תיק פרויקט

Social Hacking Penetration Test

חלק חצי מהדאטא-בייס

מגיש: דוסטונבק איסלמבקוב
מספר ת.ז: 328643358
מורה: אסף אמיר
כיתה: יב'1
תאריך הגשה: 29.5.2019 שנת לימוד תשע"ט
בית ספר: שבח מופת

העבודה נעשתה במסגרת התכנית תעשיידע בליווי חונך מחברת צ'ק פוינט

פרק ייזום:

תיאור כללי

הפרויקט שלנו הוא מערכת שעוזרת למנהלי חברות/בתי ספר לבדוק עד כמה העובדים/התלמידים שלהם מודעים לסכנות באינטרנט ויודעים איך להמנע מליפול למלכודות פישינג שנועדו לגנוב את פרטיהם ולפגוע בהם/בחברה בה הם עובדים.
המערכת מקבלת רשימת מיילים ותבנית מסוימת ושולחת מייל פיקטיבי "זדוני" שמכיל קישור בו מושתלים פרטי האדם שקיבל את המייל, וברגע שהוא נכנס ללינק הזדוני המערכת קולטת את זה ומזינה את זה לתוצאות התרגיל.
המנהל נרשם למערכת שלנו באתר אינטרנטי, מזין רשימת מיילים ל"שליחה חדשה" בוחר את התבנית של המייל הפיקטיבי, שומר את השליחה, ולוחץ על "שלח" בכל רגע שהוא רוצה, ומאחורי הקלעים מתבצעת שליחת המייל וקליטת הנתונים למערכת, שמחזירה למנהל את תוצאות השליחה בזמן אמת.
החלק שלי ושל עוד חבר לצוות הוא החלק של הדאטא-בייס, שאחראי על מה שקורה מאחורי הקלעים, שמירת כל המידע של המנהלים וה"שליחות" שהם מבצעים, שליחת המייל הפיקטיבי, שמירת תוצאות השליחה והחזרה לclient כל מידע שהוא דורש (לאחר שעובר בדיקת אבטחה).

מטרת הפרויקט

לעזור למנהלים לדעת את מצב החברה/ ביה"ס שלהם מול איומי סייבר, ולפעול בהתאם למצב שלהם, אם זה לכנס את כל תלמידי ביה"ס או עובדי החברה ולהסביר להם על הנושא, או להשתמש במערכת שלנו שוב ושוב בצורות שונות על מנת לשפר את מצבם.

קהל היעד

מנהלי חברות, ארגונים, בתי ספר או כל מוסד אחר.

פיצ׳רים ותהליכים עיקריים בפרויקט

1. הרשמה של מנהל חדש.
2. התחברות לאתר.
3. יצירת "שליחה חדשה" עם הזנת רשימת מיילים ובחירת תבנית.

4. לחיצה על שמירה ואז על שליחה.
5. המערכת קולטת שהמשתמש לחץ על שליחת "שליחה" מסוימת ושולח את המייל עם התבנית שנבחרה לרשימת המיילים שהוזנו.
6. קבלת תוצאות בזמן אמת כמה עובדים קיבלו את המייל, כמה נכנסו לקישור וכמה לא והצגה בדיאגרמה.
7. שינוי נתונים של כל משתמש בהתאם לבקשתו.
8. שמירת כל המידע על כל המנהלים וכל השליחות לשימוש מאוחר יותר.

שפת תכנות וכלים:

בצד שלי (הדאטא-בייס) השתמשנו במפרש לשפת JavaScript הנקרא Node.js על מנת שיהיה נוח ליצור את השרת ולהתחבר לדאטא-בייס בשפה פשוטה ומוכרת.

הספריות שהשתמשנו בהן:

- **Express** – הספרייה המרכזית שמאפשרת בקלות לשלוט בבקשות מאתר בעזרת פרוטוקול HTTP.
- **Mongoose** – הנותנת גישה לבסיס נתונים ועל השתלטות עליו כמו שמירת נתונים, קבלתם ושינוים שלהם.
- **Body-parser** – הנותנת אפשרות לשרת להפוך את כל הבקשה לjson.
- **Nodemailer** – הנותנת אפשרות לשלוח מיילים של gmail דרך הקוד עצמו עם הזנת פרטי משתמש, מייל היעד ותוכן המייל, כך יכולנו לשלוח את המיילים בצורה אוטומטית ברגע שהשרת מקבל את הבקשה המתאימה.

הhosting- שהשתמשנו בו בו כדי לעלות אתר הוא **Heroku**. הוא נותן אפשרות בעזרת git תמיד ובכל מקום להוריד את הקוד ולעלות שינויים בחזרה.

סביבת העבודה בה השתמשנו הייתה VisualStuido Code
תוכנת **postman** שימשה ככלי עזר לשליחת בקשות וקבלת תשובות בקלות בזמן פיתוח ושיפוץ השרת.

פרק אפיון

פ'יצרים והתהליכים עיקריים של המנהל - הרחבה

הרשמה לאתר:

- כרגע האתר לא הועלה לרשת, ואפשר לפתוח אותו דרך local host
- לאחר הרצת הקוד ומגיע לעמוד הבית
- בעמוד הבית המשתמש נכנס לכפתור REGISTER ומזין את נתוניו למקומות המתאימים.
- השרת (החלק שלי) מקבל את הנתונים, בודק שהכל תקין, ואם כן שומר את המשתמש ומחזיר תשובה חיובית לclient, אם לא, מחזיר תשובה שלילית לפי הבעיה והחלק של הclient אחראי להחזיר את התשובה למשתמש
- אם הכל תקין הלקוח מועבר לעמוד ההתחברות

התחברות לאתר:

- משתמש מזין את שם המשתמש והסיסמא שלו
- המידע נשלח לחלק של השרת (שלי) שבודק מול הדאטא בייס אם המשתמש קיים
- אם המשתמש קיים הוא מחזיר לclient את הid של המשתמש, ואם לא הוא מחזיר 000, הclient, אם הכל תקין המשתמש מועבר לדף הבית שלאחר ההתחברות, ואם לא הclient נותן למשתמש הודעת שגיאה.

אפשרות לראות ולעדכן את נתוני הלקוח:

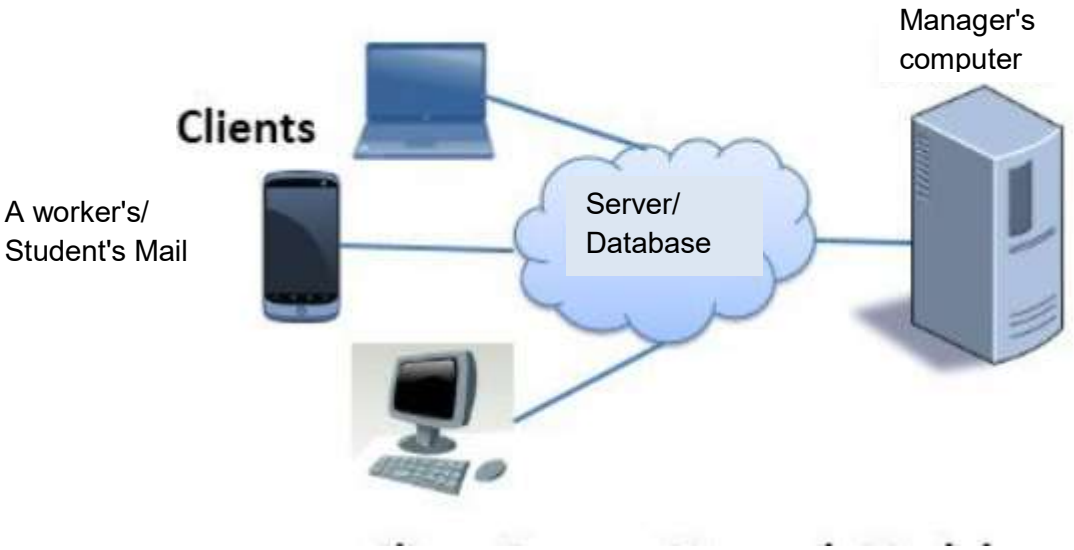
- קיימת באתר לשונית **profile** שבו הלקוח שלנו יכול להכנס לפרופיל שלו, ולעדכן את פרטיו במידה והוא רוצה.

יצירת "שליחה חדשה" ושליחת המייל:

- באתר קיימת לשונית "create experiment" אליה נכנס מנהל החברה/ביה"ס
- לאחר הכניסה ללשונית, המנהל נותן שם לשליחה החדשה אותה הוא רוצה לייצר
- השליחה שנשמרה מוצגת בלשונית אחרת באתר, לשונית בשם "all experiments" ודרך לשונית זו הוא יכול ללחוץ על כפתור edit שעומד ליד השם של השליחה, ולהוסיף את רשימת כתובות המייל אליהם הוא רוצה לשלוח את המייל המזויף
- לאחר הזנת כתובות המייל, המנהל לוחץ על כפתור send כדי לבצע את השליחה

קבלת סטטיסטיקות של שליחה מסוימת:

- בכניסה לשונית "all experiment"
- המנהל יכול לראות את כל השליחות, ולידן כפתורים
- בלחיצה על כפתור statistics המנהל יכול לראות את הסטטיסטיקות של השליחה, כמה אנשים קיבלו את המייל, וכמה מתוכם לחצו על הלינק הזדוני



פ'יצרים והתהליכים עיקריים של האדמין - הרחבה

הרשמה לאתר:

- אדמין מזין את ה-URL ומגיע לעמוד הבית
- הוא נכנס לעמוד REGISTER בעזרת לינק ב-nav-bar ומזין את הנתונים שלו
- שרת מקבל JSON של הנתונים ואם הכל תקין רושם את הלקוח למבנה נתונים – על מנת שהמידע יהיה תקין יש אבטחה מיוחדת על מנת שלא כולם יוכלו ליצור חשבונות כאלו
- אם הכל תקין הלקוח מועבר לעמוד ההתחברות

התחברות לאתר:

- משתמש (אדמין) מזין את שם משתמש וסיסמא
- שרת בודק האם שם משתמש קיים והאם הסיסמא הנשלחת מתאימה לסיסמא הנמצאת במבנה נתונים
- אם הכל תקין שרת שולח ללקוח (אדמין) את האובייקט של הלקוח עצמו עם כל הנתונים שלו ואת כל האופציות – קבלת נתונים על מנהליסת ניסויים וכו'.

לקבל את כל הנתונים על כל המנהלים/תבניות הקיימות:

- במידת הצורך המנהל יכול ללחוץ על כפתור שיציג בפניו את כל המנהלים והמידע שלהם- ובמידת הצורך הוא יכול למחוק אותם או לשנות את פרטיהם.
- במידת הצורך המנהל יכול ללחוץ על כפתור שיציג בפניו את כל התבניות והמידע שלהן- ובמידת הצורך הוא יכול למחוק אותן או לשנות את פרטיהן.
- במידת הצורך המנהל יכול ללחוץ על כפתור שיציג בפניו את כל הניסויים והמידע שלהם- ובמידת הצורך הוא יכול למחוק אותם או לשנות את פרטיהם.

3 קטעי קוד שהיו מאתגרים:

א. פונקציה שאחראית לשלוח את המייל הפיקטיבי לרשימת המיילים של ה"שליחה"

```
exports.sendIt = function(req, res, next){
    CurrentManager.findOne({_id:req.body.idkey}).then(function(data1){
//manager can watch his experiments only
        if(data1) //if it does
        {
            CurrentExperiment.findOne({_id:req.body.id}).then(function(data){
                var x = "";
                for(i=0;i<data.user_ids.length;i++)
                {
                    CurrentUser.findOne({_id:
data.user_ids[i]}).then(function(data2){
                        x = data2.to_email;

                    let transporter = nodeMailer.createTransport({
                        service: 'gmail',
                        secure: false,
                        port: 25,
                        auth: {
                            user : 'dt2001.di@gmail.com',
                            pass: 'bek19941214'
                        },
                        tls: {
                            rejectUnauthorized: false
                        },
                    });

                    let HelperOptions = {
                        from: '"Doston Steinman" <dt2001.di@gmail.com',
                        to: x,
                        subject: 'Hello YESSS',
                        text:
'bagrut.herokuapp.com/api/experiment/urlfind?' + data2._id
                    };

                    transporter.sendMail(HelperOptions, (error, info) => {
                        if(error){
                            return console.log(error);
                        }
                        console.log("The message was sent");
                        console.log(info);
                    });//end of sendmail
                }, function(err){
                    next(err);
                });//end of findOne2
```

```
                }//end of for
            }, function(err){
                next(err);
            });//end of findOne
            res.json({succes: true})
        }
        else if(!data1) // if it does not
        {
            res.json('you are not allowed to use this function');
        }
    });
}
```

ב. פונקציה שמקבלת מערך עם כתובות מייל, יוצרת user חדש עבור כל מייל, ואז שומרת את כל הusers שנשמרו אצל הexperiment הנוכחי:

```
exports.updateUsers = function(req, res, next){
    CurrentManager.findOne({_id:req.body.idkey}).then(function(data1){
//manager can watch his experiments only
        if(data1) //if it does
        {
            CurrentExperiment.findOne({_id:req.body.id}).then(function(data){
                var arr1 = new Array();
                var i = 0;
                var p1, newItem;
                var x = "";
                console.log(req.body);
                var manager_id = data.manager_id;
                var template_id = data.template_id;
                var is_fished = false;
                arr1 = req.body.users;
                for(i =0; i<arr1.length;i++)
                {
                    p1 = {manager_id: manager_id, to_email: arr1[i],
is_fished: is_fished, template_id: template_id};
                    newItem = new CurrentUser(p1);
                    newItem.save(function(err, item){
                        if(err){
                            next(err);
                        }
                    });//end of save
                    //x = newItem.id;
                    //data.user_ids[i] = x;
                    data.user_ids.push(newItem.id);
                    data.last_pos = i;
                }//end of for
```

```
                data.save(function(err, item)
                {
                    if(err){
                        next(err);
                    }
                },
                function(err){
                    next(err);
                })//end of save, end of creating users, saving them and
entering their ids to user_ids.
                //console.log(data);
        //-------------------------------------------------------------------
---------------------------

            }, function(err){
                next(err);
            });//end of findOne2
            res.json({success:true});
        }
        else if(!data1) // if it does not
        {
            res.json('you are not allowed to use this function');
        }
    });
};
```

ג. פונקציה המחזירה רשימה של כל השליחות (experiments), עם הusers manager, template שהID שלהם שמור אצל כל שליחה.

```
exports.getall = function(req, res, next){
    CurrentAdmin.findOne({_id:req.body.idkey}).then(function(data1){
//admins can watch all experiments
        if(data1) //if it does
        {
            CurrentExperiment.find( {}
).populate('manager_id').populate('template_id').populate({path: 'user_ids',
model: 'user'}).then(function(data){
            //populate('user_ids').
                res.json(data)
            },
            function(err){
                next(err);
            }
            );
        }
        else if(!data1) // if it does not
        {
```

```
            res.json('you are not allowed to use this function');
        }
    });
};
```

רפלקציה:

הפרויקט שעשינו כקבוצה והחלק שלי בנפרד, היה פרויקט מאתגר, פרויקט שלימד אותי שפה חדשה, התנסות עם editor חדש שלא הכרתי, שיתוף פעולה בתוך קבוצה בפרויקט במחשבים בפעם הראשונה, שימוש והבנה של כל מיני שפות וinterfaces שונים (דאטא בייס, שרת, Heroku, האתר עצמו front-end וכו...)

היו לנו גם אתגרים, אחד מהם היה התנסות עם שפה חדשה לחלוטין, ואתגר אחר היה להתעסק בפעם הראשונה בנושאים כמו אבטחה, Data-Base וכו...

שיתוף הפעולה, במיוחד עם השותף שלי לחלק של הData-Base והServer עזר לנו להתגבר על אתגרי קוד שונים כשבנינו פונקציות מאתגרות ומסובכות, ואפילו פונקציות קטנות, אבל שמתחברות לתהליך אחד ארוך.

ביבליוגרפיה-

https://appschool.co.il
https://nodejs.org/en/docs/guides/
https://www.w3schools.com/js/
https://javascript.info/

נספחים: