# אובונגו – מדריך משתמש

# דרישות מערכת

כל שרת פיזי עליו מותקנת המערכת (כלומר, מריץ שרת Ubongo או מכונת Ubongo) צריך לקיים את הדרישות הבאות:

* גישה לשרת הקבצים : /specific/netapp5/hezi
* התקנה של rabbitMQ ע"י צוות הסיסטם של האוניברסיטה, לפי ההנחיות המצויות בלינק הבא: [https://www.rabbitmq.com/install-rpm.html](https://www.rabbitmq.com/install-rpm.html" \t "_blank)
* Credentials של יוזר בעל הרשאות כתיבה, קריאה והרצה לכל תיקיות המערכת – מעודכן בקובץ הקונפיגורציה של המערכת. ניתן לרשום בקובץ זה סיסמא או להפנות לקובץ SSH Public Key, אותו ניתן לייצר לפי ההנחיות בעמוד [http://www.cs.tau.ac.il/system/servers#servers19](http://www.cs.tau.ac.il/system/servers%23servers19).
* Java 1.8

דרישות נוספות:

* DB מסוג MySQL שפרטיו מוגדרים בקובץ הקונפיגורציה. את בניית מסד הנתונים ניתן לבצע באמצעות הקובץ buildDatabase.sql הנמצא בתיקיית db בשרת (ובקוד המערכת).
* Tomcat (נבדק עם גרסאות 6, 7 ו-9) בו יוצב ה-Web Application (קובץ ה-WAR).

# התקנה וקונפיגורציה

המערכת מסופקת כקובץ zip המחולק למספר תיקיות:

1. תיקיית webapp – מכילה קובץ בשם ubongo.war אותו יש להעתיק כפי שהוא לתיקיית webapps תחת תיקיית ההתקנה של Tomcat. לאחר העתקת הקובץ Tomcat יפרוס את הקובץ וייצור תיקיית ubongo באותו מיקום. כדי לעדכן את שירות ה-HTTP או ממשק המשתמש יש למחוק מתיקייה זו את ubongo.war ואת התיקייה שנוצרה בעקבותיו (ubongo) ולהעתיק מחדש את הקבצים למיקום זה. אין צורך לעצור או להפעיל מחדש את Tomcat. יש לשים לב כי בתוך קובץ זה נמצא קובץ קונפיגורציה עבור Tomcat, הנקרא context.xml וממוקם בתיקיית META-INF. בקובץ זה מוגדרים מספר פרמטרים חשובים – מיקום קבצי היוניטים, מיקום השאילתות עבור מסד הנתונים ומיקום קובץ הקונפיגורציה. ניתן להגדיר עבור כל הנתיבים הללו את הנתיבים המתאימים למיקום הקבצים בשרת, כדי שלא יתרחש מצב של חוסר תאימות, בו קבצי היוניטים בשרת ובממשק המשתמש שונים, למשל.
2. תיקיית server – יש להעתיק את כל תוכנה של תיקייה זו למיקום בו תרוץ המערכת. אחד מהקבצים, המופיע בתיקיית config, הוא קובץ הקונפיגורציה הראשי. בקובץ זה יש להגדיר את המשתמש שבאמצעותו יתבצע החיבור למסד הנתונים, ולרשום את המכונות אשר יזוהו על-ידי המערכת.
3. תיקיית machine – יש להעתיק את כל תוכנה של תיקייה זו לתיקייה חדשה באותו מיקום בו הוצבו הקבצים של השרת. המלצתנו היא לקרוא לתיקייה בשם שיזהה את מספרה, למשל machine1. בהתקנת מכונה, מעבר להעתקת הקבצים יש כמובן לדאוג שבקובץ הקונפיגרוציה הראשי המכונה תירשם באופן מדויק (לפי ה-IP והשם של השרת שיריץ את המכונה) ולעדכן את קובץ הקונפיגורציה של הלוג; גשו לתיקיית config של המכונה שיצרת ובקובץ שנו את השורה <Property name="machineId">1</Property> כך שהמספר יתאים למספר המכונה. הגדרה זו תדאג שכל הודעות הלוג מהמכונה הזו יכתבו לקובץ המתאים.
4. תיקיית sources – קבצי המקור של המערכת. ניתן לערוך אותם וליצור באמצעותם גרסאות נוספות בעתיד. המערכת נבנתה באמצעות Apache Maven ולכן מומלץ להשתמש בכלי זה במקרה של יצירת גרסאות עתידיות – הוא יבטיח יצירה נכונה לפי המבנה שתואר לעיל.
5. תיקיית docs – מסמכי תיעוד עבור המערכת ובכללותם מסמך זה.

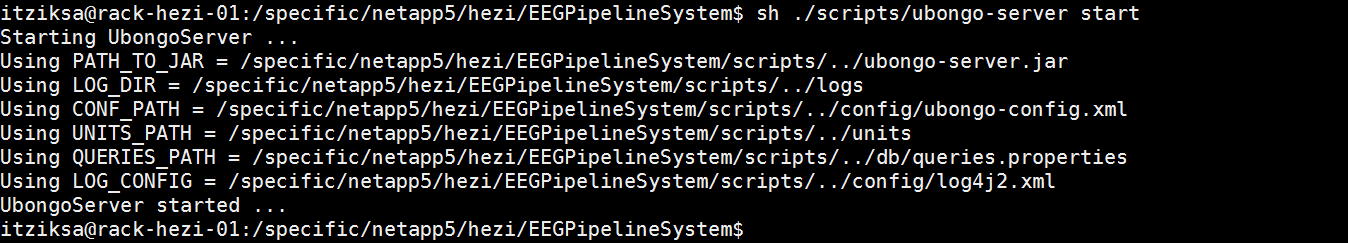
# הפעלה ותחזוקה

המערכת מורכבת כאמור משלושה חלקים והפעלתם מעט שונה:

1. Webapp – כאמור, מספיק להציב את קובץ ה-war במקום המתאים.
2. שרת – בתיקיית scripts של השרת נמצא קובץ בשם ubongo-server. בהינתן שהמיקום הנוכחי הוא תיקיית הבסיס של השרת (בה נמצא הקובץ ubongo-server.jar ותיקיית scripts), יש להפעיל את השרת באמצעות הפקודה הבאה:

sh ./scripts/ubongo-server start

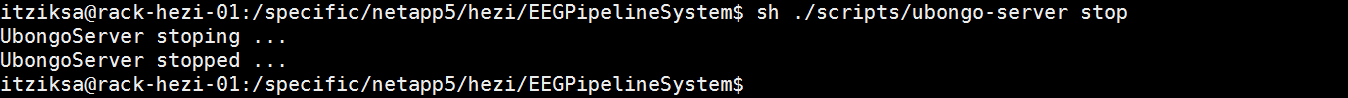
לאחר הפעלת השרת יופיעו מספר שורות המפרטות אודות מספר נתיבים בהם השרת ישתמש, כמו נתיב הקונפיגורציה לדוגמה.



עם ההפעלה יווצר קובץ בתיקיית הבסיס של השרת, עם השם ubongo-server-pid.

על-מנת לעצור את השרת יש להריץ את הפקודה הבאה:

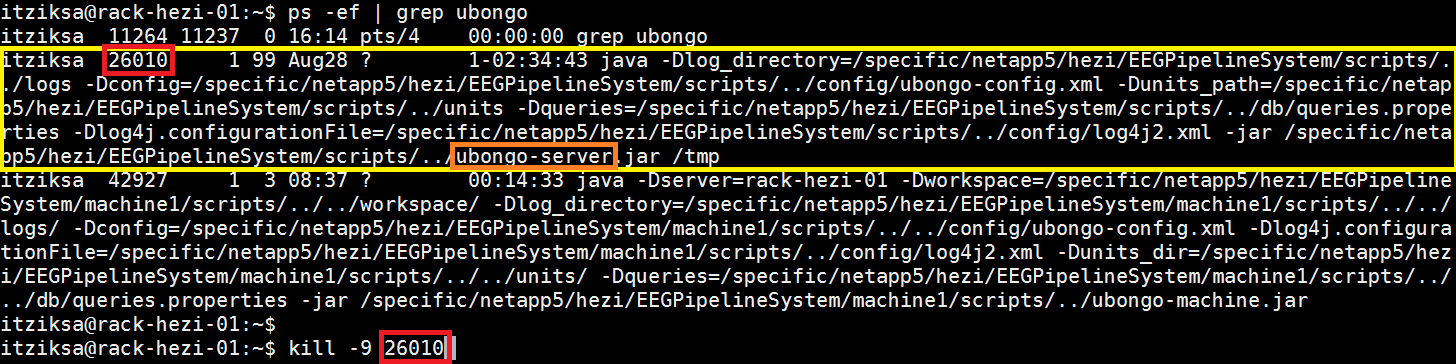
sh ./scripts/ubongo-server stop



יש לשים לב כי את שתי הפעולות יש לבצע אך ורק מתוך המכונה שמריצה את השרת (לדוגמה, rack-hezi-01). כדי לוודא שהשרת עלה/עצר ניתן להיעזר בממשק המשתמש כפי שיוסבר בהמשך (מתוך מסך הניהול). במקרה בו השרת לא עצר למרות השימוש בפקודה לעיל, יש לחפש את מזהה התהליך של השרת על-ידי הפקודה:

ps -ef | grep ubongo

הפלט שיופיע יראה דומה למסך הבא:



כעת יש להשתמש בפקודה הבאה, כאשר המספר האחרון מוחלף במזהה שיופיע בפלט כפי שמסומן לעיל:

kill -9 26010

1. מכונות – העבודה עם המכונות דומה מאוד לזו עם השרת. יש כמובן להקפיד גם כאן להפעיל ולעצור מכונות מהמכונה המתאימה (לדוגמה, rack-hezi-03). את הפקודות יש להריץ הפעם מתיקיית המכונה (ניתן להריץ אותן מכל מיקום אחר, כך הנתיב בפקודה יהיה שונה). ההבדל היחיד בין הפקודות הוא שבמקום פקודת ההרצה של השרת, יש להשתמש בפקודה הבאה:

sh ./scripts/ubongo-machine <server-hostname> start

ולהחליף בפקודה את <server-hostname> בשם המכונה שמריצה את השרת (לדוגמה, rack-hezi-01).

על-מנת לעקוב אחר ריצת הרכיבים השונים, בעיקר במקרי תקלות, נרשמים קבצי לוג.

הקונפיגורציה של קבצי הלוג מוגדרת בקבצים הנקראים log4j2.xml. ישנו קובץ כזה עבור כל מכונה בתיקיית config המתאימה, וכך גם עבור השרת. קובץ הקונפיגורציה של ה-webapp נמצא בתוך ה-war; ניתן לערוך אותו לאחר ההצבה ב-Tomcat מתוך התיקייה שתיווצר באופן אוטומטי (ubongo) או על-ידי עריכתו בתוך ה-war. כדי לערוך קובץ בתוך ה-war ניתן להוריד תוכנה ייעודית או פשוט לשנות את סיומת הקובץ ל-zip, לערוך כרגיל ולאחר מכן להחזיר את הסיומת ל-war. כיום הלוגים של ה-webapp נרשמים במיקום של Tomcat תחת תיקיית logs/ubongo אולם ניתן לשנות זאת כך שישמרו באותו מיקום עם יתר הלוגים, על-ידי שינוי קובץ הקונפיגורציה.

הלוגים של השרת והמכונות נשמרים כולם לתיקיית logs הממוקמת בתיקיה בה מותקן השרת (היכן שנמצא קובץ ה-jar ותיקיית ה-scripts שהוזכרה מוקדם יותר).

### תחזוקה שוטפת

1. הלוגים נאגרים בתיקיית ה-logs ואינם נמחקים באופן אוטומטי. על כן מידי פעם יש למחוק לוגים ישנים.
2. קבצי המטלב ולוגי ריצות המטלב הנוצרים בזמן ריצת יוניטים ע"י קבצי ה-bash – נשמרים בתיקיית units/bashTmp ואינם נמחקים משם. על כן מידי פעם יש למחוק גם מתיקייה זו קבצים ישנים.

# ממשק המשתמש

ממשק המשתמש מחולק לשלוש תצוגות, ביניהן ניתן לעבור על-ידי לחיצה על הקישורים בראש העמוד ויפורטו מיד. חשוב לציין כי בחלק גדול מהמקרים, פעולות המתבצעות בממשק המשתמש אינן מתבצעות מיד אלא יוזמות בקשה לשרת. לדוגמה, כאשר המשתמש בוחר להריץ Flow, ה-Flow נשמר למסד הנתונים ונשלחת בקשה לשרת להרצה. רק לאחר מספר שניות השרת יזהה את הבקשה ויתחיל לטפל בה. לכן, במידה ולחצת על כפתור, יש לעקוב אחר מצב הבקשות בתצוגת הניהול טרם לחיצה נוספת, על-מנת למנוע יצירת בקשות כפולות. בביצוע פעולות בממשק מופיעות הודעות בירוק או באדום מהן אפשר להסיק האם הבקשה נשלחה או לאו.

תצוגות המערכת:

* תצוגת שליטה ובקרה (Dashboard):
  + מכונות – בחלקו הימני של המסך מופיעה טבלה המציגה את המכונות המוגדרות במערכת. הערכים נקבעים בקובץ הקונפיגורציה המרכזי (ubongo-config.xml) ונטענים עם עליית השרת אל מסד הנתונים. במידה ורוצים להוסיף או להסיר מכונה מהמערכת יש לעדכן את קובץ הקונפיגורציה ולהפעיל מחדש את השרת.

הטבלה מורכבת מארבע עמודות – כתובת ה-IP של המכונה, שמה ועמודות Connected ו-Active. עמודת Connected נצבעת בירוק במידה והתקבל סיגנאל Heartbeat מהמכונה בשתי הדקות האחרונות ובאדום אחרת. תא צבוע בירוק בעמודת ה-Active מצביע על מכונה שמוגדרת במערכת כפעילה – בהינתן שמכונה פעילה ומחוברת היא למעשה זמינה להרצת מכונות מבחינת השרת.

מתחת לטבלה מופיע הזמן האחרון בו התקבל סיגנאל Heartbeat מהשרת (זמן זה יצבע באדום אם הוא ישן מדי ומצביע על שרת מנותק). בלחיצה על אחת המכונות יופיע תחת הטבלה גם זמן ה-Heartbeat האחרון של המכונה הבחורה ויופיע כפתור Activate/Deactivate.

אם ברצונך לעצור מכונה מחוברת מקבלת משימות באופן זמני (בניגוד להסרת המכונה לחלוטין מהמערכת על-ידי הסרתה מקובץ הקונפיגורציה), בחר את המכונה מתוך הרשימה. עם הבחירה יופיע כפתור באמצעותו ניתן לשנות את מצב הפעילות של המכונה. אם מופיע Activate, לחיצה עליו תהפוך את המכונה לפעילה, ואם Deactivate, לחיצה תעביר את המכונה למצב בלתי פעיל – חיווי על כך ניתן לראות בטבלה לפי הצבעים בעמודת ה-Active.

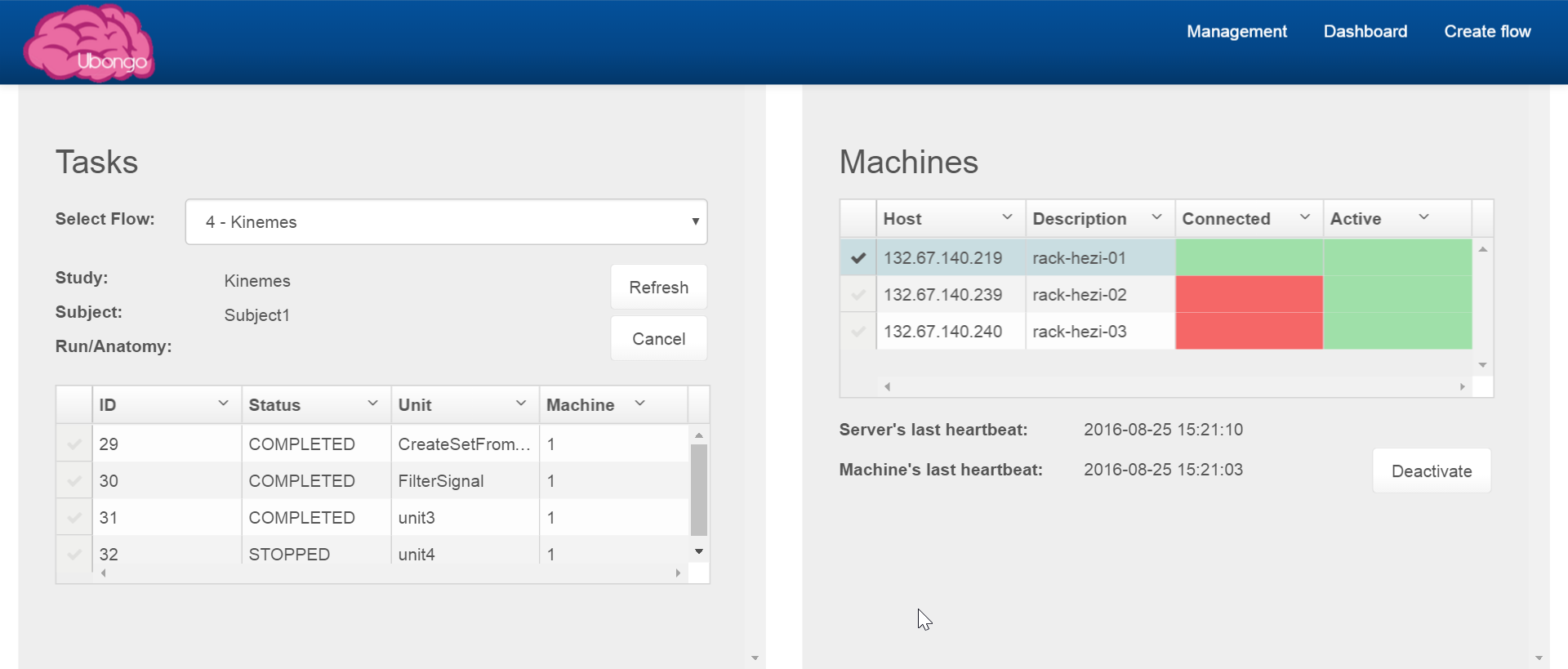
חלק המסך המציג את המכונות מתעדכן באופן אוטומטי אחת לדקה, אולם ניתן לרענן את המסך כולו במידת הצורך.

* + משימות – בחלקו השמאלי של המסך מופיע ממשק למעקב וניהול אחר משימות. במסך זה ניתן לבחור Flow, לצפות בנתוני ההקשר שלו (מחקר, נבדק, ריצה) וברשימת המשימות המרכיבות אותו ממוינות לפי מספרן הסידורי.

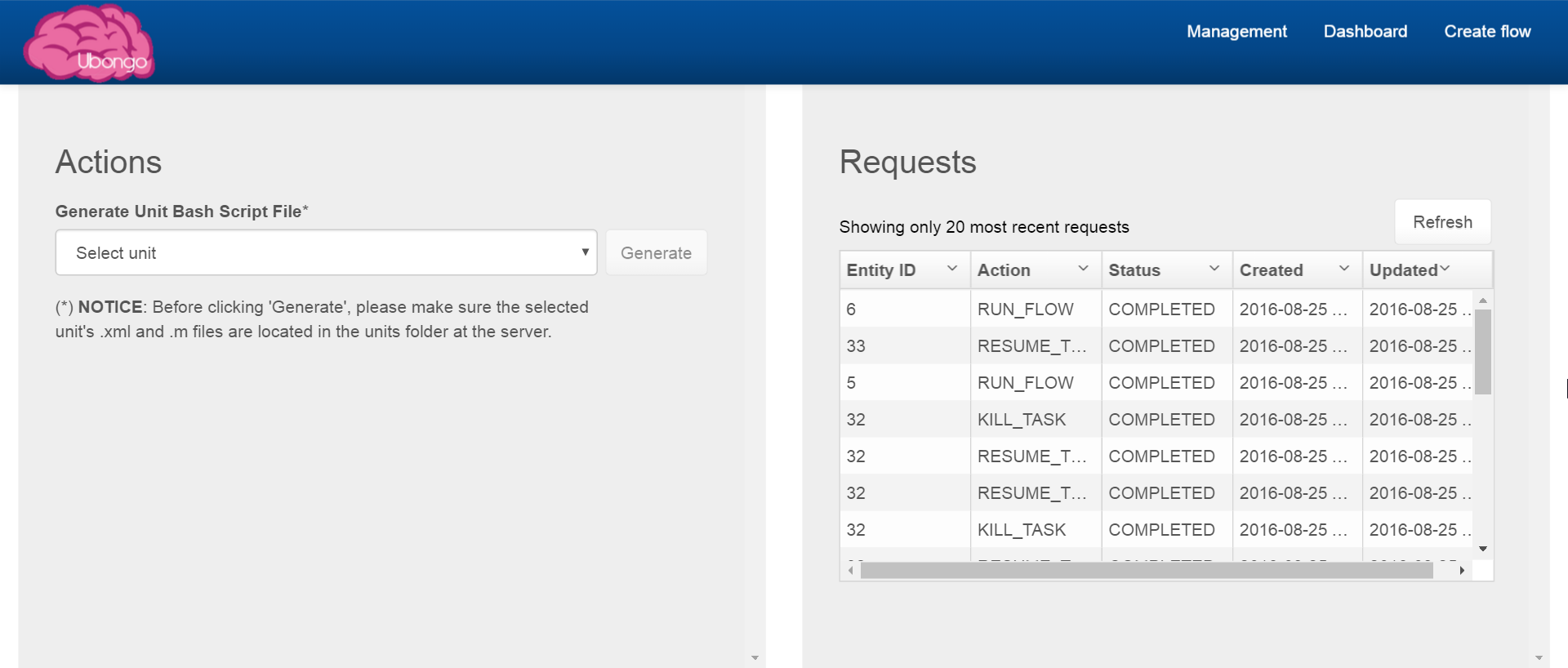
בבחירת אחת המשימות בטבלה תופיע טבלה נוספת המאפשרת לצפות בפרטים נוספים עבור המשימה (כמו הפרמטרים עבור היוניט שלה, נתיבי קלט ופלט וכו'). בין הפרמטרים מופיעים גם נתוני הקשר – נבדק וריצה. זו איננה כפילות אל-מול הנתונים המופיעים עבור ה-Flow כיוון שיתכן Flow הרץ על כלל הנבדקים (מסומן על-ידי .\*), ואז כל משימה תהיה קשורה לנבדק ספציפי אחר תחת אותו מחקר.

בנוסף לטבלה שתופיע לאחר בחירת משימה, יופיעו גם כפתורים לעצירה, ביטול והפעלה מחדש של משימה. עבור כל משימה, רק כפתור אחד מבין השלושה יהיה פעיל, בהתאם למצבה. לדוגמה, משימה הרצה על מכונה, ניתנת לעצירה בלבד; משימה שנכשלה, ניתנת להפעלה מחדש. יוצאת מן הכלל היא משימה אשר הושלמה בהצלחה ולה אין פעולות זמינות. במידה וברצונך לבטל Flow שלם, ניתן ללחוץ על כפתור הביטול המופיע לצד נתוני ההקשר של ה-Flow.

במידה וה-Flow הנצפה רץ ברגעי הצפיה, מומלץ להשתמש בכפתור הרענון (מופיע מעל כפתור הביטול, מימין לנתוני ההקשר). כפתור זה שימושי גם לאחר ביצוע פעולות על משימות או לאחר לחיצה על ביטול Flow. כפתור זה יטען גם Flows נוספים ולכן ניתן ללחוץ עליו כדי לקבל אותם. חשוב לזכור כי לחיצה על אחד מהכפתורים בממשק תגרום לשליחת בקשה לשרת ולכן יתכן עיכוב בטיפול בבקשה, לכן עליך להמתין מספר שניות טרם המשימות תשננה מצבן בעקבות פעולה אשר יזמת. על-מנת לעקוב אחר מצב הבקשות ניתן לעבור למסך הניהול.



* תצוגת ניהול (Management):
  + בקשות – בחלקו הימני של מסך זה מופיעה טבלה המציגה את הבקשות במערכת. בקשות אלה נשלחות מממשק המשתמש (או שירות ה-HTTP ליתר דיוק) אל השרת המרכזי ומבוצעות על-ידי האחרון. הממשק מציג את 20 ההודעות האחרונות בלבד. באמצעות מסך זה ניתן לעקוב אחר ביצוע פעולות, כגון עצירת משימה, הרצת Flow וכן הלאה. כאשר נוצרת בקשה היא במצב New, אם הושלמה בהצלחה היא עוברת למצב Completed ואם נכשלה היא עוברת למצב Failed.
  + יצירת סקריפט Bash עבור יוניט – בחלקו השמאלי של המסך ניתן לבחור יוניט מתוך הרשימה ולייצר עבורו סקריפט Bash, הנחוץ להרצת היוניט (פעולה זו צריכה להתבצע רק פעם אחת ולאחר מכן הסקריפט קיים עבור היוניט). חשוב לדאוג שקבצי ה-XML והמטלב עבור היוניט יהיו קיימים בנתיב המתאים בשרת טרם הלחיצה על יצירת הסקריפט.



* תצוגת יצירת flow (Create Flow):

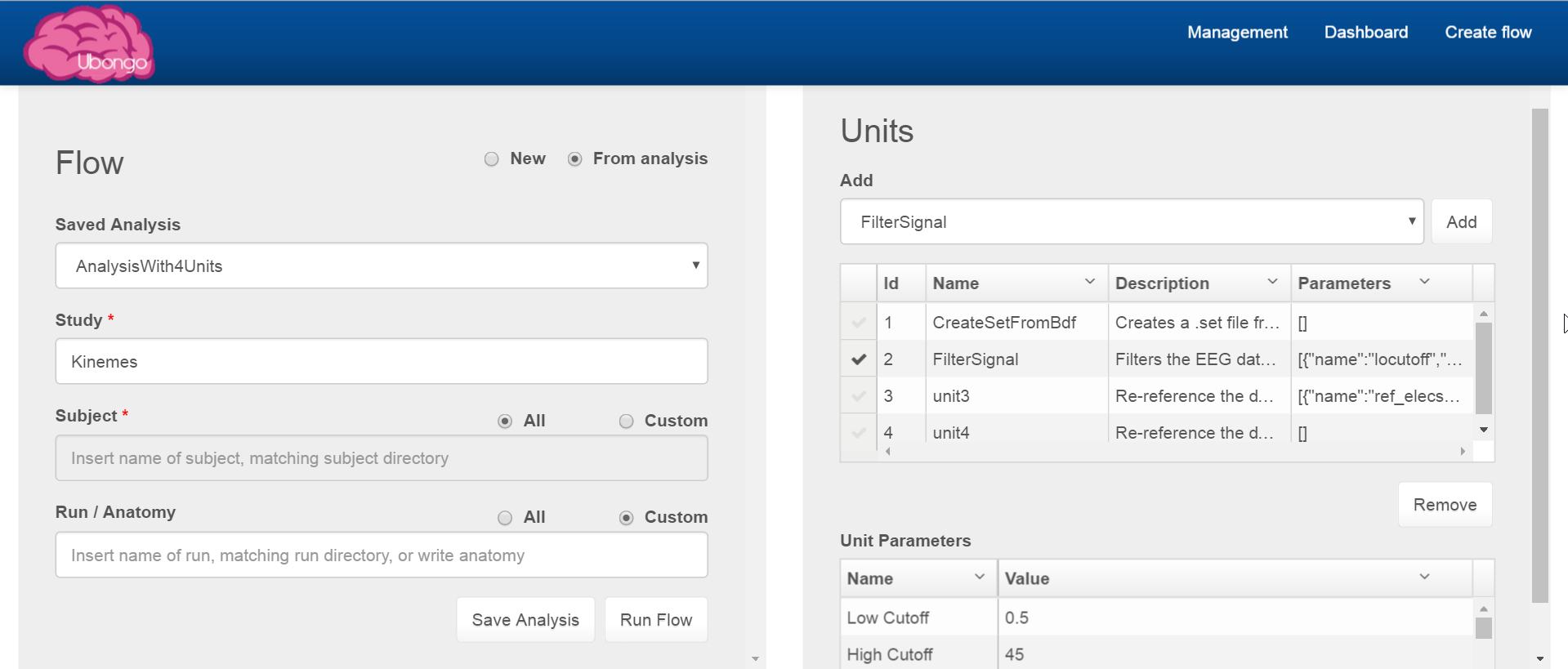
מסך זה נועד ליצירת Flows ואנליזות חדשים. אנליזה היא רצף יוניטים חסר פרטי הקשר ו-Flow הוא אנליזה עם הקשר (מחקר, נבדק, ריצה).

* + הקשר – בצדה השמאלי של התצוגה המשתמש יכול להזין פרטי הקשר, כאשר ניתנת בחלקם האפשרות לבחור ב-All להרצה על כל הנבדקים/הריצות במחקר/בנבדק המתאים. כמו כן, ניתן לתת שם לאנליזה לשימוש חוזר ולשמור אותה באמצעות הכפתור בתחתית המסך (הרצת Flow אינה שומרת אנליזה באופן אוטומטי!), או לטעון אנליזה קיימת. טעינת אנליזה (From analysis) תטען לחלק המסך הימני את היוניטים המרכיבים את האנליזה שנבחרה. יש לציין כי בשמירת אנליזה נשמרים היוניטים המרכיבים אותה עם ערכי ברירת המחדל לפרמטרים ולכן יש ביצירת Flow לוודא כי ערכי הפרמטרים ליוניטים השונים, הם הערכים הרצויים.
  + יוניטים – בחלקו הימני של המסך מופיעה טבלה של יוניטים. טבלה זו, בשונה מכל יתר הטבלאות במערכת, ניתנת לסידור מחדש – כלומר, ניתן לגרור שורות בטבלה על-מנת לשנות את הסדר שלהן ובכך לקבוע את סדר הריצה של היוניטים. ניתן גם לקבוע סדר זה על-ידי הוספת יוניטים לפי הסדר מלכתחילה. היוניטים המופיעים בטבלה מגיעים מאנליזה שמורה או על-ידי הוספה מתוך רשימת היוניטים המופיעה מעל הטבלה.

ניתן לבחור יוניט מתוך הרשימה החלקו העליון של המסך וללחוץ על כפתור ה-add על-מנת להוסיף אותו לרשימת היוניטים באנליזה/Flow. כדי להסיר יוניט מתוך הרשימה, יש לבחור את השורה המתאימה וללחוץ על כפתור Remove שיופיע תחת הטבלה.

בבחירת אחד מהיוניטים בטבלה תופיע טבלה נוספת בתחתית העמוד. טבלה זו מפרטת את הפרמטרים עבור יוניט זה ואת הערכים עבורו. לפני עדכון יופיעו ערכי ברירת המחדל כפי שמוגדרים בקבצי הקונפיגורציה של היוניטים. באמצעות לחיצה כפולה (דאבל קליק) על הערך בשורה המתאימה, ניתן לערוך את הערך אותו יקבל הפרמטר.

בסיום עריכת כל הפרטים הרלוונטיים יש ללחוץ על כפתור Run flow אשר ישמור את ה-Flow למסד הנתונים וישלח בקשה לשרת להרצתו. ניתן לעקוב אחריו במסך ה-Dashboard.



# קבצי ה-Units

כל מכונת Ubongo צריכה גישה לתיקיית היוניטים המיועדים להרצה במערכת. הנתיב לתיקייה זו הוא חלק ממשתני הקלט של ubongo-machine ומוגדר בסקריפט.

כל יוניט במערכת מורכב הקבצים הבאים, כאשר המספר <ddd> מייצג את מספר היוניט, שהינו מזהה **ייחודי** בעל 3 ספרות.

* unit\_<ddd>.xml
* unit\_<ddd>.m
* unit\_<ddd>.sh

## unit\_<ddd>.xml

זהו קובץ הקונפיגורציה של היוניט.

קובץ זה מכיל את השדות הבאים :

* unit id - מזהה ייחודי של ה-unit. מזהה זה צריך להיות ייחודי ותואם למזהה בשמות קבצי ה-unit (זהו המזהה <ddd>).
* Name – שם היוניט כפי שיוצג בממשק המשתמש.
* Description – תיאור קצר של היוניט.
* Input-files – נתיב לקבצי הקלט של היוניט.
  + קיימות מספר אפשרויות עבור קונפיגורציה זו:
    - נתיב מלא של תיקייה – **אם נתיב זה מייצג נתיב לתיקייה – עליו להסתיים ב-"/".**
    - נתיב מלא לתיקייה, אך עבור קבצי קלט מסויימים המתאימים לביטוי רגולרי (Regex) :
      * במקרה זה הנתיב יסתיים ב-Regex. לדוגמא:

/specific/…/{study}/{subject}/RawData/(.\*).bdf

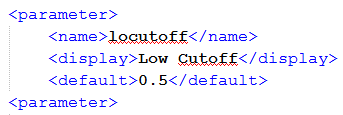
* + בנוסף לאפשרויות הנ"ל, הנתיב יכול לכלול Regex לתיקיות. לדוגמא:
* /specific/…/Experiments/**.\***/{subject}/RawData/(.\*).bdf
  + הנתיב יכול גם להכיל משתני Context שיקבלו ערך ע"י המשתמש בעת הרצת Flow. לדוגמא:

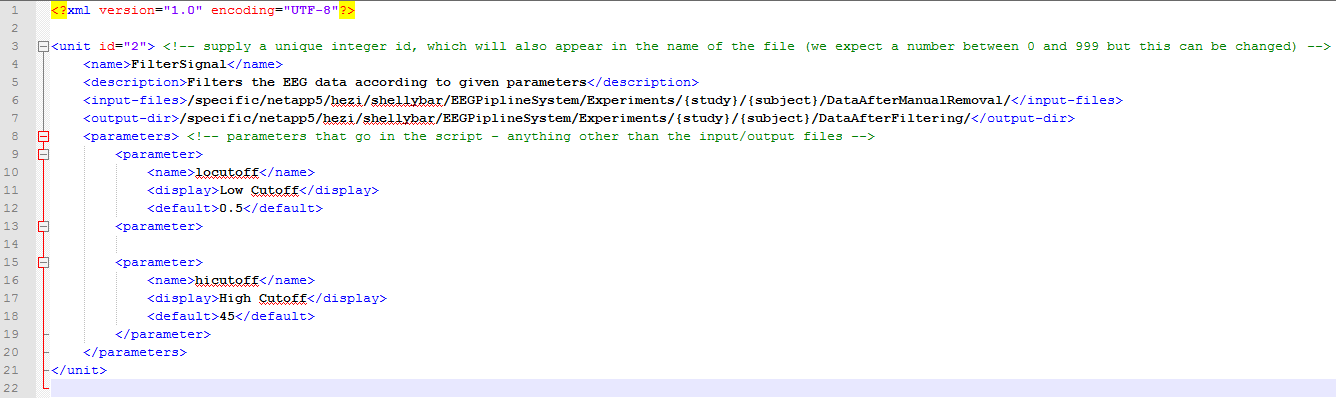
/specific/…/Experiments/{study}/{subject}/DataAfterManualRemoval/

* Output-dir – נתיב לקבצי הפלט של היוניט.
  + בשונה מנתיב הקלט, נתיב זה הינו יחיד ועל כן אינו יכול לכלול בתוכו Regex (לא לקבצים ולא לתיקיות).
  + הנתיב כן יכול להכיל משתני Context שיקבלו ערך ע"י המשתמש בעת הרצת Flow. לדוגמא:

/specific/…/Experiments/{study}/{subject}/

* parameters– הפרמטרים הדרושים לעיבוד היוניט (כלומר לקובץ המטלב של היוניט).
  + שדה זה מתייחס לכל הפרמטרים הנוספים מלבד ה-input-files וה- output-dir- מאחר ואלו נשלחים לכל יוניט באופן דיפולטי.
  + שדה זה יכול להיות ריק אם אין צורך בפרמטרים.
  + לכל פרמטר יש להגדיר :
    - name – שם הפרמטר כפי שמופיע בקובץ המטלב של היוניט (unit\_<ddd>.m)
    - display – תיאור קצר שיופיע ב-UI.
    - default - שדה זה אינו חובה. זהו ערך דיפולטי לפרמטר, אשר יהיה בשימוש כל עוד המשתמש לא הזין ערך אחר בעת הרצת היוניט ב-UI.
  + דוגמא לפרמטר המוגדר תחת השדה – parameters:



להלן דוגמא לקובץ קונפיגורציה ליוניט:

## unit\_<ddd>.m

זהו קובץ המטלב של היוניט.

קובץ זה מכיל את רק שורות המטלב המהוות את העיבוד ביוניט.

בזמן ריצה, נוצרים קבצי מטלב חדשים ע"י קבצי ה-bash, כאשר הם מורכבים מהגדרת המשתנים הדרושים ליוניט, התלויות של היוניט בספריות שונות, ולבסוף – שורות המטלב המוגדרות בקובץ זה.

### דרישות חשובות:

* יש לוודא שכל הפרמטרים שבשימוש קובץ המטלב – מוגדרים בקובץ ה-xml של היוניט, וכן שפרמטרים אלו נכתבים לקובץ המטלב הנוצר בזמן ריצה עם השמות המתאימים.
  + לדוגמא: אם קובץ המטלב unit\_<ddd>.m מכיל את השורה הבאה:

file\_source = fullfile(input\_dir,input\_file\_name)

יש לוודא שקובץ ה-bash של אותו יוניט מכיל את השורות הבאות בהתאם:

echo "input\_file\_name ='${currfilename}';" >> ${filename}

echo "inputDir='${INPUT\_DIR}';" >> ${filename}

בדוגמא המוצגת, הפרמטרים שהיו בשימוש הינם:

* + - input\_dir – אשר מוגדר לכל יוניט בקובץ הקונפיגורציה בשדה Input-files
    - input\_file\_name – אשר אינו מוגדר כפרמטר של יוניט, אלא נוצר ע"י קובץ ה-bash
  + אילו קובץ המטלב היה מכיל גם את השורה הבאה :

EEG = pop\_eegfilt( EEG, str2num(locutoff), str2num(hicutoff), [], str2num(revfilt), str2num(usefft), str2num(plotfreqz), firtype, str2num(causal));

היה עלינו לוודא :

* + - שקובץ ה-bash מכיל את השורות המתאימות, לדוגמא:

echo "locutoff='${LOCUTOFF}';" >> ${filename}

* + - שקובץ הקונפיגורציה unit\_<ddd>.xml של היוניט מכיל את הפרמטרים המתאימים. לדוגמא:

<parameter>

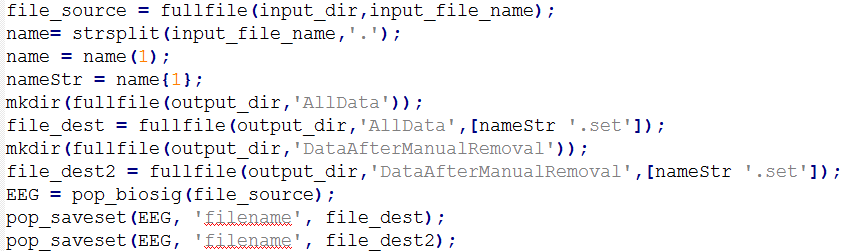
<name>locutoff</name>

<display>Low Cutoff</display>

<default>0.5</default>

<parameter>

להלן דוגמא לקובץ מטלב של יוניט:



## unit\_<ddd>.sh

זהו קובץ ה-bash של היוניט.

קובץ זה הינו הסקריפט שהמכונה מריצה בפועל, כאשר היא מעבירה לו את הארגומנטים שהוגדרו בקובץ הקונפיגורציות של היוניט, unit\_<ddd>.xml.

ניתן לייצר קובץ זה באופן עצמאי ולצרפו למערכת לתיקיית ה-units (כפי שמתבצע עבור שאר קבצי היוניט – קובץ הקונפיגורציה וקובץ המטלב),

וניתן גם לייצרו ב-UI באופן אוטומטי ע"י המערכת.

נציג כעת את שתי הדרכים ליצירת קובץ bash ואת ההנחיות המתאימות לכל דרך.

### קובץ bash הנוצר ע"י המשתמש ומתווסף למערכת ככלל קבצי היוניט

למשתמש יש אפשרות לצרף כל קובץ bash, העונה על הדרישות הבאות:

* ל-bash יש לפחות 3 פרמטרים, המופיעים בסדר הבא ובשמות הבאים:

TASK\_ID="$1"

INPUT\_DIR="$2"

OUTPUT\_DIR="$3"

* אם בקובץ הקונפיגורציה של היוניט unit\_<ddd>.xml מוגדרים פרמטרים נוספים, הם צריכים להופיע ב-bash באופן הבא:

TASK\_ID="$1"

INPUT\_DIR="$2"

OUTPUT\_DIR="$3"

LOCUTOFF="$4"

HICUTOFF="$5"

REVFILT="$6"

USEFFT="$7"

PLOTFREQZ="$8"

FIRTYPE="$9"

CAUSAL="$10"

* קבצי המטלב והלוגים שלהם הנוצרים ע"י קובץ ה-bash ישמרו בתיקייה bashTmp הנמצאת בתוך תיקיית ה-units.
  + שמות הקבצים הנ"ל יסתיימו במזהה המשימה:

\_${TASK\_ID}

לדוגמא:

currfilename=$(basename "$currfilename")

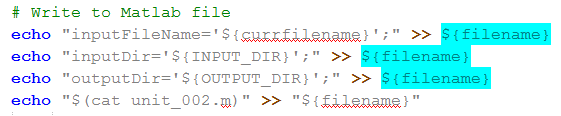
parmstr="${currfilename}"

filename="bashTmp/run\_${parmstr}\_task\_${TASK\_ID}.m"

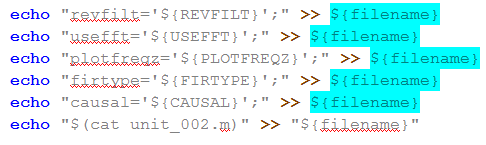
logfile="bashTmp/log\_${parmstr}\_task\_${TASK\_ID}.txt"

* קובץ ה-bash יסתיים בפקודה : wait
* קובץ ה-bash יכתוב לתוך קבצי המטלב אותן הוא מייצר את כל הפרמטרים הדרושים וערכם, וכן את תוכן קובץ המטלב של היוניט.

לדוגמא:

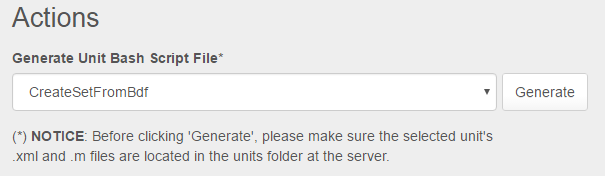
****

* כמובן שאם יש ליוניט פרמטרים נוספים, מלבד הפרמטרים הקבועים (נתיב הקלט והפלט), גם הם יופיע:

****

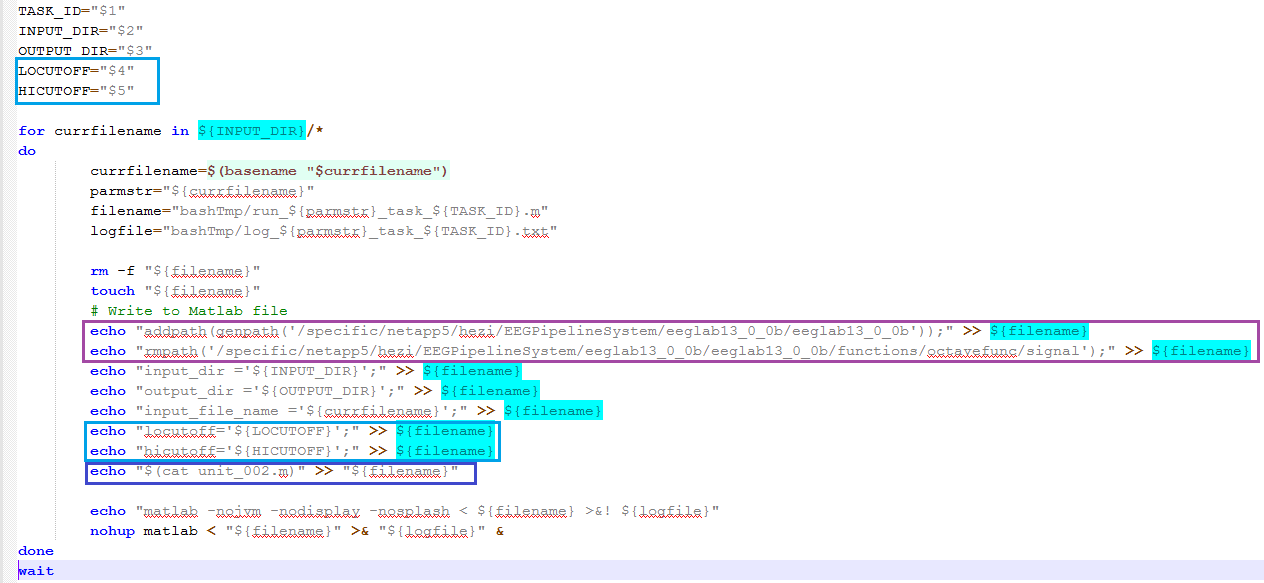
### קובץ bash הנוצר באופן אוטומטי בעזרת ה-UI

למשתמש יש אפשרות לבחור שהמערכת תייצר עבור היוניט את קובץ ה-bash.

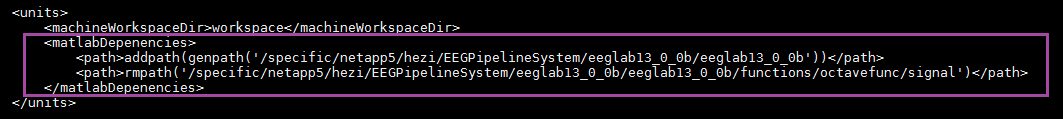


! חשוב לזכור – לפני לחיצה על Generate – יש לוודא שקובץ הקונפיגורציה של היוניט וקובץ המטלב של היוניט נמצאים בשרת בתיקיית units.

הקובץ נוצר במבנה הבא:

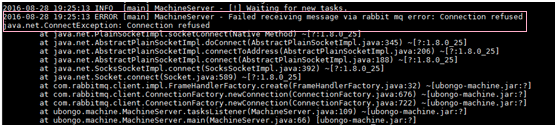


כאשר:

* הפרמטרים נלקחים מקובץ הקונפיגורציה של היוניט (מסומנים במלבן בצבע תכלת).
* שורות ה-dependencies של המטלב (מסומנות במלבן סגול) – מתווספות בהתאם להגדרת ה-matlabDependencies בקובץ הקונפיגורציה של המערכת (config/ubongo-config.xml):

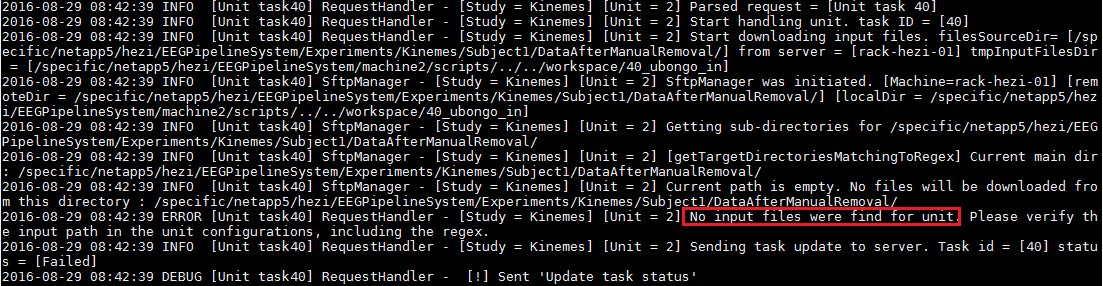
# תקלות נפוצות

1. שגיאה בלוג המכונה:



* פתרון: ה-RabbitMQ למטה – יש לפנות לצוות הסיסטם.
  + ברגע שצוות הסיסטם יעלו את ה-RabbitMq – המכונה תחזור לפעול בצורה תקינה.

1. שגיאה בלוג המכונה:

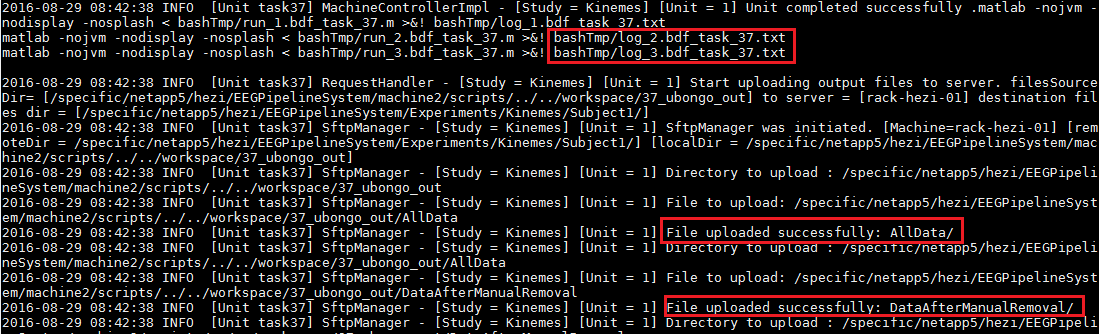


* במקרה זה ניתן לראות בלוג ש-unit2 נכשל.

במקרה המדובר ה-Flow התחיל ב-unit1, ולאחריו רץ unit2 כאשר קבצי הקלט של unit2 הוגדרו להיות קבצי הפלט של unit1.

לפי השגיאה המסומנת, לא נמצאו קבצי קלט מתאימים עבור unit2.

* לכן נתבונן במעלה הלוג ונבדוק את תוצאות הרצת unit1 בכדי להבין מדוע לא נוצרו עבורו קבצי הפלט המתאימים (שנועדו להיות קבצי הקלט של unit2):

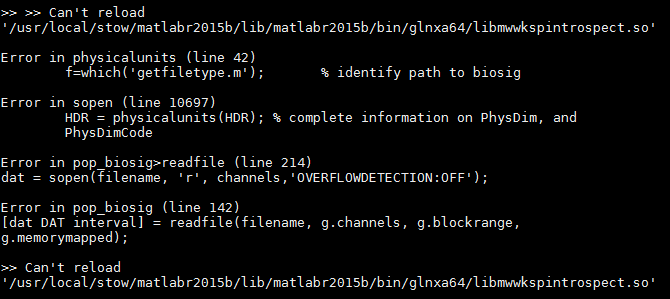


* לפי הלוג נראה שנוצרו ורצו קבצי מטלב, וכתוצאה מריצה זו נוצרו תיקיות פלט ע"י unit1, אך לא נוצרו קבצי פלט.
* לכן נתבונן בלוגים של ריצת המטלב, בנתיב:

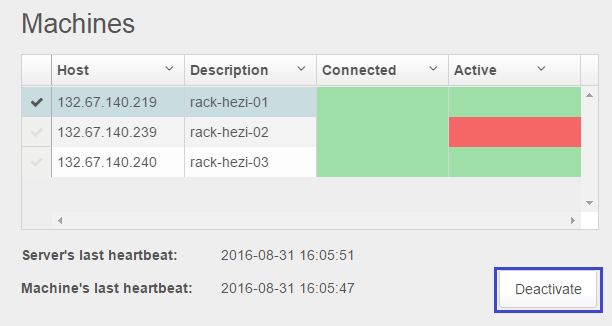
Units/bashTmp

כאשר בתיקייה זו נתבונן בקבצים שלהם מזהה המשימה המתאים כפי שמופיע בכל שורת לוג – task\_37

* אכן ניתן לזהות בעיות בריצה המטלב. מדובר בתקלה שאינה קשורה למערכת Ubongo, אלא למטלב המותקן על המכונה.



* פתרון : נוציא את מכונה זו ממערך המכונות ע"י בחירת המכונה ולחיצה על Deactivate:



ונריץ Flow חדש. ה-Flow החדש כבר לא יישלח למכונה התקולה.

במקביל, נבדוק מדוע לא ניתן לטעות את ספריות המטלב במכונה זו. כאשר התקלה תיפתר נוכל בקלות להחזיר את המכונה למערך המכונות של Ubongo.