio Code, Solidity and nodeJS packeges

## ידע חדש שנדרש ללמוד לצורך ביצוע הפרויקט:

לימוד תיאורטי על מגמת הבחירות ועקרונותיה.

## לימוד מכליל על מערכת הבלוקצ'יין ומרכיביה - הלוגיקה, הטכנולוגיה, הטכניקה,

## ההתממשקות אליה, השפות הנפוצות בה והקנית שליטה מלאה בכל כליה.

## הכרות עם ספרית React מהבסיס ועד לתחושת ביטחון ותוצאות מספקות.

## ספרות ומקורות מידע:

## ללימוד :**React** ReactDocs & Scrimba

gitHub להורדת ספריות שונות לדוגמא: String.sol, w3schools.com

## להבנת טכנולוגית **הבלוקצ'יין**: חיפושי רשת וסרטוני וידאו שונים, TruffleSuite, Solidity docs & Truffle docs, DappUniversity Vidoes, blockchaintechnologies.com, investopedia.com, tutorialspoint.com, guru99.com

## לחשיפת התיעוד הפונקציונאלי של**Web3** : Read The Docs

# תכנית עבודה ושלבים למימוש הפרויקט:

|  |  |  |
| --- | --- | --- |
| **שלב** | **משך זמן לעבודה זוגית** | **תאריכים משוערים** |
| * + ייזום הרעיון | **30 שעות** | **1.6-5.6** |
| * + ניתוח מערכת | **50 שעות** | **6.6-12.6** |
| * + ניתוח מבנה נתונים | **20 שעות** | **13.6-18.6** |
| * + אפיון UX - UI | **7 שעות** | **18.6-20.6** |
| * + כתיבת הלוגיקה העסקית | **150 שעות** | **21.6-10.10** |
| * + כתיבת ממשק המשתמש | **40 שעות** | **21.6-10.10** |
| * + עיצוב | **40 שעות** | **21.6-10.10** |

**חתימת הסטודנט חתימת המנחה האישי**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

הערות ראש המגמה במכללה:

אישור ראש המגמה במכללה:

שם:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ חתימה:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ תאריך:\_\_\_\_\_\_\_\_\_\_\_\_\_\_

הערות הגורם המקצועי מטעם מה"ט:

אישור הגורם המקצועי מטעם מה"ט:

שם:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ חתימה:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ תאריך:\_\_\_\_\_\_\_\_\_\_\_\_\_\_