מדריך למשתמש

1. התקנת התוכנה

הגרסה הארוזה של התוכנה זמינה כ-git repository באתר GitHub תחת הכתובת:
https://github.com/tfiling/FinalProjectWrapping
בדף הנ"ל מצוינות הוראות ההתקנה של התוכנה.

קוד המקור זמין כ-git repository גם כן, תחת הכתובת:
https://github.com/miamiHits/FinalProject

2. שינוי קונפיגורציה

קיימים שלושה סוגים של הגדרות עבור הפרויקט אותן ניתן לערוך בשלושה מקומות שונים:

- קובץ SHAS/start.ini - קובץ זה אחראי על הגדרות המשפיעות על הרצת שרת ה-jetty המקומי אשר עוטף ומריץ את הסימולטור. קובץ הגדרות זה מאפשר רמת פירוט גובהה של הגדרות השרת הנ"ל. ההגדרות היחידות אשר בוצעו עבור השרת הנן קביעת ה-port דרכו יש לגלוש לאתר(תחת http module בקובץ) ופתיחת האפליקציה ל-debugging (תחת החלק "configure JVM arguments"). השארנו את הקובץ עם כמות גדולה של הגדרות אפשריות מבוטלות תחת הערה במטרה להנגיש מאפיינים אלו. יש לשים לב ששינוי הגדרות אלו יכול לשבש את ההפעלה של התוכנה מתוך הקבצים: runSHAS.bat, runSHAS.sh ולכן יש לבצע שינויים בקבצים אלו בהתאמה לשינויים בקובץ start.ini.
<u>הערה חשובה</u> - מומלץ לשנות הגדרות אלו רק אם אתם בקיאים בהגדרות הפרויקט, במבנה שלו ובדרך שבה הוא רץ. כמו כן אנו ממליצים לתעד כל שינוי הגדרות במידה וידרשו פעולות שחזור לאחור.
- קובץ SHAS/resources/conf.properties - קובץ זה אחראי להגדרות הריצה של התוכנה עצמה (בנפרד מהגדרות השרת). הגדרות אלו נטענות בעת פתיחת לשונית חדשה בדפדפן אל כתובת האפליקציה ולכן כל שינוי בקובץ בא לידי ביטוי בצורה מיידית. קובץ זה מכיל שורות מהצורה הבאה:
<profile>-<property name> = <value>
ניתן להוסיף ולערוך מספר פרופילים של הגדרות כאשר הפרופיל שנטען בפועל הוא זה שמצוין במאפיין הראשון "mode". על כל פרופיל להכיל את אותן ההגדרות בשני הפרופילים אשר סופקו: prod ו-dev. פרופיל "dev" מותאם להפעלת התוכנה מתוך IDE בעזרת קוד המקור (קיצור של development). פרופיל "prod" מותאם להפעלת התוכנה בדרך המתוארת בחלק "הפעלת התוכנה" (קיצור של production). ניתן להוסיף מאפיינים חדשים לקובץ זה בהתאם לשינוי בקוד המקור אשר יטען את אותם מאפיינים חדשים. ניתן למצוא הסבר מפורט בחלק "תחזוקת המערכת".
- קובץ log4j.properties - קובץ זמין אך ורק בקוד המקור של התוכנה ומוטמע בתוך הקובץ המכיל את התוכנה עצמה (SHAS/webapps/SHAS.war). קובץ זה אחראי על הגדרות הקשורות אל מנגנון ה-log של המערכת. ברירת המחדל היא הדפסת הודעות log, ברמת תעדוף DEBUG ומעלה, אל המסך ואל קבצים בתיקייה SHAS/log. תוכלו לגשת לקוד המקור ולבחון את ההגדרות אל מול ההערות המסבירות אותן. במידה ותרצו לשנות הגדרות אילו יהיה עליכם לארוז מחדש את התוכנה לקובץ war חדש כפי שמוסבר בחלק "תחזוקת המערכת" שכן קובץ זה מוטמע בקובץ הwar העוטף את התוכנה.

3. הפעלת התוכנה

התוכנה מופעלת בעזרת שרת jetty מקומי אשר גלישה אליו (עם ה-port המוגדר), בעזרת דפדפן, מספקת את ממשק המשתמש של התוכנה.
על מנת להפעיל את התוכנה במערכת הפעלה windows יש להפעיל את הקובץ runSHAS.bat (לחיצה כפולה על הקובץ).
על מנת להפעיל את התוכנה במערכת הפעלה linux יש להפעיל את הקובץ runSHAS.sh מתוך ה-terminal.
כדי לעשות זאת עליכם:

1. לפתוח terminal
2. לנווט אל התיקייה הראשית בעזרת פקודות "cd"
3. במידה ואין הרשאות הרצה עבור הקובץ יש להגדיר אותן בעזרת הפקודה "chmod +x runSHAS.sh" מתוך התיקייה הראשית (יש להשתמש בפקודת sudo במידת הצורך)
4. להריץ את הקובץ בעזרת הפקודה: "./runSHAS.sh"

לאחר הרצת הקובץ, ניתן להשתמש בתוכנה דרך ממשק המשתמש בדפדפן (באמצעות פניה לכתובת localhost:<xxxx> - כאשר במקום <xxxx> יש להכניס את הport הנבחר, 8888 הוא הדיפולטי).

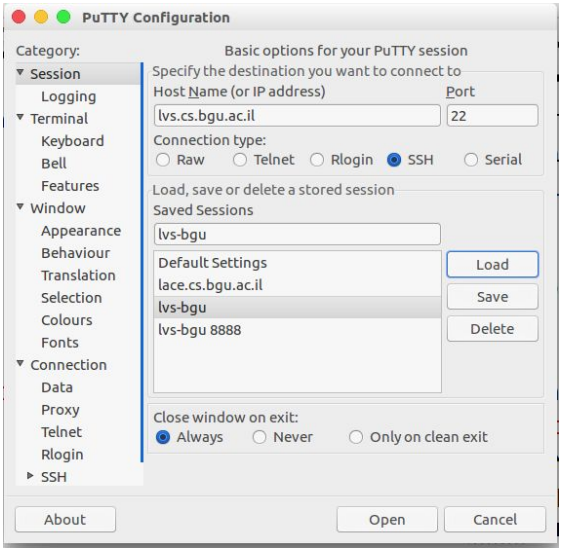
מומלץ להשתמש במערכת בעזרת הדפדפן google chrome, ולמען הנוחות מומלץ להגדיר אותו כדפדפן ברירת המחדל.

הערה חשובה - יש לסגור את התוכנה אך ורק בעזרת פקודת kill בחלון ההפעלה (חלון ה-cmd ב-windows וחלון ה-terminal שממנו הופעלה התוכנה במערכת הפעלה linux). סגירת החלונות בלבד לא תסגור את התוכנה אשר תמשיך לרוץ ברקע.

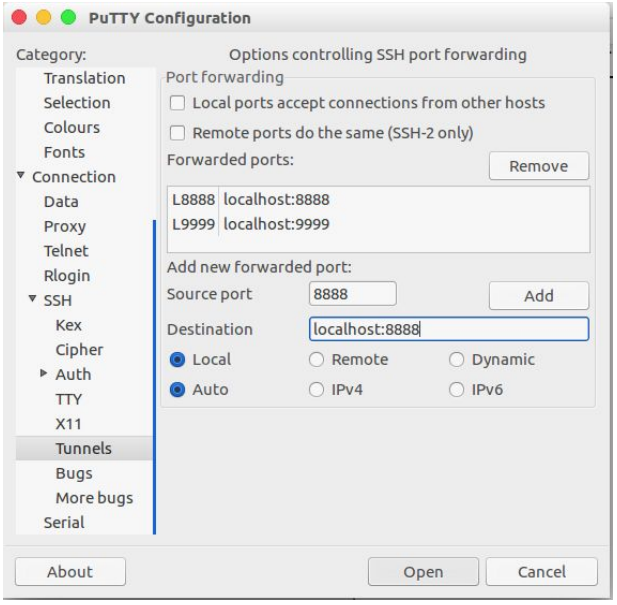
הרצה של המערכת על שרת מרוחק:

במידה וברצונכם להפעיל את המערכת על גבי שרת linux פנימי של האוניברסיטה, ראשית עליכם להחזיק בכתובת הIP של אותו שרת ובשם משתמש (וסיסמה) בעל הרשאות sudo.
עליכם להוריד ולהתקין putty אל המחשב בו אתם משתמשים ולבצע את ההגדרות הבאות(מומלץ לשמור אותן ברגע שעובדות):

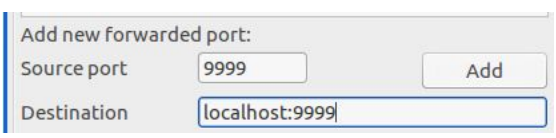
- במסך הנ"ל יש להזין את כתובת ה-IP, לבחור SSH ולהגדיר Port להיות 22:

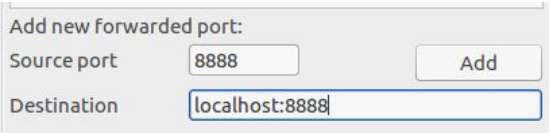


- הגדרת SSH Tunnels אשר דרכן יהיה ניתן לגלוש אל ממשק המשתמש ואף לבצע debugging עבור המערכת:



ברשימה יש את שתי ההגדרות שצריך לעשות ובשני השדות יש ערכים לדוגמא בשביל פורט 8888, כלומר אלו שתי ההגדרות אשר צריך לבצע (יש ללחוץ על Add כדי להחיל את ההגדרות):





כעת, לאחר שהגדרתם ופתחתם חיבור SSH עם SHH Tunnels מתאימים, עליכם לשכפל את המערכת אל השרת מן ה-repository שלו ב-GitHub (ע"פ ההוראות בפרק 1 - "התקנת התוכנה").

לאחר מכן כל שנותר הוא להתקין על השרת java. במידה וברצונכם להוסיף למערכת הרצה על גבי השרת אלגוריתמים נוספים עליכם להתקין JDK בגרסה 1.8 ומעלה ולהגדיר את המיקום של javac בקובץ conf.properties (מתואר בפירוט בדף הפרויקט בGitHub).
לאחר הרצת המערכת בעזרת script ההפעלה runSHAS.sh גלישה אל הכתובת:
localhost:8888/SHAS במחשבכם האישי (כאשר החיבור אל השרת בעזרת Putty עדיין פעיל)
תנווט אל ממשק המשתמש של המערכת המותקנת על שרת ה-linux.

הערה חשובה - שימו לב כי ה-IP של השרת אכן זמין מן הרשת אליה אתם מחוברים. לרוב, שרתים פנימיים של אוניברסיטת בן גוריון זמינים אך ורק מתוך הרשת של האוניברסיטה.

4. הרצת ניסוי

לאחר הפעלת התוכנה כפי שמוסבר בסעיף 3, יעלה העמוד הראשי.
בעמוד זה ניתן לבחור את האלגוריתמים והבעיות שישתתפו בניסוי, כמו גם את מספר האיטרציות שירוצו כל האלגוריתמים בניסוי.

בצד ימין מופיעים האלגוריתמים אשר נמצאים במערכת, הוספת אלגוריתם לניסוי תתבצע ע"י לחיצה כפולה על שמו, או לחילופין סימונו ולחיצה על החץ הפונה ימינה ונמצא לידו. ניתן גם לסמן כמה אלגוריתמים ביחד - באמצעות כפתור control - ולהעבירם יחדיו. ניתן גם להכניס את כל האלגוריתמים באמצעות שימוש בכפתור Add all.
הסרת אלגוריתם שנבחר תתאפשר ע"י לחיצה כפולה על שמו תחת selected algorithm.

בצד שמאל של המסך מופיעות רשימת התיקיות של הבעיות מחולקות ע"פ גודלן. לחיצה על שם התיקייה תכניס את כל הבעיות שבה לניסוי, לחיצה על החץ לצד שם התיקייה תפתח את רשימת תתי התיקיות - שם תיקייה הוא מספר הבתים החכמים שבבעיות שבה. באופן דומה, לחיצה על שם התיקייה תכניס את כל הבעיות בגודל זה לניסוי, בעוד לחיצה על החץ תציג את כל הבעיות בגודל זה לצורך בחירת חלק מהם. ניתן גם להכניס את כל הבעיות באמצעות שימוש בכפתור Add all.
על מנת להסיר מהניסוי בעיה שנבחרה יש ללחוץ על שם הבעיה תחת הטבלה של - selected problems.

בתחתית העמוד יש להכניס מספר איטרציות שבהם ירוצו כלל האלגוריתמים.

לאחר לחיצה על כפתור Start Experiment הניסוי מתחיל ומגיעים לעמוד ריצת הניסוי.
בעמוד זה ניתן לראות את התקדמות ריצת הניסוי. במידה וברצונך להפסיק את ריצת הניסוי לפני סיומה, ניתן ללחוץ על כפתור Stop Experiment בתחתית העמוד ולחזור לעמוד הראשי.
5. צפייה בתוצאות ניסוי

לאחר שכלל הבעיות סיימו את ריצתן, תתאפשר לחיצה על כפתור go to results screen - שיעביר אותנו למסך התוצאות בו יופיעו כל 6 הגרפים, כולל מקרא גרף לכל אחד מהם. ניתן להוריד ולהחזיר את ה error bars מהגרפים ע"י לחיצה על כפתור With/Without error bars בתחתית העמוד.
הגרפים המוצגים (משמאל למעלה ועד ימין למטה):

1. **Total grade per iteration** - ציון כללי ממוצע על פני כל הבעיות בניסוי לכל אחד מהאלגוריתמים שבניסוי בכל איטרציה. לכל אלגוריתם קיימת גם עקומה anytime השומרת את הציון הממוצע הטוב ביותר של האלגוריתם עד איטרציה זו.
2. **Cheapest agent by iteration** - עלות מינימלית בין הסוכנים בממוצע על כל הבעיות בניסוי, לכל אחד מהאלגוריתמים, בכל איטרציה.
3. **Most Expensive agent by iteration** - עלות מקסימלית בין הסוכנים בממוצע על כל הבעיות בניסוי, לכל אחד מהאלגוריתמים, בכל איטרציה.
4. **Average messages size (Byte) per algorithm** - גודל הודעה ממוצעת בין הבעיות בניסוי לכל אחד מהאלגוריתמים שבניסוי, בכל איטרציה.
5. **Average number of messages per algorithm** - מספר ההודעות שנשלחו בממוצע על פני כל הבעיות בניסוי, לכל אחד מהאלגוריתמים שבניסוי, בכל איטרציה.
6. **Average run time per iteration** - זמן ריצה ממוצע על פני הבעיות בניסוי, לכל אחד מהאלגוריתמים שבניסוי, בכל איטרציה.

בסיום ריצת ניסוי נשמר בצורה אוטומטית דו"ח המפרט את מהלך הניסוי תחת התיקייה SHAS/reports. הקובץ הנ"ל מקבל שם מהמבנה: <*date*>_<*time*>_results.csv

6. הוספת אלגוריתם

לצורך הוספת אלגוריתם למערכת תחילה יש לממש מחלקה אשר יורשת מהמחלקה האבסטרקטית SmartHomeAgentBehaviour.
למען נוחיות התחזוקה, כלל האלגוריתמים מלאים בהערות המסבירות את מהות כל פונקציה ודוגמת מימוש מפורטת במיוחד תוכלו למצוא באלגוריתם **Simulated Annealing** בכתובת:
https://github.com/miamiHits/FinalProject/blob/master/src/main/java/FinalProject/BL/Agents/SA.java

מהלך הרצת האלגוריתם בקוד

לאחר תחילת הרצת ניסוי, כל אחד מהבתים בבעיה הנתונה יתורגם לסוכן (jade agent) מסוג האלגוריתם אותו בחרת, לדוגמה, אם נבחר אלגוריתם SA הרי שכל סוכן בבעיה יהיה מסוג SA.java שממש בתורו את **SmartHomeAgentBehaviour**.

1. מחלקת **Expirement** תיצור כל אחד מהסוכנים ותפעיל אצלם את הפונקציה: initializeBehaviourWithAgent הממומשת במחלקת **SmartHomeAgentBehaviour** ותפקידה לבצע אתחול לשדות הסוכן.
2. בשלב הבא תיקרא הפונקציה **action** אף היא ממומשת במחלקת **SmartHomeAgentBehaviour** ותפקידיה:
   a. ביצוע הפונקציה **doIteration** <u>אותה יש לממש</u> במחלקה של האלגוריתם שברצונך להוסיף. בפונקציה זו יתבצע כל תוכן האיטרציה, ליבו של האלגוריתם, (למעט שליחת הודעה לסוכן האיסוף והודעות על תוצאות סיום האיטרציה - אלו כבר נשלחות בהמשך).
   b. שליחת הודעה לסוכן האיסוף על תוצאת האיטרציה - כדי שהודעה זו תישלח ותעובד בהצלחה, יש למלא את השדה **agentIterationCollected** שבמחלקה **SmartHomeAgentBehaviour** .
   c. שליחת הודעות לכלל השכנים של הסוכן (בית) הנוכחי - לצורך שליחה מוצלחת של הודעה זו יש למלא את השדה **currIteration** שבמחלקת **SmartHomeAgent**.
3. בסיום הריצה - כאשר הסוכן יסיים את האיטרציה האחרונה שלו - שדה **currentNumberOfIter** - תיקרא הפונקציה **done** הממומשת במחלקה **SmartHomeAgentBehaviour** אשר הורגת את הסוכן הנוכחי ומפעילה את הפונקציה **onTermination** שעליך לממש במחלקת האלגוריתם החדש. בפונקציה זו תוכל לכתוב logs או לבצע כל פעולה אחרת שברצונך לעשות לפני הריגת הסוכן.

נקודות חשובות במימוש:

- בכל איטרציה על כל סוכן בבעיה לשלוח **<u>הודעה אחת בדיוק</u>** לסוכן האיסוף, הודעה זו הנה מסוג IterationCollectedData והיא נשלחת אוטומטית בסוף כל איטרציה כחלק מהמימוש של המחלקה האבסטרקטית SmartHomeAgentBehaviour, ועל כן **אין** לשלוח הודעות לסוכן האיסוף מתוך קוד האלגוריתם החדש. כדי שהודעה זו תישלח ותעובד בהצלחה, יש

למלא את השדה **agentIterationCollected** שבמחלקה **SmartHomeAgentBehaviour**.

- בסיום כל איטרציה ישלח מכל סוכן לכלל השכנים הודעה המכילה את תוצאות האיטרציה הנוכחית שלו - - לצורך שליחה מוצלחת של הודעה זו יש למלא את השדה **currIteration** שבמחלקת **SmartHomeAgent**.
- כדי שכל סוכן ידע שהגיע לאיטרציה האחרונה שלו ועליו לסיים את פעולתו, יש להגדיל ב1 בכל איטרציה את השדה **currentNumberOfIter** הנמצא במחלקה **SmartHomeAgentBehaviour.**

לאחר המימוש, יש להפעיל את התוכנה, במסך הראשי יש ללחוץ על כפתור "Add New Algorithm" ולגרור את הקובץ אל החלון החדש אשר הופיע במסך.
האלגוריתם ייטען במערכת, יעבור תהליך הידור (compilation) ובדיקות בסיסיות.
בסוף התהליך האלגוריתם החדש יתווסף, בצורה קבועה, לרשימת האלגוריתמים הניתנים לבחירה.
<u>הערה חשובה</u> - מומלץ להוסיף אלגוריתמים חדשים בעלי שמות ייחודיים. אלגוריתם חדש ידרוס אלגוריתם קיים שאינו מובנה במערכת בעל שם זהה.

7. הוספת בעיות למערכת

לצורך הוספת בעיות חדשות למערכת יש להוסיף קבצי json תקניים בפורמט הבעיות המוצג בנספח א' בקובץ ה ARD. את הבעיות יש לשים בתיקיה SHAS/resources/problems ולדאוג ששמם יהיה בפורמט: *<city name>_<size>_<IDentifier>_<number>*
על מנת שישתלבו במקום הנכון בין תיקיות הבעיות בתצוגת ה UI שבעמוד הראשי.

שינויים שבוצעו ישתקפו ברשימת הבעיות הניתנות לבחירה בהרצה חדשה של המערכת או בעת רענון / פתיחת לשונית חדשה אל התוכנה בדפדפן.

8. תחזוקת המערכת

ככלל, המערכת בנויה כך שניתן לשנות את דרך ההתנהגות שלה בעזרת קבצי ההגדרות הרלוונטיים ורק אם נדרש שינוי מהותי יותר יש לגשת ולערוך את קוד המקור.

<u>קבצים רלוונטיים לתחזוקה בתיקיית התוכנה:</u>

- <u>אופן הפעולה הכללי של הקבצים runSHAS.bat, runSHAS.sh</u>:
  - ניווט אל תיקיית SHAS
  - הרצת השרת מתוך אותה התיקייה
  - פתיחת דפדפן ברירת המחדל אל הכתובת הקשורה לתוכנה מספר שניות מרגע ההפעלה
  - האזנה לפקודות:
    - פקודת kill אשר תסגור את התוכנה
    - פקודת open אשר תפתח דפדפן ברירת המחדל אל הכתובת הקשורה לתוכנה

- <u>הקובץ runSHAS.bat</u>
  זהו batch script אשר אחראי על הפעלת התוכנה במערכת הפעלה windows. מעבר לחופש של המשתמש לערוך קובץ זה כרצונו, ניתן לערוך בקלות את השורות:
  - "timeout X" - שינוי זמן ההמתנה X בשניות מהפעלת התוכנה ועד פיחת הדפדפן
  - "start X" - שינוי הכתובת X אליה גולש הדפדפן אל התוכנה
  - "if %command% == X goto Y" ניתן להוסיף פקודות נוספות אליהן הscritp מאזין כאשר Y הוא תווית אשר תחתיה פעולות ובסיומה במידת הצורך המתבצע " goto loop" בדומה למימוש הפקודה open

- <u>הקובץ runSHAS.sh</u>
  זהו bash script אשר אחראי על הפעלת התוכנה במערכת הפעלה linux.
  בדומה ל-runSHAS.bat גם אותו המשתמש רשאי לערוך בהתאם לרמת הידע ברשותו.
  שורות אשר ניתן לערוך בקלות:
  - "sleep X" - שינוי זמן ההמתנה X בשניות מהפעלת התוכנה ועד פיחת הדפדפן
  - "url=X" - שינוי הכתובת אליה נפתחת לשונית הדפדפן (נא לשים לב שאין רווח מיותר בביטוי)

- הוספת פקודה חדשה X והקוד שלה Y אליה הscript מאזין:

```
if [ “$command” == “X” ]
then
        Y
fi
```

- <u>הקובץ SHAS/webapps/SHAS.war</u>
זהו הקובץ המכיל את הקוד המקומפל של התוכנה. במידה וברצונכם לשקף שינויים בקוד אל דרך ההרצה הטבעית(runSHAS.bat, runSHAS.sh) עליכם לארוז את קוד המקור אל קובץ war מעודכן ולדרוס בעזרתו את הקובץ הנ"ל (חשוב לשמר את שם הקובץ להיות SHAS.war). את האריזה ניתן לבצע בעזרת פקודת "mvn install" ב-terminal. כדי לבצע פקודה זו עליכם להתקין Maven על המחשב בו ברצונכם לבצע פעולה זו. ניתן להשיג את אותו אפקט ע"י הרצת הפעולה מתוך תוסף Maven ב-IDE בו הפרויקט פתוח. אנו ממליצים בחום על שימוש ב intellij.

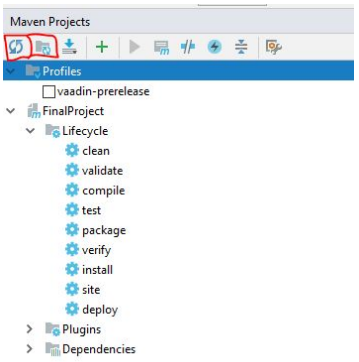
- <u>תיקיית SHAS/resources/algorithms/FinalProject/BL/Agents</u>
בתיקייה זו מאוחסנים האלגוריתמים אשר התווספו למערכת. במידה וברצונכם למחוק אלגוריתם אשר הוספתם למערכת ניתן למחוק את הקובץ בעל שם זהה בתיקייה זו ובפעם הבאה שהדף הראשי ייטען האלגוריתם הנ"ל לא יופיע.
<u>הערה חשובה</u> - אלגוריתמים שנוספו אינם מאוחסנים בשום מקום אחר מלבד בתיקייה המדוברת ולכן לא ניתן לשחזר את האלגוריתם שנמחק למעט הוספתו מחדש דרך ממשק המשתמש.

- <u>SHAS/conf.properties</u> - ניתן למצוא תיאור מפורט בחלק "שינוי קונפיגורציה".

- <u>SHAS/start.ini</u> - ניתן למצוא תיאור מפורט בחלק "שינוי קונפיגורציה".

<u>קוד המקור</u>
<u>הערה חשובה</u> - ההוראות הבאות נכתבו בהנחה ונעשה שימוש ב-intellij IDEA, בה השתמשנו לפיתוח התוכנה. קיימת גירסה חינמית של IDE זה המספק את כל היכולות הנדרשות כדי לעקוב אחר ההוראות. באופן כללי אנו ממליצים בחום להשתמש בIDE זה.
קוד המקור של התוכנה זמין כ-repository פומבי באתר GitHub תחת הכתובת:
https://github.com/miamiHits/FinalProject .

כדי לערוך את הפרויקט מומלץ להוריד אותו בעזרת פעולת "git clone" ולפתוח אותו בעזרת intellij IDE. כמו כן עליכם ללחוץ על הכפתורים "Reimport all Maven projects" ו-"generate sources and update folders for all projects" בחלון ה-Maven של intellj:



הרצת קוד המקור - כדי להריץ את הפרויקט יש להגדיר קונפיגורציית הרצה מסוג maven ובה להגדיר:

- תחת לשונית Parameters בשדה Command line את הפקודה "jetty:run"
- תחת לשונית Runner בשדה VM options את הפרמטרים:

"-Djetty.http.port=<Port> -Dlog4j.configuration=file://<log4j> -Dlog4j.debug"

הפרמטר הראשון מציין את ה-port אליו יש לגלוש דרך הדפדפן

הפרמטר השני מציין את מיקום קובץ ההגדרות של הלוג - log4j.properties

הפרמטר השלישי הנו אופציונלי ומציג הדפסות של תהליך טעינת log4.properties, אשר יסייעו לכם לפתור בעיות של טעינת קובץ זה.

הרצת קונפיגורציית ריצה הנ"ל תריץ את התוכנה ויהיה ניתן לגשת אליה בדפדפן דרך הכתובת:
localhost:<Port>

כאשר ה-port הוא זה שהוגדר בפרמטר הראשון.

remote debugging - כפי שצוין בחלק שינוי קונפיגורציה קיימת הגדרת בקובץ SHAS/start.ini המאפשרת לבצע debugging אל אפליקציה שרצה ולא הופעלה דרך intellij. כדי לעשות זאת יש להגיד קונפיגורציית ריצה מסוג remote המכוונת אל כתובת ה-IP של המחשב המריץ את התוכנה(במידה וה-debug מתבצע באותו המחשב אז זהו "localhost") ואל ה-port אשר הוגדר ב-start.ini

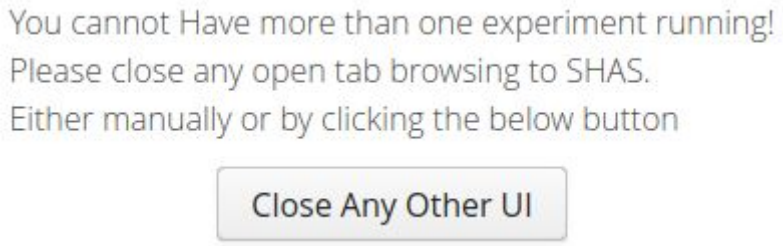
conf.properties - כפי שצוין בחלק "שינוי קונפיגורציה" ניתן להוסיף מאפיינים נוספים הנטענים דינמית

מ-conf.properties. טעינת קבצים אלו המתבצעת דרך שימוש במחלקה Config הטוענת את הקובץ הנ"ל אל תוך מופע מסוג Properties (מובנה ב-Java). כל שעליכם לעשות הוא להוסיף קריאה לConfig.getStringProperty במקומות הרלוונטיים בקוד ולהוסיף ערכים עבור אותו property בכלל הפרופילים בקובץ.

הערה חשובה - למנגנון זה אין ערכי ברירת מחדל ולכן יש לבצע עריכות אלו בצורה קפדנית ביותר כדי להימנע מבעיות אשר ינבעו משינויים אלו.

9. פתרון תקלות אפשריות (troubleshooting)

- <u>פתחתי דף חדש של התוכנה דרך הדפדפן וקיבלתי את העמוד הבא:</u>

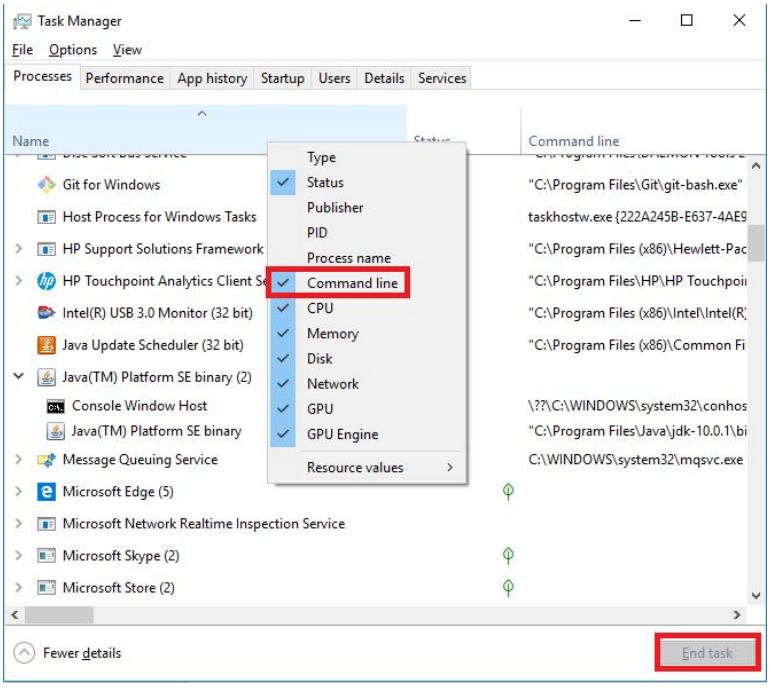


<u>פתרון:</u>
התוכנה אינה מאפשרת יותר מריצה של ניסוי אחד במקביל. כדי לאכוף זאת, לא ניתן לגלוש אל התוכנה בעזרת יותר מלשונית דפדפן אחת. הודעה זו תופיע אם כבר קיימת לשונית שכזו, גם אם הלשונית נפתחה מדפדפן אחר ואף ממחשב אחר.
במצב כזה יש ללחוץ על הכפתור הנ"ל או לחלופין, לסגור ידנית את כלל הלשוניות הקיימות ולרענן את העמוד.

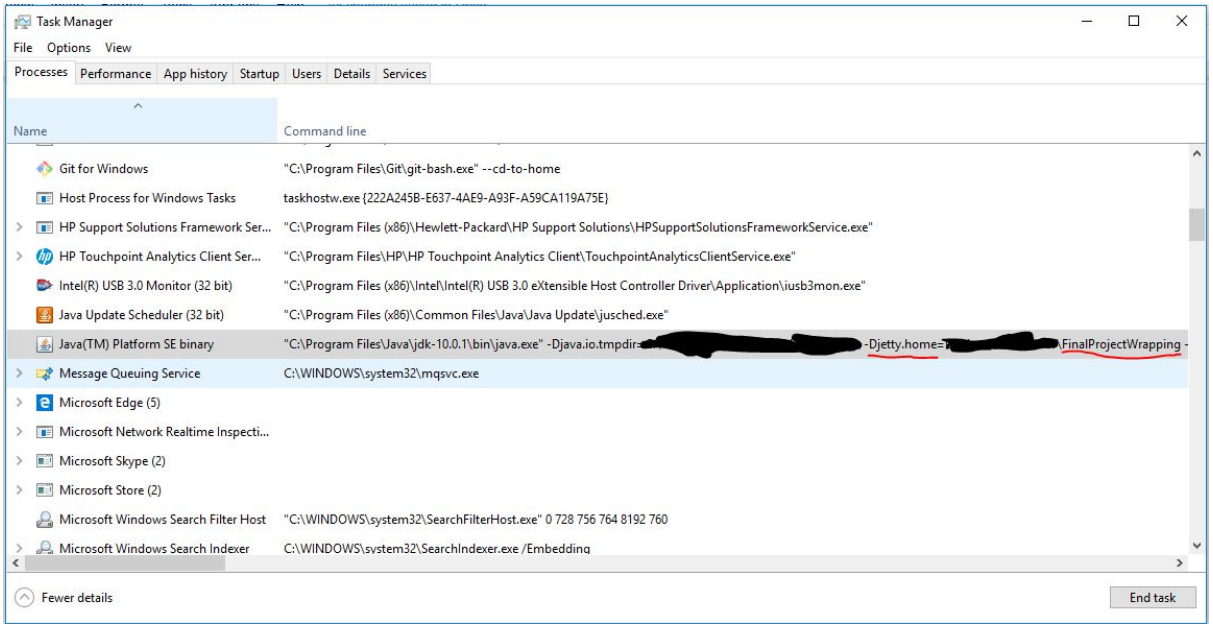
- <u>סגרתי את הדף של המערכת בדפדפן:</u>
<u>פתרון:</u>
ניתן לפתוח מחדש את הדף ע"י הקלדת הפקודה "open" בחלון אשר נפתח בעת הרצת המערכת. לחלופין, ניתן לפתוח לשונית חדשה בדפדפן ולגלוש אל הכתובת localhost:8888/SHAS

- <u>סגרתי את החלונות של המערכת ללא שימוש בפקודת "kill" וכעת אינה פועלת כשורה:</u>
<u>פתרון:</u>
במצב כזה עליכם לסגור את התהליך בצורה ידנית שכן לא ניתן להריץ שני מופעים של המערכת במקביל.
<u>במערכת הפעלה windows</u> עליכם לעשות זאת בעזרת Task manager. על מנת לזהות את התהליכים אשר קשורים אל המערכת וכדי למנוע סגירה של תהליכים אשר אינם קשורים

יש לאפשר את העמודה המתארת את שורת פקודת ההרצה:

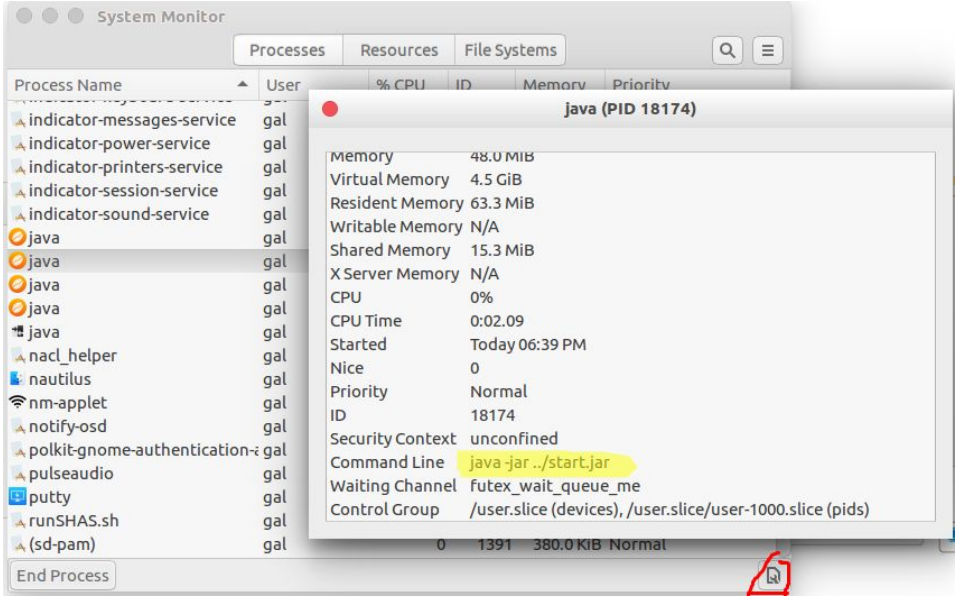


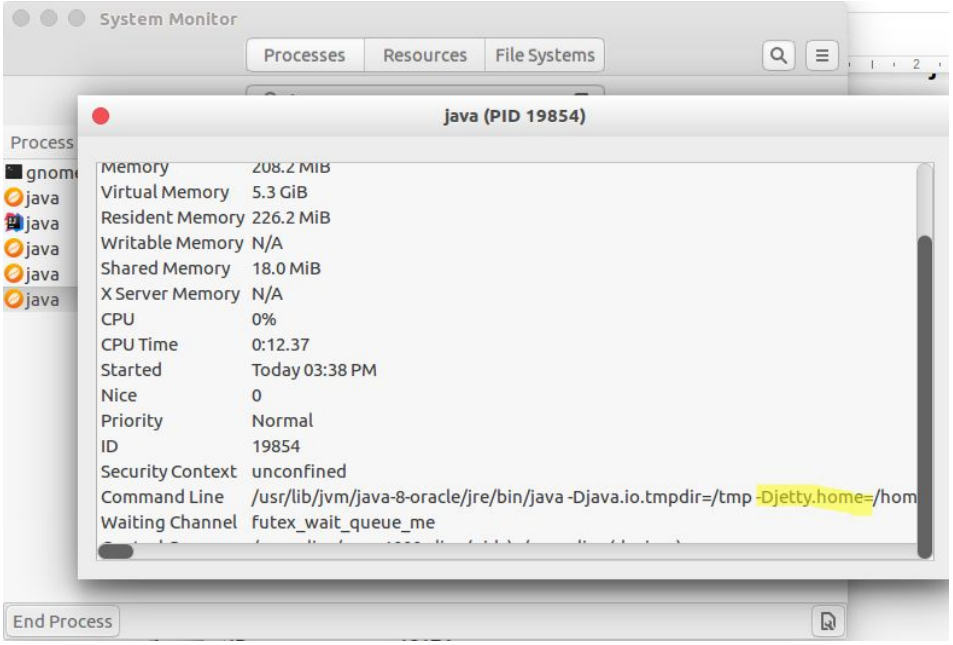
התהליכים אותם יש לסגור ע"י סימון ולחיצה על כפתור "End Task" הם תהליכי Java אשר מכילים אזכורים אל המיקום של המערכת בכונן הקשיח ובפרט את הארגומנט "-Djetty.home":



או כאלו אשר שורת הפקודה שלהם היא "java -jar ../start.jar" באופן דומה.

- במערכת הפעלה ubuntu עליכם לעשות זאת בעזרת System Monitor. אפשרי שיהיו מספר תהליכי Java ולכן עליכם לסגור את התהליך שפקודת ההפעלה שלו היא "java -jar ../start.jar" או כזה אשר מכיל אזכורים אל המיקום של המערכת בכונן הקשיח ובפרט את הארגומנט "-Djetty.home":





- <u>הוספתי בעבר אלגוריתם למערכת אך כעת ברצוני להסיר אותו</u>

<u>פתרון:</u>

בתיקייה *SHAS/resources/algorithms/FinalProject/BL/Agents* מאוחסנים האלגוריתמים אשר התווספו למערכת. במידה וברצונכם למחוק אלגוריתם אשר הוספתם למערכת ניתן למחוק את הקובץ בעל שם זהה בתיקייה זו ובפעם הבאה שהדף הראשי ייטען

האלגוריתם הנ"ל לא יופיע.
הערה חשובה - אלגוריתמים שנוספו אינם מאוחסנים בשום מקום אחר מלבד בתיקייה המדוברת ולכן לא ניתן לשחזר את האלגוריתם שנמחק למעט הוספתו מחדש דרך ממשק המשתמש.

- ניסיתי להוסיף אלגוריתם למערכת אך קיבלתי הודעה על שגיאת קומפילציה:
שגיאת הקומפילציה יכולה לנבוע משתי סיבות:
1) הקובץ אותם ניסית להוסיף למערכת אינו קובץ Java חוקי (כלומר קיימת שגיאה תחבירית (סינטקטית) בקובץ.
2) הקובץ אינו מחלקה המממשת את הממשק עבור אלגוריתם, כלומר אינה יורשת מן המחלקה SmartHomeAgentBehaviour או שאינה מממשת את אחת השיטות (methods) הנדרשות מירושה זו.
בכל מקרה, המערכת תציג פירוט עבור השגיאה שקרתה.

- תיקיית log:
ניתן להתחקות אחר כל תקלה אחרת בעזרת קבצי הlog. לאורך כל הריצה מודפסות אל קובץ זה הודעות המעידות על מצב המערכת ופעולות אשר התבצעו. קבצים אלו מחולקים כך שהקובץ המכיל את ההודעות האחרונות ביותר. כל הודעה מציינת מתי והיכן בקוד נכתבה ומהי דרגת החשיבות שלה. כאשר קובץ זה מגיע לגודל של 10MB הוא משנה את הסיומת שלו ל"log.1" והודעות חדשות נכתבות לקובץ חדש בשם "FinalProject.log". במידה וקיים קובץ בשם "FinalProject.log.1" קובץ זה וכל הקבצים לאחריו "נדחפים אחורה", כלומר הקובץ "FinalProject.log.x" משנה את שמו להיות "FinalProject.log.x+1". לדוגמא:
"FinalProject.log.18" ==> "FinalProject.log.19"
מתאפשרים עד 100 קבצים כאלו ובשלב מסוים הקובץ הישן ביותר יידרס על ידי זה הקודם לו.