****

מדיניות אבטחת מידע

תחום אפליקציה

Version 2.2

Contents

[כללי 4](#_Toc339220505)

[העבודה מול צוות אפליקציה 5](#_Toc339220506)

[שירותי צוות האפליקציה בשלבי החיים של האפליקציה 5](#_Toc339220507)

[תהליך פיתוח מאובטח 5](#_Toc339220508)

[שלב הארכיטקטורה 5](#_Toc339220509)

[שלב עיצוב מפורט 5](#_Toc339220510)

[שלב הקידוד 5](#_Toc339220511)

[שלב הבדיקות 6](#_Toc339220512)

[שלב הפריסה 6](#_Toc339220513)

[בדיקות אבטחת מידע 6](#_Toc339220514)

[תחזוקה 6](#_Toc339220515)

[בדיקה תיקון ליקויים (רגרסיה) 7](#_Toc339220516)

[עדכון גרסה 7](#_Toc339220517)

[ציוד מיוחד 7](#_Toc339220518)

[נספח א- הנחיות ארכיטקטורה 8](#_Toc339220519)

[חוקיות 8](#_Toc339220520)

[סיווג המידע 8](#_Toc339220521)

[רגישות מידע \* 8](#_Toc339220522)

[הזדהות 8](#_Toc339220523)

[ממשק ניהול 8](#_Toc339220524)

[ממשקי עדכון תוכן 8](#_Toc339220525)

[הפרדת שכבות 9](#_Toc339220526)

[מנגנון ניהול אירועים 9](#_Toc339220527)

[העלאת קבצים \* 9](#_Toc339220528)

[קוד צד לקוח \* 9](#_Toc339220529)

[הפניית לקוח לשירות צד שלישי 9](#_Toc339220530)

[שירותי אינטרנט 9](#_Toc339220531)

[נספח ב' - הנחיות פיתוח מאובטח 10](#_Toc339220532)

[מדיניות סיסמאות 10](#_Toc339220533)

[נעילת משתמשים 11](#_Toc339220534)

[ניהול משתמשים והרשאות 11](#_Toc339220535)

[אימות קלט 12](#_Toc339220536)

[הגנה על מידע רגיש 12](#_Toc339220537)

[הגנה על מידע בתעבורה 13](#_Toc339220538)

[ניהול מופעי משתמשים (Session Management) 13](#_Toc339220539)

[ניתוק שיחה 14](#_Toc339220540)

[שימוש בתעודות והצפנוה 14](#_Toc339220541)

[הגנה על מידע תפעולי רגיש 15](#_Toc339220542)

[ניהול שגיאות 16](#_Toc339220543)

[חיווי ובקרה 16](#_Toc339220544)

[חתימת קבצים וקוד 17](#_Toc339220545)

[CAS 17](#_Toc339220546)

[ניהול הגדרות 18](#_Toc339220547)

[הגנה מפני מתקפות אפליקטיביות 18](#_Toc339220548)

# כללי

מסמך זה בא לתת קווים מנחים בתחום אבטחת המידע בממשל זמין. המסמך מכיל הנחיות שמנוסחות בסגנון dos and don'ts. יש לעיין בו טרם אירוח או פיתוח של יישומים

# העבודה מול צוות אפליקציה

## שירותי צוות האפליקציה במהלך מחזור החיים של האפליקציה

צוות אבטחת המידע של ממשל זמין ישתלב בכל שלבי מחזור החיים של פיתוח תוכנה החל משלבי הייזום ודרך אפיון והתקנה, עד לחשיפת האפליקציה לגישה פומבית של המשתמשים בעולם.

פרק זה בא לפרט את הצורך והן את התוצרים, אותם יספק הצוות בכל שלבי החיים של חיי האפליקציה בממשל זמין.

## תהליך פיתוח מאובטח

תהליך פיתוח המאובטח, הינו תהליך פיתוח תוכנה, המלווה על ידי גורם אבטחת מידע בכל השלבים, החל משלב הייזום, דרך כל שלבי פיתוח התוכנה.

בסיום כל סקר המתבצע על ידי גורם אבטחת המידע מתבצע ניתוח איומים ובסיומו מוגש מסמך ניתוח האיומים למנהל הפרויקט. על מנהל הפרויקט להשיב למסמך ניתוח האיומים תוך זמין סביר

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| שלב | ארכיטקטורה | עיצוב מפורט | קידוד | בדיקות | פריסה | תחזוקה |
| סקר | סקר ארכיטקטורה | סקר עיצוב | סקר קוד | בדיקות אבטחת מידע  QA | סקר פריסה | ליווי השנויים על בסיס הליכי פיתוח מאובטח מקוצרים |
| ניתוח איומים | ניתוח איומי ארכיטקטורה | ניתוח איומי עיצוב | ניתוח איומי סקר קוד | ניתוח איומי פריסה |
| מענה לאיומים | מענה לאיומי ארכיטקטורה | מענה לאיומי | מענה לאיומי סקר קוד | מענה לאיומי פריסה |
| קווים מנחים | \* | \* | \* | \* |

### שלב הארכיטקטורה

בשלב הארכיטקטורה מתבצע מעבר על דרישות המערכת בדגש על מימוש הצורך העסקי. השלב יכיל פירוט הuse cases והאינטראקציה שנדרשת עם משתמשי המערכת ומערכות אחרות. לאחר הצגת הארכיטקטורה, יפיק צוות אבטחת המידע מסמך ניתוח איומי ארכיטקטורה אשר יוגש למנהל הפרויקט .

### שלב עיצוב מפורט

עיצוב תוכנה הינו תהליך מציאת פתרון לשאלות ודרישות העומדות בפני המערכת המפותחת. בשלב זה אחריות כלל הגורמים להחליט go \ no go על המערכת או המוצר. במידה ולמחלקת אבטחת מידע יש סייגים עקרוניים, הם יוצגו בכתב למנהל הפרויקט בשלב זה .

### שלב הקידוד

מערכות שמפותחות על ידי צוות פיתוח של ממשל זמין תפותחנה בצורה מאובטחת. מנהל אבטחת המידע אחראי למנות איש מטעמו אשר ינהל את הפרויקט הפיתוח המאובטח, אשר ילווה את צוות הפיתוח בכל שלב, ויענה על שאלות וצרכי אבטחת מידע מידיים. בהגעה לכל אבן דרך תבוצע בדיקת קוד על ידי איש אבטחת מידע. בתום פעילות זו יוצג מסמך ניתוח איומי סקר קוד למנהל הפרויקט.

### שלב הבדיקות

התקנת המערכת בסביבת ייצור תתבצע על ידי צוות תפעול בלבד. לאחר ההתקנה יבצע צוות תפעול הקשחה של כלל השרתים של הסביבה. יש ליידע את צוות אבטחת מידע בכל חריגה מהבאים:

* נהלי הקשחה סטנדרטיים
* הנחיות אבטחת מידע שניתנו למערכת
* חריגה מ Best Practices

צוות אבטחת מידע יתייחס בכתב לכל אחת מהחריגות שיתגלו במהלך שלב ההתקנה.

### שלב הפריסה

באחריות מנהל הפרויקט לבצע בדיקת קבלה למערכת שמתמקדת בשמירה על כל הפונקציונליות של המערכת לאחר ביצוע ההקשחה. כחלק מפעילות זו יש לבצע בדיקת נגישות של המערכת מכתובות בעולם. כמו כן, לוודא כי כתובות צוות אבטחת המידע הינן נגישות למערכת ולצוות אבטחת מידע יש את כלל ההרשאות והעזרים הנדרשים לגישה תקינה למערכת ב**כל** הרמות ההרשאות.

### בדיקות אבטחת מידע

שלב בדיקות אבטחת מידע יתבצע כשלב ה**אחרון** לפני עליית המערכת לאוויר. בדיקת המערכת תתבצע בסביבת ייצור בלבד, כאשר המערכת במצב מוכן לעלייה לאוויר, ומנהל הפרויקט הכריז על הקפאת התצורה.

1. הכרזה על סיום פיתוח
2. הכרזה על סיום התקנה
3. השלמה מוצלחת של בדיקות קבלה
4. מסירה של תיק מערכת הכולל פירוט ארכיטקטורה
5. מסירה של המערכת לראש צוות אפליקציה במחלקת אבטחת מידע
6. מתן נגישות ברמת תקשורת
7. מתן הרשאות, כולל כרטיסים חכמים אם יש צורך בכך.

### תחזוקה

במהלך חיים רגיל של מערכת מידע, יש לעדכנה מעת לעת. עדכונים אלו נובעים ככלל הן מדרישות פונקציונליות חדשות והן מהתגלותן של תקלות אבטחת מידע חדשות בתשתית ובקוד כאחד. התחזוקה תתבצע על ידי איש תפעול ממשרדי ממשל זמין בלבד.

למען הסר ספק: לא תתאפשר גישה מרוחקת לצרכי תפעול מאתר פיזי שאינו ממשל זמין

### בדיקה תיקון ליקויים (רגרסיה)

בדיקת תיקון לקויים תתבצע על ידי אותו הבודק שביצע את הבדיקה המקורית . הבדיקה תתבצע טרם עליית המערכת לאוור. באחריות מנהל הפרויקט לשריין זמן מתאים לכך.

בדרך כלל זמן הבדיקה יהיה כ 20 אחוז מזמן בדיקה ראשונית שבוצעה במערכת.

### עדכון גרסה

עדכן גרסה של האפליקציה מחייב בדיקה מחודשת על מנת לוודא כי העדכון לא משנה את רמת אבטחת המידע במערכת. יש לתאם את הבדיקה לפי הנוהל של בדיקת תיקון ליקויים.

## ציוד מיוחד

בממשל זמין מתאכסנות גם מערכות, הדורשות ציוד חומרה או תוכנה מיוחדים לצורך עבודתן התקינה, כגון אפליקציות לטלפון נייד או למערכת הפעלה אחרת. באחריות מנהל הפרויקט למסור חומרה, תוכנה וכל פריט אחד טרם תחילת העבודה. כמו כן, מנהל הפרויקט אחראי לבצע הדרכה על הציוד לצוות הבודק.

# נספח א- הנחיות ארכיטקטורה

## חוקיות

המערכת המתאכסנת בממשל זמין תהיה כפופה לחוקי מדינת ישראל והחוק הבינלאומי. בין היתר, יש לשים דגש בעמידתה של המערכת בחוק הגנת הפרטיות של מדינת ישראל ותקנותיו השונות. על המערכת לבצע עדכונים לצורך תאימות לחוק תוך זמן סביר.

## סיווג המידע

סיווג המידע בכלל המערכות המתאכסנות בממשל זמין הינו – **בלתי מסווג** –

## רגישות מידע

עסקי

כל מידע אחר, אשר חשיפתו לציבור אינו רצוי, ואשר לא נופל תחת ההגדרות תפעולי או פרטי.

תפעולי

מידע תפעולי ישמר על פי best practices הנהוגות בסביבה המדוברת ובהתאם להנחיות אבטחת מידע שניתנו למערכת

פרטי

מידע פרטי רגיש יאוחסן בצורה מוצפנת במאגר נתונים בהתאם להנחות פיתוח מאובטח.

## הזדהות

1. משתמשים רגילים

ככלל גישת משתמשי המערכת תהיה ברמה של כתיבה בלבד מזוהה על ידי כרטיס חכם או מערכת המאפשרת סיסמא חד פעמית

1. מנהלי מהערכת

מנהלי מהערכת יזוהו על בסיס כרטיס חכם בלבד. על המערכת להתממשק למערכות הזדהות כרטיס חכם של ממשל זמין או לממש זאת בצורה עצמאית.

## ממשק ניהול

על המערכת לפרסם את ממשק הניהול בכתובת IP שונה מהכתובת שחשופה למשתמשי הקצה. המערכת לא תאפשר גישה למשתמש הקצה להגיע לממשק ניהול.

## ממשקי עדכון תוכן

ממשק עדכון תוכן יונגש ל מעדכני תוכן על בסיס כתובת IP אחרת מהכתובת הפומבית למשתמשי המערכת.

## הפרדת שכבות

על המערכת להפריד בין שכבת הנתונים לשכבת התצוגה.

במידה וגורמי אבטחת מידע דרשו הפרדה של שכבת הלוגיקה העסקית (Business Layer ) במערכת יש להפרידה לשרת נפרד גם כן.

## מנגנון ניהול אירועים

על למערכת מידע להכיל מנגנון ניהול אירועים. מנגנון מסוג זה מועיל במיוחד במקרים של תפקוד לא נכון, שגיאות וניסיונות פריצה. טרם פיתוח המערכת יש לסכם עם צוות אבטחת מידע את סוג, תבנית וכמות האירועים במערכת. צוות אבטחת מידע יאסוף נתונים אלו במערכותיו לצורך חיווי על רמת אבטחת מידע וניסיונות תקיפה אפשריים.

## העלאת קבצים

העלאת קבצים תבוצע על בסיס תשתיות תוכנה והלבנה הקיימות בממשל זמין. על המערכת להתממשק לתשתיות אלו .

## קוד צד לקוח

ככלל, המערכות לא יפעילו קוד בצד הלקוח במטרה אחרת מאשר עיצוב ותצוגה נוחים יותר.  
במידע וקיים צורך עסקי אחר, כגון, הצפנה , חתימה או הפעלת פעילות אחרת הדורשת המצאות של קוד במחשבו של הלקוח, הסיבה תתואר, ותוגש לאישורו של מנהל אבטחת מידע של ממשל זמין.

היה ואישר מנהל אבטחת המידע את הצורך , הקוד יהיה חתום דיגיטאלית ולא ידרוש שינוי בהגדרות אבטחת המידע במחשב המשתמש.

**חריגה מהנחיה זו תהיה באישור מנהל אבטחת מידע של ממשל זמין.**

## הפניית לקוח לשירות צד שלישי

לצורך סעיף זה, מערכות צד שלישי הינן אותן המערכות שניהול אבטחת מידע שלהן אינו ברשות ממשלת ישראל.

כאשר מתבצעת הפנייה ייזומה של הלקוח ל אתר צד שלישי, על המערכת להוציא הודעה ברורה על כך שתוכן האתר אינו באחריות ממשלת ישראל.

כאשר האתר המאוחסן בממשל זמין מבצע embedding של רכיב שמקורו באתר צד שלישי, חלק זה בעמוד ייצבע בצבע אדום בולט עם תוספת של טקסט שמסביר זאת.

## שירותי אינטרנט

שירותי אינטרנט (web services) יעבדו דרך ממשק הDATA POWER ויפותחו על פי תקן ה wsdl.gov.il .

# נספח ב' - הנחיות פיתוח מאובטח

## מדיניות סיסמאות

* הסיסמא לא תעבור גלויה ברשת אלה בצורה מוצפנת \ ב hash או על גבי תווך מוצפן.
* המערכת תספק למשתמש את היכולת להחליף את הסיסמא בעצמו, בצורה בטוחה, בכל עת.
* אורכם של שמות המשתמשים של המערכת יהיה לפחות 8 תווים.
* לא יוגדרו במערכת משתמשים בעלי שם משתמש טריוויאלי, כגון 'admin'.
* המערכת תכיל בדיקה שתוודא ששם המשתמש והסיסמא יהיו שונים זה מזה ולא יכילו אחד את השני (שם המשתמש לא יהיה חלק מהסיסמא ולהיפך).
* סיסמת המשתמש לא תהייה קצרה מ-10 תווים ותהייה מורכבת לפחות מ-3 קבוצות תווים מתוך הארבע הבאות:
  + אותיות קטנות;
  + אותיות גדולות;
  + ספרות;
  + תווים מיוחדים.
* סיסמתו של מנהל המערכת (אדמיניסטראטור) תהייה באורך של 12 תווים לפחות.
* תוקפה של סיסמת המשתמש יפוג כל 60 יום ועל המשתמש יהיה להחליף את סיסמתו בהתאם למבנה המתואר לעיל.
* מנגנון החלפת הסיסמא ישמור היסטוריית הסיסמאות של 5 מחזורים לפחות ולא יאפשר למשתמש לחזור על אף אחת מהסיסמאות הללו בעת החלפת הסיסמא.
* טרם החלפת הסיסמא על המשתמש יהיה להקיש את סיסמתו הנוכחית.
* החלפת הסיסמא לא תתאפשר בטווח של 24 שעות מהחלפת הסיסמא האחרונה.
* הסיסמא הראשונית של המשתמש תהייה רנדומאלית ובהתאם למבנה שהוגדר לעיל.
* המערכת תחייב את המשתמש להחליף את סיסמתו הראשונית בעת ההתחברות הראשונה למערכת.
* תוקף הסיסמא הראשונית יהיה 3 ימים, ולאחר מכן המשתמש ינעל ולא יוכל להשתמש בה.
* במקרה בו המשתמש שכח את סיסמתו, המערכת תיצור לו סיסמא חדשה. כמו הסיסמא הראשונית, סיסמא זו תהייה מוגבלת בתוקף והמשתמש יהיה מחויב להחליפה בעת השימוש הראשון בה.
* יצירת סיסמא חדשה במקרה בו המשתמש שכח את הנוכחית תהייה אך ורק לאחר זיהוי המשתמש באמצעים אחרים, כגון כתובת דואר אלקטרוני, שאלות סודיות וכדומה. מומלץ להשתמש בשילוב של השיטות כגון: שליחת מייל למשתמש עם קישור חד פעמי ומוגבל בזמן המשמש לאיפוס סיסמא, ואחרי שהמשתמש גולש הוא נשאל שאלות בטחון ואם הוא מצליח אז ניתנת לו האפשרות לשנות את סיסמתו.
* אין להציג בשום שלב במחשבי המערכת, במחשבים של משתמשי המערכת, בקוד המקור של דפים וטפסים המועברים למשתמש, את מזהי האימות של המשתמשים השונים במערכת.
* סיסמת המשתמש תשמר בצורת hash בבסיס המידע.
* יש להשתמש באלגוריתמי Hash בטוחים כגון Sha-256 או Sha-512. יש להימנע משימוש באלגוריתמים לא בטוחים כגון Sha-1 ו-MD5.

## נעילת משתמשים

* + נעילת המשתמשים תתבצע לאחר 3 ניסיונות הזדהות כושלים.
  + במקרה של מיעוט משתמשים אין להשתמש במנגנוני שחרור אוטומטי, אלא השחרור יבוצע על ידי מנהל המערכת לאחר קבלת הפניה מהמשתמש ווידוי זהותו.
  + במקרה של מערכת פתוחה לציבור ו/או בעלת משתמשים רבים יש ליישם מנגנון שחרור אוטומטי לאחר פרק זמן קבוע (למשל 15-30 דקות).
  + נעילת המשתמש תתבצע בצד שרת המערכת ולא ברמת ה-Session או ה-Client.
  + משתמש ניהול המערכת לא ינעל לאחר ניסיונות זיהוי כושלים על מנת למנוע מצב של מניעת שירות של המערכת.
  + במקרה של נעילת המשתמש אין להודיע על כך באופן מפורש בכדי למנוע מצב של מיפוי שמות משתמשים בעזרת נעילה מכוונת. ניתן לדווח על הנעילה ישירות לדואר האלקטרוני של המשתמש או להטמיע אותה בהודעת הכישלון בהתחברות הרגילה ("שם משתמש וסיסמא לא נכונים, יתכן והמשתמש ננעל אם ניסית מספר רב של פעמים")

## ניהול משתמשים והרשאות

* הרשאותיהם של המשתמשים יקבעו לפי עקרון ההרשאות המינימאליות הדרושות, כלומר כל משתמש מערכת יקבל את הרשאותיו בהתאם לדרישות עבודתו במערכת ולא מעבר לכך.
* הרשאות המשתמש ייבדקו בכל השכבות ובכל הרכיבים של המערכת.
* יש לבצע בדיקת הרשאות משתמש בכניסה לכל דף במערכת.
* יש לבצע בדיקת הרשאות משתמש טרם ביצוע פעולות במערכת לרבות פעולות צפייה במידע, מחיקה, עדכון או הוספה.
* אין להסתמך על מנגנון זיהוי כמנגנון הרשאות. משתמש מזוהה במערכת אינו בהכרח מורשה לכל חלקיה.
* בקרת הגישה תתבצע בצד השרת בלבד ולא תסתמך על נתונים השמורים במחשבו של הלקוח, לדוגמא cookies

## אימות קלט

* בקרת קלט מהמשתמש תיבדק בשכבות השונות בהתאם לסוג המידע שאמור להתקבל שימוש ב regular expression – white list, קרי, סינון על פי ערכים מותרים ידועים מראש ולא שלילת ערכים, וזאת משום שניתן להציג קלטים ביותר מצורה אחת על ידי שימוש בקידוד שונה.
* הבדיקות יתבצעו גם בצד המשתמש וגם בצד הלקוח.
* מומלץ לבצע אימות קלט בכל אחת משכבות האפליקציה – למשל קוד, מסד נתונים, שכבות Web Services וכו'.
* בגישה ל WS וגישות SOAP באמצעות XML, יש לבצע בדיקות לקלט שמועבר למערכת לפי סכמות XSD מוגדרות מראש לכל פעולה \ מתודה בשרות אליו מתבצעת הגישה, על פי נוהל פיתוח סכמות מאובטחות WS-Gov.il.

## הגנה על מידע רגיש

* יש להצפין נתונים רגישים במערכת. כגון:
  + נתונים רגישים וחסויים של משתמשי המערכת.
  + במידה וקיימים קבצים (כגון קבצי Word, PDF, תמונות וכדומה) אשר מכילים מידע רגיש בבסיס הנתונים יש להצפינם גם כן.
  + נתוני זיהוי של רכיבי תוכנה שונים, כגון נתוני הזיהוי של שרת האפליקציה לשרת בסיס הנתונים (Connection String) וכדומה.
  + מפתח ההצפנה יישמר במקום מאובטח על שרת המערכת, כגון ה-Registry. הגישה למפתח תוגבל לאפליקציה ולאדמיניסטראטור של השרת בלבד.
  + יש לבצע הצפנה של המפתח ע"י שימוש בהצפנת DPAPI, יש לשמור עותק של המפתח במקום מוגן נפרד (פיזי) למקרה שלא ניתן לשחזר את המפתח המקורי.
* אחסון סיסמאות באופן מאובטח תעשה באופן הבא:
  + הסיסמאות אינן דורשות הצפנה דו כיוונית כיוון שאין צורך באחזורם, לפיכך הסיסמאות ישמרו בבסיס הנתונים לאחר ביצוע HASH על ערכן.
  + לכל משתמש בעת יצירת סיסמא ייבחר ערך רנדומאלי אשר ישורשר לסיסמא טרם ביצוע ה-HASH. ערך זה נקרא ערך SALT.
  + ערך ה-SALT ישמר בבסיס הנתונים יחד עם פרטי המשתמש.
  + על מנת לבדוק כי הסיסמא שהמשתמש הזין הינה נכונה, משרשרים אליה את ערך ה-SALT מבסיס הנתונים ומבצעים על הערך החדש את פעולת ה-HASH שבוצעה בעת שמירת הסיסמא. אם ערך ה-HASH החדש תואם את ערך ה-HASH אשר שמור בבסיס הנתונים, הרי שהסיסמא נכונה.
* יש להשתמש באלגוריתמי Hash בטוחים כגון Sha-256 או Sha-512. יש להימנע משימוש באלגוריתמים לא בטוחים כגון Sha-1 ו-MD5.
* יש להשתמש בהצפנות מקובלות כיום בשוק, כגון RSA, ולא לבנות אלגוריתם הצפנה ייחודי למערכת.
* אין לאפשר שמירת נתונים רגישים של המערכת במחשבו של המשתמש.
* יש למנוע את שמירת נתוני המערכת בספריית הקבצים הזמניים ובמנגנוני ה-Cache במחשב המשתמש.

## הגנה על מידע בתעבורה

עקב רגישות המידע והעברתו בתווך אינטרנט, יש להעביר את כלל המידע הרגיש באופן מוצפן, לרבות כל המידע והמסכים המוצגים לאחר הזדהות כלשהי בפני מערכת בתהיל"ה. כל התעבורה תתבצע בתווך מוצפן.

מומלץ להוסיף הגבלות אלו ברמת האפליקציה.

אין חובה להוסיף הגנה באמת התעבורה על אתרים פומביים שאינם מכילים מידע רגיש או הזדהות.

## ניהול מופעי משתמשים (Session Management)

* יש להבטיח כי נתוני session נשמרים בצורה בטוחה במהלך חיי המערכת ובפעולת המערכת השונות המתבצעות עם האובייקטים \ משתמשים.
* יש להבטיח כי קיימת הפרדה בין ניהול הזהויות לבין שימוש ב session כך שלא יתכן מצב כי משתמש שלא ביצע הזדהות יוכל להשתמש ב session פעיל של משתמש שביצע הזדהות כנדרש (גניבת זהות), כלומר יש להבטיח כי המערכת אינה מסתמכת על נתוני session בכדי לאפשר למשתמש חשיפה למידע ופעולות רגישים במערכת.
* יש להשתמש ברכיבי session רק עבור שמירת מצב משתמש בין בקשות http שונות במערכת וכן לצורך ביצוע personalization עבור משתמש.
* אין לשמור מידע רגיש ב SESSION , במידה ונדרש יש לבצע הצפנה של מידע זה.
* בכל מצב שבו נשמר מידע רגיש ב session יש להבטיח כי המידע נשמר בצורה בטוחה ולא תתאפשר גישה אליו שלא דרך מקור מוסמך ומאושר ( כלומר מהאפליקציה שייצרה את המידע) .
* על המערכת להימנע במידת האפשר בשימוש ב   
  client – side state management כגון view state, cookies ,hidden files לצורך קבלת נתונים עבור session .
* האפליקציה תעשה שימוש רק בזהות אשר נתקבלה בתהליך ההזדהות בכניסה לאפליקציה ואשר מבצעת שימוש ב- Session ID ייחודי וזמני.
* יש למנוע ביצוע גישה למערכת ללא SESSION תקין.
* יש למנוע ביצוע גישות מרובות מאותו SESSION למערכת.
* אין להעביר את נתוני הזיהוי של המשתמשים בין מחשב המשתמש לשרתי המערכת, למעט דף הכניסה למערכת.
* יש לקיים מנגנון Idle Timeout אשר יסיים את ה-Session של המשתמש לאחר מספר דקות מוגדר, כ-15 דקות, של חוסר פעילות במערכת.
* יש לקיים מנגנון Session Timeout אשר יסיים את ה-Session לאחר זמן ארוך של פעילות במערכת, כ-8 שעות. מנגנון זה נועד למנוע שימוש במערכת באמצעות סקריפטים וכדומה.
* יש לשקול את ניתוק Session במצבי שגיאה מסוימים.
* ניתוק ה-Session יבוצע על ידי סיום תוקף ה-Session בצד השרת, ולא על ידי העברת הלקוח לדף הכניסה בלבד.

## ניתוק שיחה

* האפליקציה תאפשר יציאה מסודרת ונוחה מהמערכת בכל דף החל מדף הכניסה (Login).
* ניתוק זה יבטיח כי משתמש לא יוכל לבצע שימוש חוזר במערכת ללא ביצוע הזדהות מלאה מחדש וזאת על ידי סגירת ה-Session שלו וכלל המשאבים שהוקצו לו בטרם הניתוק.
* במקרה של זיהוי פעילות חשודה במערכת (כפי שהוגדרה במידול הסיכונים) כגון ניסיונות לביצוע sql injection או הזנת סקריפטים זדוניים בשדות קלט, נדרש לבצע ניתוק כפוי של המשתמש, לבצע רישום ללוג וכן להתריע על כך למנהל המערכת.

## שימוש בתעודות והצפנוה

עבור מידע המוגדר כרגיש, יש לאפשר טיפול באמצעי מידור הן ברמת מנהלי המערכת והן ברמת המשתמש. כולל:

* תמיכה בסוגי מידע שונים.
* יכולת הגדרה במערכי ה-Audit לרישום גישה או ניסיונות גישה למידע המוגדר כרגיש. רישום ה-Audit יבוצע באופן מלא בכל שכבה, ובביצוע האחזור ניתן יהיה להפריד באופן מובהק בין התהליכים ובין השכבות השונות שבהם בוצע ה-Audit.

במערכת מידע נדרש לבצע הצפנה לפי הכללים הבאים:

כאשר מתבצעת הצפנה למידע רגיש יש לממש אלגוריתמי הצפנה לפי הכללים הבאים:

* אין לבצע שימוש באלגוריתמים שפותחו בצורה עצמאית.
* יש לבצע שימוש באלגוריתמים מוכרים כגון:
* AES עבור הצפנה סימטרית
* RSA עבור הצפנה א-סימטרית
* Sha-2, Sha-256 או Sha-512 עבור hash חד כיווני
* עבור יצירת מספרים רנדומאליים יש להשתמש במנגנונים מומלצים על ידי יצרני המוצר או הסביבה.

במקרים של ספק ניתן לפנות לצוות אבטחת מידע לקבלת ייעוץ

## הגנה על מידע תפעולי רגיש

כללי

* על המערכת להימנע משמירת מידע רגיש בקובצי הגדרות, קבצים זמניים, cookies, זיכרון מטמון וכו'. במידה ומידע נשמר במקומות אלו, נדרש לוודא כי לאחר סיום עבודה במערכת מידע שיעורי זה ימחק.

מפתחות הצפנה

* יש לאבטח את מפתח \ מפתחות ההצפנה הנמצאים בשימוש המערכת מפני גישה \ שימוש זדוני ללא הרשאה בהתאם לסוג המפתח – ציבורי \ פרטי.
* יש להגן על המפתח מפני הרס או שינוי בצורה לא מורשת.
* יש לנהל בקרה ודיווח לגבי ביצוע גישות ושימוש במפתחות הצפנה.
* יש להבטיח יכולות שיחזור וגיבוי בשימוש במפתחות הצפנה ( כדי להבטיח שיהיה ניתן לשחזר מידע רגיש שהוצפן עם מפתח שאבד).

התחברות לבסיס הנתונים

* יש להשתמש בזיהוי מבוסס מערכת הפעלה (Windows Authentication) ולהעדיפו על פני זיהוי מסד נתונים במידה והדבר נתמך (למשל ב-SQL Server)
* יש לשמור את פרטי ההתחברות לבסיסי נתונים בקבצי ההגדרות בצורה מוצפנת
* יש להגביל גישה לקבצי ההגדרות קבוצה מינימאלית האפשרית

## ניהול שגיאות

* הודעות שגיאה שיוצגו למשתמש כתוצאה משגיאות המתרחשות באפליקציה יהיו הודעות שאין בהן כדי לחשוף את אמצעי האבטחה במערכת. יש לוודא כי הודעות שגיאה אינם חושפות מידע רגיש בנוגע למבנה המערכת ומשאבי המערכת. הודעות השגיאה שיוצגו יהיו ג'נריות וכלליות.
* הודעות שגיאה שיוצגו למשתמש יהיו הודעות שאין בהן כדי לחשוף את התשתית האפליקטיבית לגרסאותיה השונות כגון: מערכות הפעלה, שרתי web, שרתי אפליקאציה, בסיסי נתונים, פרוטוקולים בשימוש, Web Services בשכבות נמוכות וכדומה.
* אין להציג כל מידע רגיש (כולל : מספרי אשראי, סיסמאות, מפתחות הצפנה וכו') בהודעות שגיאה המוצגות למשתמש.
* כאשר קלט המשתמש אינו מתאים לתבנית הנדרשת בשדה קלט, יש להציג למשתמש הודעת שגיאה המפרט מהי התבנית בה נדרש להשתמש.
* על המערכת לנהל מערך ללכידת שגיאות בזמן ריצה :
  + יש לצפות שגיאות מראש וללכוד אותן בקוד המערכת.
  + בשגיאות שהוגדרו כשגיאות כתוצאה מפעילות הקשורה באבטחת מידע יש לנהוג לפי מה שהוגדר במידול הסיכונים של המערכת, כולל דיווח למנהל המערכת, חסימת משתמש וכו'.
* יש לדאוג לכך שמידע משגיאות יהיה מתועד ע"י המערכת בדפי הlog שלה.
* ניתן להציג למשתמש קוד המתאים לרשומה בלוג לשם טיפול בשגיאה. קוד הרשומה לא ירמוז בשום צורה על קוד השגיאה והסיבה להתרחשותה.
* על המערכת להתמודד עם שגיאות בהיבט של זמינות כך שאם למשתמש מסוים מתרחשת שגיאה הוא אינו חוסם גישה למשתמשים אחרים שמריצים את המערכת ( קריסה כללית ).
* במערכות רגישות ובסיכון בינוני ומעלה, המערכת תכלול יכולת לאחזור הודעות שגיאה (אחזור מלא, אחזור חלקי לפני פרמטרים שונים).
* פרמט הדיווח של הלוגים צריך להתאים לפורמט מערכת SIM \ SOC כך שיהיה ניתן לאסוף את הודעות השגיאה.

## חיווי ובקרה

* האפליקציה תתעד את הנתונים הבאים, במידה והם מוגדרים, עבור כל פעולה במערכת:
  + Timestamp.
  + זיהוי המשתמש (ללא סיסמא)
  + מיקום המשתמש (מחשב/IP).
  + מיקום המשתמש במערכת (מסך, טופס, טבלה וכדומה).
  + פרטים מלאים של הפעולה המבוקשת.
* בנוסף יתועדו הפעולות הבאות:
  + צפייה במידע במערכת.
  + עדכון מידע במערכת.
  + כתיבה ומחיקה של מידע במערכת.
  + כל פעולות הניהול במערכת.
  + כל פעולות הזיהוי במערכת, כולל כישלונות של פעולות אלו והסיבה לכך.
  + כל פעולות ההרשאות במערכת, כולל כישלונות של פעולות אלו.
  + שגיאות מערכת.
  + ועוד, בהתאם לצורך.
* התיעוד יתבצע בשתי שכבות: תיעוד פעולות משתמשי מערכת באפליקציה, תיעוד גישה לנתוני המערכת בבסיס הנתונים.
* חשוב להדגיש כי התיעוד לא יכיל את נתוני הזיהוי של משתמשים או נתונים רגישים אשר שמורים בבסיס הנתונים של המערכת, בדגש על נתונים רגישים.
* כל פעולת תיעוד, בכל הרמות של המערכת, חייבת להכיל את המשתמש המבצע את הפעולה בפועל על מנת למנוע התכחשות משתמשים לפעולותיהם.

**מעקב:**

* יש לוודא כי נתוני התיעוד והמעקב נשמרים באופן מאובטח במערכת.
* יש לוודא כי רישומי התיעוד אינם נגישים למשתמשים ללא הרשאות מנהל מערכת.
* יש לוודא כי נתוני התיעוד מגובים יחד עם שאר נתוני המערכת.
* יש לקיים מנגנון ארכיב לנתוני תיעוד ישנים אשר אינם נחוצים לשם פעולתה התקינה של המערכת.

## חתימת קבצים וקוד

* יש לבצע שימוש ב strong name ולחתום את קוד הפרויקט לאחר יצירת גרסת ייצור יציבה.
* יש להחליף חתימה זאת בכל שחרור של גרסה חדשה.

## CAS

* מומלץ לממש מנגנון CAS במערכת על מנת להגביל את גישת האפליקציה למקורות מידע שלא נדרש לבצע אליהם גישה כגון FTP, Unmanaged code וכו'.

## ניהול הגדרות

* על המערכת לפרט באפיון את כל אמצעי גישות ניהול ההגדרות שבמערכת:
* יש להגדיר תחת איזה חשבון רצה המערכת ( משתמש , מנהל , חשבון מערכת ... ) , הדרישה היא כי המערכת תרוץ תחת חשבון עם רמת הרשאות הנמוכה ביותר הניתנת כך שלא תאפשר ביצוע פעולות לא רצויות ע"י משתמשים רגילים. לא יופעל שום רכיב עם זיהוי SYSTEM ו/או הרשאות ADMIN או מקבילות להם.
* במידה והמערכת דורשת הרשאות גבוהות רק בחלק קטן מהמערכת, יש להריץ את המערכת תחת משתמש נמוך הרשאות ולהתחזות למשתמש בעל הרשאות גבוהות יותר רק בקטע הקוד הרלוונטי.
* יש להגדיר דרכי גישה מאובטחות למשאבים חיצוניים כגון בסיס מידע, מערכת קבצים וכו' (למשל ע"י הצפנת מחרוזת קישור לבסיס מידע).
* יש להגדיר גישה מאובטחת לאדמיניסטרציה במערכת כולל זיהוי חזק והגבלת הגישה למורשים בלבד. יש לשקול הגדרת כתובות IP מסוימות שרק מהן ניתן לגשת לממשק הניהול.
* יש לדאוג לכך שלא יהיה ניתן לחשוף\לגשת למידע רגיש הקיים בקבצי הגדרות למשתמשים ללא הרשאות מתאימות. לשם כך יש להצפין הגדרות רגישות (ברמת האפליקציה ו/או ברמת מערכת הקבצים) ולהגביל אליהם גישה ברמת מערכת ההפעלה.
* אין לשמור מידע רגיש בקבצי הגדרות. במידה ונדרש יש להצפין אותו ע"י שימוש במנגנוני הגנה המומלצים על ידי יצרן מערכת ההפעלה.

## הגנה מפני מתקפות אפליקטיביות

**מניעת התקפות cross site scripting**

יש לבצע בדיקות תקינות בצד השרת על כל הקלט המגיע מצד המשתמש. בדיקת הקלט תכלול את הבדיקות הבאות:

* יש לבדוק את קיומו של הקלט ולא לאפשר הזנת ערכים ריקים.
* יש לבדוק ולהגביל את אורך הקלט (עפ"י האפיון שימוש ברשימות white list וב regular expression לפני הכנסת קלט למערכת).
* יש לבדוק שטיפוס הקלט המתקבל הוא מהסוג המצופה.
* יש לבדוק כי טווח הערכים שמתקבל מתאים להגבלות שנקבעו.
* יש לבדוק את הרכב התווים בקלט, ולוודא שהוא אינו מכיל תווים אסורים. ככלל יש להימנע ככל האפשר מקבלת קלט שאינו מכיל ערכים אלפא נומריים, למעט רווחים.
* יש לוודא כי ערכו של הקלט תואם ללוגיקה העסקית של רכיב היעד.
* יש לוודא כי הקלט ב encoding המתאים למערכת.
* יש להעביר את כלל התווים שאינם אלפאנומריים קידוד HTML בטרם הצגתם למשתמש. תהליך הקידוד יבטיח כי קוד שתול יוצג כטקסט ולא ירוץ על הדפדפן. מומלץ להשתמש בתשתית אפליקטיבית מוכרת (כגון AntiXSS ב-.NET).
* אין להכניס לבסיס הנתונים תווים הנובעים מקלט ישירות לתחום הפעולה של client side scripting (תגי script, אירועי HTML וכדומה).

**מניעת הזרקות SQL**

* אין לאפשר גישה ישירה לבסיס הנתונים. גישה לבסיס הנתונים תתבצע באמצעות שיכבה מתווכת כגון WS או DAL בפרויקט נפרד \ תשתית .
* בכל מקרה יש לבצע סינון מסודר של תווים למניעת הזרקת שאילתות SQL.
* כל תעבורת השאילתות תבוצע ע"י שימוש ב- stored procedures באופן הנכון וללא שימוש בהעברת פרמטרים בקריאה ל- stored procedure.

בגישה לבסיס הנתונים (בשכבות ה-DAL או בכל מקום אחר) יש להשתמש בתשתית האפליקטיבית המתאימה והמומלצת לשם מניעת התקפות SQL Injection:   
ב-.NET: Parameterized Queries, LinQ (Linq to SQL, Entity Framework).  
ב-Java: Prepared Statements.

בכל מקרה יש להימנע משימוש בשרשור מחרוזות בעת הרכבת שאילתות או פקודות מסד הנתונים (לרבות פקודות עדכון נתונים).

**מניעת מתקפות חסימת שירות**

* יש לקחת בחשבון את כלל האיומים העלולים לגרור מתקפות Denial of service (מניעת שירות) ולגבש בקרות כנגדם.
* במנגנון נעילת משתמשים יש לקחת בחשבון את אלמנט הזמינות כך שיהיה ניתן לשחרר משתמש שננעל בצורה מהירה יחסית.

**הגנה מפני buffer overflow**

* יש לאמת פרמטרי מחרוזות כקלט ופלט – יש לוודא את אורך המחרוזת שלא תחרוג מהמקסימום.
* יש לאמת גבולות של מערכים.
* יש לאמת אורך נתיב לקבצים.
* בקוד unmanaged (C, ++C) יש להימנע משימוש בפונקציות שאינן מומלצות כגון strcpy ותחתם להשתמש בגרסאות הבטוחות של אותן פונקציות כגון strncpy (פונקציות מתוך Strsafe.h).

**הגנה מפני מתקפות Network eavesdropping**

* הצפנת תווך תקשורת\ הודעה בזמן ביצוע הזדהות.
* הצפנת תווך תקשורת \ הודעה בזמן העברת מידע הקשור בפרטי זיהוי משתמש כגון החלפת סיסמא וכו'.
* הצפנת תווך תקשורת \ הודעה בזמן העברת מידע עסקי/אישי רגיש.

**הגנה מפני מתקפות Brute force & Dictionary attacks:**

* מימוש מדיניות סיסמאות חזקה.
* מימוש מנגנון נעילת משתמשים.
* שמירת סיסמאות ע"י שימוש ב hash בתוספת ערך רנדומאלי (Salt)

**הגנה מפני מתקפות session hijacking ו session replay.**

* אין לאפשר פתיחה של יותר מ session אחד עבור משתמש (במערכות רגישות בלבד).
* יש לבצע בדיקות אימות ל session לפני מתן גישה כלשהי.
* שימוש לבצע שימוש בתווך מוצפן כדי שלא יהיה ניתן לגנוב cookie המועבר לאפליקציה.
* למניעת מתקפות replay יש ליצור ערך חד ערכי עבור כל הודעה הנשלחת, כמו כן מומלץ לשלב חתימה בגוף ההודעה – timestamp.

**מניעת שמירת נתונים במטמון הדפדפן**

* אופציית אחסון הדפים (Caching) תהיה מבוטלת עבור כל הדפים באפליקציה ולכל סוגי הדפדפנים.

**מניעת חשיפת תוכן תיקיות השרת**

* יש לבטל את מאפיין ה-Directory Listing בכל אחת מהתיקיות הוירטואליות על שרת האפליקציה.

**מניעת אפשרות אחסון פרטי הזדהות בדפדפן**

* יש לבטל אפשרות ה- Password Auto complete ע"י שליחת מאפיין מתאים בתגי ה-Password וה-Form בדף ה-HTML. דוגמא:

***<INPUT TYPE="password" AUTOCOMPLETE="off">***

**מניעת גישה לדפי בדיקות ודפים שאין אליהם קישורים נדרשים באפליקציה**

* יש למנוע גישה לדפי סביבת הבדיקות ולהסיר אותם מסביבת הייצור של המערכת.
* יש לפעול לפי נוהל העברה ליצור מסודר.