A picture containing graphical user interface

Description automatically generated

**פרויקט שם קליט ומגניב**

**שם:**

**ת.ז.:**

**שם המנחה:**

**חלופה: הגנת סייבר ומערכות הפעלה**

חודש ושנה

**תוכן עניינים – כאן יש רק דוגמא , אבל זוהי החלוקה המומלצת של ספר בפרוייקט את התוכן יש ליצור באופן אוטומטי ב WORD**

**תקציר ורציונל הפרוייקט** 3

**מבוא ורקע כללי** 4

**מטרת הפרוייקט** 6

**שפת התכנות וסביבת העבודה** 7

**ניסוח וניתוח הבעיה האלגוריתמית** 8

**תיאור אלגוריתמים קיימים** 9

**מבנה וארכיטקטורה , הפתרון הנבחר** 10

**מימוש הפתרון בשכלול הקוד עם שפת התכנות** 13

**תיאור המודולים של מערכת התכנה** 19

**תיעוד הקוד** 20

**השוואת העבודה עם פתרונות ויישומים קיימים** 24

**הערכת הפתרון לעומת התכנון והמלצות לשיפורו** 25

**תיאור של הממשק למשתמש – הוראות הפעלה** 26

**מבט אישי על העבודה ותהליך הפיתוח** 28

**ביבליוגרפיה** 29

**קוד התוכנית** 30

**תקציר ורציונל הפרויקט**

תקציר מתוך מסמך הייזום – מה עושה הפרוייקט בגדול

**בחרתי לעשות את הפרויקט על הנושא הזה מכמה סיבות:**

* למה בחרתי לעשות דווקא פרוייקט זה

**מבוא ורקע כללי**

רקע כללי על ה"עולם" שבו עובד הפרוייקט, קצת מושגי ייסוד, המטרה היא שהקורא יבין את מסגרת העבודה שבה יעבוד הפרויקט, לדוגמא:

* תוכנת הגנה על גלישה – התקדמות האנטרנט, גלישה בגילאים שונים, הרצון להגביל גלישה לפי גילאים, התראות וכו'.....
* משחק – עולם משחקי המולטי יוזר, מה קורה היום בעולם זה וכו'...
* תוכנת ניהול כיתה – איך מעבירים שיעור בעידן של תלמיד מול מחשב, מצגות, שליטה כל עבודה של מחשב וכו'
* לימוד מכונה – איך לנצל את המחשב לבצע עבודת חשיבה המחליפה אדם – זיהוי שירים, זיהוי פנים וכו'

**לפחות עמוד שלם** – לא לזרוק הצהרות אלא לפתח אותם, לאור המטרה – הקורא יבין ויכיר את העולם הכללי שבה עובדת התוכנה

**מטרת הפרויקט**

ממש העתק הדבק מתוך מסמך האפיון המפורט:

**מה המוצר המוגמר אמור לבצע:**

טיפה יותר בפירוט מהחלק הראשון

**דרישות מרכזיות:**

זוכרים דרישות פונקציונליות אל מול לא פונקציונליות – כמו במסמך הייזום

**תרחישים במערכת:**

התרחישים מתוך מסמך האפיון

**שפת התכנות וסביבת העבודה**

**שפת התכנות וסביבת העבודה:**

מהן שפות התכנות במסגרת הפרוייקט.

פירוט קצר על כל שפה

למה בחרתי דווקא בהן ולא באחרות

**ניסוח וניתוח הבעיה האלגוריתמית**

כל פרוייקט שלכם יש בו אלגוריתם מסויים ובעיה אלגוריתמית לדוגמא:

* פרוקסי – איך אני מנתב ללא פגיעה במידע, איך אני בודק את המידע שאני מנתב ועוד ועוד
* תוכנת ניהול כיתה: איך אני שולט על מחשב, איך אני מעביר מצגת מאחד לשני ועוד
* משחק מולטי יוזר – איך אני מסנכרן את המשחק בין המשתמשים, איך מאפשר חוויה רציפה ועוד

**יש להדגיש את הבעייה בצורה ברורה**

**תיאור אלגוריתמים קיימים**

אחרי שהוצגו הבעיות – מציגים אלגוריתמים קיימים שפותרים את הבעיה

חלק זה הוא חשוב מכמה סיבות:

* מראה את המחקר שעשיתם לפני שהתחלתם לפתח
* מראה את הסיבוכיות של הפרוייקט ומסביר איך הוא פותר את הבעיה

**הפתרון הנבחר**

הצגתם בעיות ושיטות לפתרון – עכשיו יש להציג את הפתרון שאתם בחרתם בו.

**האלגוריתם הנבחר:**

הסבר על האלגוריתם הנבחר, כולל תרשים זרימה איך הוא עובד במערכת שלכם,

**מבנה המערכת:**

שרטוט של המרכיבים העיקריים במערכת – ברמת שרת לקוח ותקשורת

**מבנה Classes:**

דיאגרמת class , איזה classes עיקריים במערכת ואיך הם מקושרים

**פרוטוקול תקשורת:**

דיאגרמת פרוטוקול כפי שלמדנו, אל מול כל תרחיש שהוצג במסמך האפיון

ההסבר צריך להיות ברור ומפורט – לפחות 5 עמודים

**פיתוח הפתרון בשכלול הקוד עם שפת התכנות**

החלק האחרון של הספר והארוך ביותר, הסבר מפורט של הפתרון ואיך הוא נראה בקוד, כאן מראים class, ומסבירים את הרציונל שלו – לא כל שורה **(את זה עושים בקוד עצמו שחייב להיות מתועד)**

בנוסף מראים את הפונקציות העיקריות והחשובות, כולל הסבר על האלגוריתם שבהן הן עובדות ואיך הוא משוקף למעשה

לדוגמא:

**ניתוב וסידור המידע בין כל המשתמשים:**

למען סידור המשתמשים, אני משתמש בClass של משתמש:

# ----- IMPORTS -----

import threading

from socket import \*

# ----- CONSTANTS -----

PORT\_DEST = 5050

# ----- CLASS -----

class SessionWithClient(threading.Thread):

# ----- DATA -----

#clientSock = None

def \_\_init\_\_(self, Server, clientSock, addr ):

print 5.1

threading.Thread.\_\_init\_\_(self)

print 5.2

self.Server = Server

print 5.3

self.clientSock = clientSock

print 5.4

self.addr = addr

print 5.5

# self.Server = self.proxyServer.connectToServer(addr[0],addr[1])

def fromClient(self):

data = self.clientSock.recv(4096)

return data

def run(self):

request = self.fromClient()

print "From Client :----------------------------"

print request

def SendData(self,data): #step 4

self.clientSock.send(data)

ופעולה המשתמשת בClass כדי לחבר בין המשתמשים:

def WorkWithClients(self):

listenerSock = socket(AF\_INET, SOCK\_STREAM)

listenerSock.bind(LISTENER\_ADDR)

listenerSock.listen(5)

while 1:

clientSock, address = listenerSock.accept()

addr = address[0]

sessionWithClient = SessionWithClient(self, clientSock, addr )

self.open\_clients.append(sessionWithClient)

print self.open\_clients

sessionWithClient.start()

client\_exists = False

for item in self.cdatabase.DB:

if addr == item[0]:

client\_exists = True

if not client\_exists:

self.tree\_data.append((addr, "None"))

self.tree.insert('', 'end', values=(addr, "None"))

self.Send()

כל משתמש שאינו קיים כבר במסד נתונים מתווסף למסד נתונים.

**סריקה של הקבצים ובדיקתם כקבצים מותרים או אסורים:**

ראשית יש רשימה של מMagic Numbers:

MAGICNUM = [("Adobe Illustrator", ".ai", "25 50 44 46"),

("Bitmap", ".bmp", "42 4D"),

("Class", ".class", "CA FE BA BE"),

("JPEG", ".jpg", "FF D8"),

("JPEG 2000", ".jp2", "00 00 00 0C 6A 50 20 20 0D 0A"),

("GIF", ".gif", "47 49 46 38"),

("TIF", ".tif", "49 49"),

("PNG", ".png", "89 50 4E 47"),

("Photoshop", ".psd", "38 42 50 53"),

("Windows Meta File", ".wmf", "D7 CD C6 9A"),

("MIDI", ".mid", "4D 54 68 64"),

("Icon", ".ico", "00 00 01 00"),

("MP3", ".mp3", "49 44 33"),

("AVI", ".avi", "52 49 46 46"),

("Shockwave", ".swf", "46 57 53"),

("Flash", ".flv", "46 4C 56"),

("Mpeg 4", ".mp4", "00 00 00 18 66 74 79 70 6D 70 34 32"),

("MOV", ".mov", "6D 6F 6F 76"),

("Windows Video file", ".wmv", "6D 6F 6F 76"),

("Windows Audio file", ".wma", "30 26 B2 75 8E 66 CF"),

("Zip", ".zip", "50 4B 05 06"),

("GZip", ".gz", "1F 8B 08"),

("Tar", ".tar", "75 73 74 61 72"),

("Microsoft Installer", ".msi", "D0 CF 11 E0 A1 B1 1A E1"),

("Object Code File", ".obj", "4C 01"),

("DLL", ".dll", "4D 5A"),

("CAB Installer", ".cab", "4D 53 43 46"),

("RAR", ".rar", "52 61 72 21 1A 07 00"),

("Executable", ".exe", "4D 5A"),

("Help File", ".hlp", "3F 5F 03 00"),

("PDF", ".pdf", "25 50 44 46"),

("Office Document", ".doc, .xls, .ppt", "D0 CF 11 E0 A1 B1 1A E1"),

("Office Document(2010)", ".docx, .xlsx, .pptx", "50 4B 03 04"),

("Microsoft Database", ".mdb", "53 74 61 6E 64 61 72 64 20 4A 65 74"),

("Outlook Message File", ".msg", "D0 CF 11 E0 A1 B1 1A E1"),

("Jar", ".jar", "50 4B 03 04 14 00 08 00 08 00")]

ובנוסף 2 פעולות אשר סורקות ובודקות את הקבצים:

def match(mnum):

for i in MAGICNUM:

if mnum.startswith(i[2]):

return i

return ("Unknown", "Unknown", "Unknown")

def MagicNumScan(path, find):

pathdir = os.listdir(path)

filelist = []

filelist.append(path)

for i in pathdir:

if os.path.isfile(path +"/" + i):

try:

str1 = open(path +"/" + i,"rb").read()

hexStr=""

if len(str1) > 0:

if len(str1) > 20:

uselen = 20

else:

uselen = len(str1)

for x in range(0, uselen):

if(len(hex(ord(str1[x]))) == 3):

hexStr=hexStr+"0"+hex(ord(str1[x]))+" "

else:

hexStr=hexStr+hex(ord(str1[x]))+" "

newhexStr = hexStr.replace("0x", "")

newhexStr = newhexStr.upper()

filetype = match(newhexStr)

if(filetype[1] == find):

os.remove(path +"/" + i)

filelist.append(i + " is a " + filetype[0] + " and will be deleted")

else:

filelist.append(i + " is a " + filetype[0])

else:

filelist.append(i + " is empty and suspicious")

except:

filelist.append(i + ''' doesn't have "read" permissions''')

else:

filelist.append(MagicNumScan((path +"/" + i), find))

return filelist

**הקשבה למערכות המחשב לייצור קבצים חדשים ובדיקתם:**

ראשית יש סריקה דומה לזאת בפתרון הקודם:

def SpecificMagicNumScan(path, find):

str1 = open(path,"rb").read()

hexStr=""

if len(str1) > 0:

if len(str1) > 20:

uselen = 20

else:

uselen = len(str1)

for x in range(0, uselen):

if(len(hex(ord(str1[x]))) == 3):

hexStr=hexStr+"0"+hex(ord(str1[x]))+" "

else:

hexStr=hexStr+hex(ord(str1[x]))+" "

newhexStr = hexStr.replace("0x", "")

newhexStr = newhexStr.upper()

filetype = match(newhexStr)

if(filetype[1] == find):

os.remove(path)

ולאחר מכן, הClass אשר משתמש בHook שמשתמש בסריקה

class HookChange():

def \_\_init\_\_(self, path):

self.actions = {

1 : "Created",

2 : "Deleted",

3 : "Updated",

4 : "Renamed from something",

5 : "Renamed to something"

}

self.FILE\_LIST\_DIRECTORY = 0x0001

s = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)

s.connect(("gmail.com",80))

self.Own\_IP = (s.getsockname()[0])

s.close()

self.path\_to\_watch = path

self.files\_to\_watch = []

self.hDir = win32file.CreateFile (

self.path\_to\_watch,

self.FILE\_LIST\_DIRECTORY,

win32con.FILE\_SHARE\_READ | win32con.FILE\_SHARE\_WRITE | win32con.FILE\_SHARE\_DELETE,

None,

win32con.OPEN\_EXISTING,

win32con.FILE\_FLAG\_BACKUP\_SEMANTICS,

None

)

def hook(self, args, q):

exec args

files = []

for i in self.DB[0][1]:

files.append(i)

print self.DB

for i in self.DB:

if i[0] == self.Own\_IP:

for j in i[1]:

files.append(j)

for i in files:

self.files\_to\_watch.append(getEnd(i[0]))

while 1:

if not q.empty():

print "bye"

queuedata = q.get()

q.task\_done()

return

results = win32file.ReadDirectoryChangesW (

self.hDir,

1024,

True,

win32con.FILE\_NOTIFY\_CHANGE\_FILE\_NAME |

win32con.FILE\_NOTIFY\_CHANGE\_DIR\_NAME |

win32con.FILE\_NOTIFY\_CHANGE\_ATTRIBUTES |

win32con.FILE\_NOTIFY\_CHANGE\_SIZE |

win32con.FILE\_NOTIFY\_CHANGE\_LAST\_WRITE |

win32con.FILE\_NOTIFY\_CHANGE\_SECURITY,

None,

None

)

for action, file1 in results:

full\_filename = os.path.join (self.path\_to\_watch, file1)

for ftype in self.files\_to\_watch:

try:

SpecificMagicNumScan(full\_filename, ftype)

except:

print "error"

**תיאור המודולים של מערכת התכנה**

win32con.FILE\_FLAG\_BACKUP\_SEMANTICS,

None

)

def hook(self, args, q):

exec args

files = []

for i in self.DB[0][1]:

files.append(i)

print self.DB

for i in self.DB:

if i[0] == self.Own\_IP:

for j in i[1]:

files.append(j)

for i in files:

self.files\_to\_watch.append(getEnd(i[0]))

while 1:

if not q.empty():

print "bye"

queuedata = q.get()

q.task\_done()

return

results = win32file.ReadDirectoryChangesW (

self.hDir,

1024,

True,

win32con.FILE\_NOTIFY\_CHANGE\_FILE\_NAME |

win32con.FILE\_NOTIFY\_CHANGE\_DIR\_NAME |

win32con.FILE\_NOTIFY\_CHANGE\_ATTRIBUTES |

win32con.FILE\_NOTIFY\_CHANGE\_SIZE |

win32con.FILE\_NOTIFY\_CHANGE\_LAST\_WRITE |

win32con.FILE\_NOTIFY\_CHANGE\_SECURITY,

None,

None

)

for action, file1 in results:

full\_filename = os.path.join (self.path\_to\_watch, file1)

for ftype in self.files\_to\_watch:

try:

SpecificMagicNumScan(full\_filename, ftype)

except:

print "error"

המערכת משתמש במודולים שונים על מנת להריץ את האלמנטים השונים בתוכנה.

להלן אפרט כמה ממודולי המערכת:

* MainGUI.py

מחלקה אשר מקשרת בין כל מחלקות השרת וגם יוצרת את הממשק של השרת.

* RuleGUI.py

הממשק של מתן ההרשאות.

* database.py

מודול המכיל את כל הפעולות הקשורות בסריקת הקבצים ובדיקתם.

* ClientDatabase.py

יוצר מבנה נתונים השומר את כל ההרשאות של כל המשתמשים השונים.

* SessionWithClient.py

מחלקה היוצרת את הקשר בין השרת למשתמשים השונים.

* Scrambler.py

מודול המערבל את מבנה הנתונים במטרה לשמור אותו כחסוי.

* threadclass.py

מחלקה שיוצרת את התהליכונים השונים בתוכנה.

* Client.py

הצד של המשתמש.

* HookChange.py

מחלקה אשר עושה את הפניה לAPI על ייצור הקבצים.

**תיעוד הקוד**

**פירוט מבניי הנתונים:**

|  |  |
| --- | --- |
| **שם הרשימה** | **תפקיד המערך** |
| MAGICNUM | מכיל את רשימת מספרי הזיהוי של הקבצים השונים אשר עליה לסרוק. |
| DB | מכיל את הIP של המשתמשים יחד עם הרשאות הקבצים שלהם. |

עתה, אפרט על כל פונקציה שנמצאת בקוד. בטבלאות הבאות (טבלה לכל טופס, מחלקה או עצם של המערכת) יצוין שם הפונקציה, מה היא מקבלת, מה היא מחזירה ומה תפקידה.

**MainGUI.py**

|  |  |  |  |
| --- | --- | --- | --- |
| **כותרת הפונקציה** | **טענת כניסה** | **טענת יציאה** | **תפקיד** |
| def CreateHeader (self) | מקבלת את מחלקת הממשק | - | יוצרת את ראש הממשק |
| def CreateBody(self) | מקבלת את מחלקת הממשק | - | יוצרת את גוף הממשק |
| def CreateButtons(self) | מקבלת את מחלקת הממשק | - | יוצרת את הכפתורים של הממשק |
| def WorkWithClients(self) | מקבלת את מחלקת הממשק | - | מוסיפה משתמשים לטיפול אצל השרת |
| def Send(self) | מקבלת את מחלקת הממשק | - | שולחת את הנתונים למשתמשים |
| def CreateRule(self) | מקבלת את מחלקת הממשק | - | פותחת את הממשק של מתן ההרשאות ונותנת לו את הIP של המשתמש הנבחר |

**RuleGUI.py**

|  |  |  |  |
| --- | --- | --- | --- |
| **כותרת הפונקציה** | **טענת כניסה** | **טענת יציאה** | **תפקיד** |
| def CreateHeader(self) | מקבלת את מחלקת הממשק | - | יוצרת את ראש הממשק |
| def CreateBody(self) | מקבלת את מחלקת הממשק | - | יוצרת את גוף הממשק |
| def CreateButtons(self) | מקבלת את מחלקת הממשק | - | יוצרת את הכפתורים של הממשק |
| def Add(self) | מקבלת את מחלקת הממשק | - | מוסיפה מרשימת הקבצים האופציונאליים אל רשימת הקבצים הנבחרים |
| def Remove(self) | מקבלת את מחלקת הממשק | - | מסירה מרשימת הקבצים הנבחרים ומחזירה לרשימת הקבצים האופציונאליים |
| def Ok(self) | מקבלת את מחלקת הממשק | - | סוגרת את החלון ומעדכנת את מבנה הנתונים DB |

**database.py**

|  |  |  |  |
| --- | --- | --- | --- |
| **כותרת הפונקציה** | **טענת כניסה** | **טענת יציאה** | **תפקיד** |
| def match(mnum) | מקבלת מספר זיהוי MagicNumber | מחזירה את סוג הקובץ | מגלה את סוג הקובץ לפי הMagicNumber שלו |
| def getEnd(name) | מקבלת שם של סוג של קובץ | מחזיר את הסיומת של הקובץ | מביאה את הסיומת של סוג הקובץ לפי שם סוג הקובץ |
| def SpecificMagicNumScan(path, find) | מקבלת מיקום של קובץ, וסוג קובץ | - | בודקת אם הקובץ מותר ומוחקת אותו אם לא |
| def MagicNumScan(path, find) | מקבלת תיקיה וסוג קובץ | - | סורקת את התיקייה ובודקת כל קובץ אם הוא מותר ומוחקת אותו אם לא |
| def printTreeDir(li, count) | מקבלת רשימת log על הסריקה | - | מדפיסה את log הסריקה בצורת עץ |

**ClientDatabase.py**

|  |  |  |  |
| --- | --- | --- | --- |
| **כותרת הפונקציה** | **טענת כניסה** | **טענת יציאה** | **תפקיד** |
| def updatedatabase(self, new\_list) | מקבלת את מחלקת הממשק ורשימה חדשה של מבנה נתונים | - | מחליפה את מבנה הנתונים הקיים במבנה חדש |

**SessionWithClient.py**

|  |  |  |  |
| --- | --- | --- | --- |
| **כותרת הפונקציה** | **טענת כניסה** | **טענת יציאה** | **תפקיד** |
| def fromClient(self) | מקבלת את המחלקה | מחזירה את הנתונים שקיבלה מהמשתמש | מקבלת מידע מן המשתמש |
| def run(self) | מקבלת את המחלקה | - | מתחברת למשתמש |
| def SendData(self,data) | מקבלת את המחלקה ונתונים לשליחה | - | שולחת את המידע המבוקש למשתמש |

**Scrambler.py**

|  |  |  |  |
| --- | --- | --- | --- |
| **כותרת הפונקציה** | **טענת כניסה** | **טענת יציאה** | **תפקיד** |
| def Decode(self) | מקבלת את המחלקה | מחזירה את מבנה הנתונים לאחר פענוח | מפענחת את המבנה המוצפן |
| def Encode(self, new\_data) | מקבלת את המחלקה, ומבנה חדש | - | מצפינה את המבנה החדש ושומרת אותו |

**threadclass.py**

|  |  |  |  |
| --- | --- | --- | --- |
| **כותרת הפונקציה** | **טענת כניסה** | **טענת יציאה** | **תפקיד** |
| def run(self) | מקבלת את המחלקה | - | מפעילה את התהליכון של המחלקה |

**Client.py**

|  |  |  |  |
| --- | --- | --- | --- |
| **כותרת הפונקציה** | **טענת כניסה** | **טענת יציאה** | **תפקיד** |
| def scan(filetypes) | מקבלת רשימה של הקבצים אשר יש לסרוק | - | קוראת לפעולת הסריקה מ-database.py |
| def listen(args) | מקבלת את הערכים הנתונים על ידי התהליכון | - | קוראת לפונקציה scan עם הערכים ומחזירה לשרת תשובה על סיום התהליך |
| def (args) | מקבלת את הערכים הנתונים על ידי התהליכון | - | קוראת לפעולת הhook מ-HookChange.py |

**HookChange.py**

|  |  |  |  |
| --- | --- | --- | --- |
| **כותרת הפונקציה** | **טענת כניסה** | **טענת יציאה** | **תפקיד** |
| def hook(self, args, q) | מקבלת את המחלקה, את הערכים של הקבצים, ותור(לבדיקת יציאה) | - | עושה את הפניה API על התיקייה הנדרשת, לשינויים וקוראת לפונקציית הסריקה כדי לבדוק את הקבצים אשר השתנו |

**השוואת העבודה עם פתרונות ויישומים קיימים**

השוואה לפתרונות קיימים מה ההבדל בסופו של דבר...

**הערכת הפתרון לעומת התכנון והמלצות לשיפורו**

מה הצלחתם בסוף, האם אתם מרוצים ומה עוד כדאי לשפר בפרוייקט בעתיד

**תיאור של הממשק למשתמש – הוראות הפעלה**

**הסבר – איך עובדים עם התוכנה , זה ממש manuel של הפרוייקט שלכם**

**לדוגמא:**

\* הוראות ההפעלה נכתבו בלשון זכר בשל טעמי נוחות בלבד.

שלום מנכ"ל נכבד,

באמצעות התוכנה הזו תוכל לצמצם את הסחות הדעת של העובדים שלך בכך  
שתבחר אלו קבצים תיתן להם לשמור על המחשב ואלו לא.

תרצה להתקין את תוכנת השרת על השרת כך שיישאר דלוק בין אם המחשבים האחרים דלוקים ובין אם לא. בנוסף לכך תרצה להתקין ולהפעיל את השרת לפני שתתקין ותפעיל את המשתמשים.

כאשר השרת מותקן ומופעל, יפתח חלון בעל רשימה של המחשבים המחוברים וככל שמתחברים עוד מחשבים, הרשימה גדלה.  
תוכל לעשות 2 הגדרות:

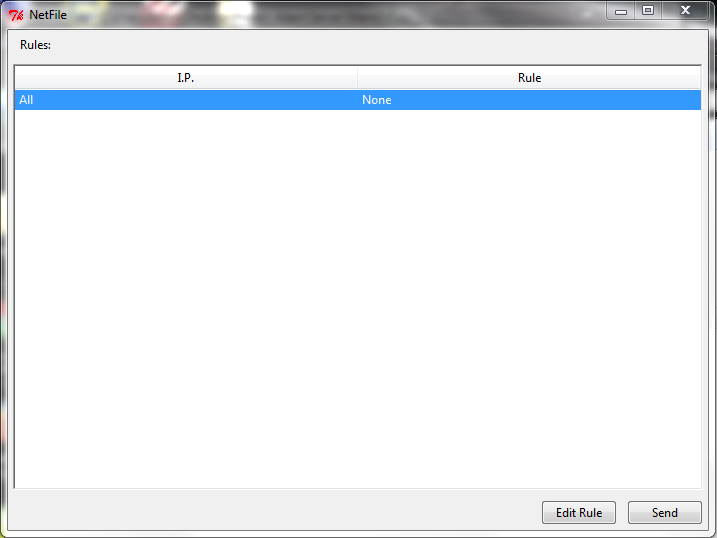
* כדי להגדיר הרשאות על כל המחשבים, תבחר באפשרות ALL ותלחץ על הכפתור Edit Rule.
* כדי להגדיר הרשאות על מחשב ספציפי, תבחר את הIP שלו ותלחץ על הכפתור Edit Rule.

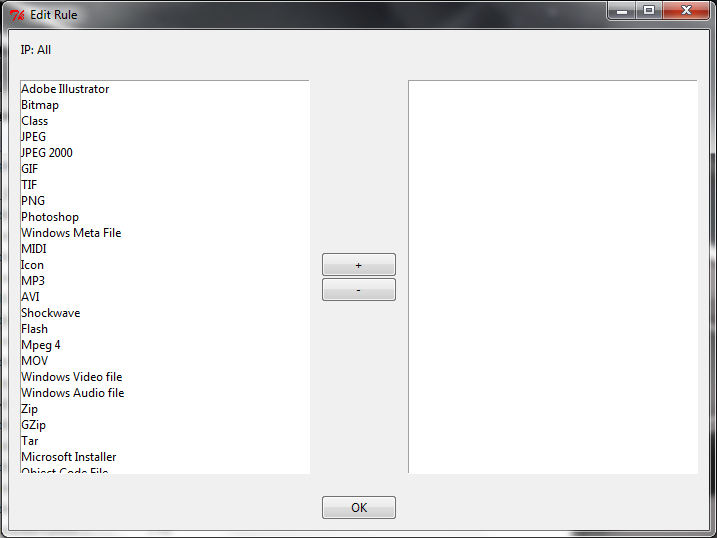
כאשר תלחץ על Edit Rule, יפתח חלון עם אפשרויות איסור. תלחץ על אפשרות ועל כפתור ה'+' כדי להעביר את האפשרות לרשימת הקבצים הנבחרים. תלחץ על כפתור ה'-' כדי להסירו מרשימת הקבצים הנבחרים ולהחזירו לרשימת האפשרויות. כאשר סיימת לבחור הרשאות, לחץ על כפתור ה'Ok'.

כאשר סיימת לבחור הרשאות לכל המחשבים כרצונך, לחץ על כפתור ‘Send’ כדי לשלוח את הנתונים למשתמשים, והתוכנה תעשה את השאר בשבילך.

בהצלחה.

ממשק השרת:





**מבט אישי על העבודה ותהליך הפיתוח**

איך ראיתם באופן אישי את הפיתוח מה לקחתם איתכם ומה הייתם מוותרים

**ביבליוגרפיה**

חומר חיצוני שהיה בשימוש, ספרים, אתרים, מאמרים, כתבות בעיתון ועוד.

**קוד התוכנית:**

**Copy paste of the code**