**אילוצים:**

|  |
| --- |
| 1. לכל משתמש יש שם המזהה אותו באופן יחיד במערכת. |
| context User inv:  self.username->isUnique(username) |
| 2. לכל משתמש יש דרגה, המיוצגת כערך מספרי מ-1 עד 10. |
| context User inv:  self.rank <= 10 AND self.rank > 0 |
| 3. הרשאת כתיבה תינתן רק למשתמשים בעלי דרגה גבוה או שווה לזו של המשתמש המוגדר כאחראי על האירוע מטעם הארגון. |
| context Participate inv: self.permission = ‘in charge’  implies  self.Event.UserOrganization->select(user:UserOrganization | user.OrgName = self.UserOrganization.OrgName)->forAll(user:UserOrganization | (user.rank >= self.UserOrganization.rank AND user.Participate.premission=’write’) OR (user.rank < self.UserOrganization.rank AND user.Participate.premission=’read’)) |
| 4. כל משתמש בעל הרשאת כתיבה באירוע, יכול להעביר דיווח בעת הצורך. |
| context Report inv: self.sender.permission = "write" |
| 5. משתמש בארגון לא יכול להעביר דיווח למשתמש בארגון אליו הוא שייך. |
| context Participate inv: self.sender. orgName <> self.receiver.orgName |
| 6 משוב מהווה דירוג מספרי 1 עד 5. |
| context Feedback inv: self.rate <= 5 AND self.rate > 0 |
| 7. משתמש אינו יכול לדרג את עצמו. |
| context Feedback inv: self.gives -> collect(p: participate | p.username) -> excludes (self.username) |
| 8. משתמש יכול להעביר פקודה למשתמש בדרגה שמתחתיו באותו הארגון. |
| context Order inv: self.receiver.rank < self.sender.rank AND self.receiver.orgName = self.sender.orgName |
| 9. משתמש יכול לרשום תלונה על כל משתמש אחר בארגון שלו, מלבד ה-Admin. |
| context complain inv: self. Complainee.username <> self. Complainer.username AND  self.Comlainee.orgName = self.Complainer.orgName AND NOT  self.complainee.oclIsTypeOf(OrgAdmin) |
| 10. משתמש יכול לערוך רק עדכונים שהוא יצר. |
| context Update inv:  self.publish = self.first\_edit.publish |
| 11. משתמש יכול למלא משוב רק על המשתמשים שהוסיפו עדכונים לאירוע ולאחראי האירוע |
| context feedback inv:  self.event.updates -> collect(update: Update| update.publisher)as Set -> include(self.resiver) OR self.event.incharge -> include(self.resiver) |

**Events:**

1. **יצירת אירוע:**   
   - makeNewEvent(title, date, EOCpublisher, firstUpdate, status, users, categories): יצירת אובייקט "אירוע", עם האחראים עליו.
2. **שליחת דיווח:**

* CreateNewReport(receiverID, senderID, eventID, description): יצירת דיווח חדש עם התיאור הנדרש.
* SendReport(report): שליחת דיווח למשתמש הרצוי מארגון אחר, ומינויו לאחראי על האירוע מטעם הארגון אליו הוא שייך.

1. **כתיבת תלונה:**

* createComplaint(complaineeID, complainerID, description): יצירת תלונה חדשה.
* sendComplaintToApprooved(admin, complain): שליחת התלונה למשתמש ה- Admin של הארגון.
* changeComplaintStatus(complain): משנה את סטטוס התלונה. במידת הצורך נשלח למשתמש שעליו התלוננו ודרגתו תשונה במידת הצורך.

1. **יצירת עדכון:**

* addUpdate(user, date, event, description): יצירת עדכון חדש וקישורו לעדכונים קודמים.

1. **עריכת עדכון:**

* editUpdate(userID, update, newDetails): עריכת העדכון בפרטים החדשים.

1. **יצירת קטגוריה:**

* createNewCategory(adminUser, name, description): יצירת קטגוריה חדשה ע"י משתמש ה- Admin של המוקד.

1. **דירוג משתמש ושינוי דרגה בהתאם:**

* sendFeedbacksAllUsers(eventID): שליחת משוב לכל המשתמשים שלקחו חלק באירוע.
* Rank(users, event): מענה על משוב
* sendReport(feedbacks, event): שליחת הדו"ח למשתמשי ה- admin של כל ארגון שלקח חלק באירוע.
* AdminRankChanges(Report \*,event description) : מעדכן את הדירוג של המשתמשים אשר הגיעו לכמות אזהרות הדורשת עדכון.

Report \* הוא מבנה נתונים המכיל בעבור כל משתמש את הדירוג הממוצע שלו.

**חוזים:**

1. **יצירת אירוע:**
   1. Operation: makeNewEvent(title, date, EOCpublisherID, firstUpdate, status, users,categories)

Reference: Use Case: Create Event

Preconditions:

Postconditions:

* A event instance e was created.
* e was associated with the in-charge users and the categories that fit the event.
* e was associated to the EOC

1. **שליחת דיווח:**
   1. Operation: SendEvent(receiverID, senderID, eventID)

Reference: Use Case: Send Report

Preconditions:

Postconditions:

* A new Participate instance p has been created.
* p associated with receiver, with the sender and with the correct event.
* p’s permission field set to “in charge”.

1. **כתיבת תלונה:**
   1. Operation: createComplaint(complaineeID, complainerID, description)

Reference: Use Case: Complaint

Preconditions:

* Complainee is not an admin.
* Complainee and complainer are in the same emergency organization.

Postconditions:

* Complaint created.
* Complainer association to complaint created.
* Complaint association to complaint created.
* Association between complaint and the relevant admin is created.
  1. Operation: changeComplainStatus(complain, boolApprove)

Reference: Use Case: Complaint

Preconditions:

* Complaint is created
* complaint association to admin created

Postconditions:

* Complaint status is changed
* Case: complaint approved and complainee rank should change:
  + Complainee rank changes
  + If needed complainee account becomes “disabled”

1. **יצירת עדכון:**
   1. Operation: addUpdate(user, date, event, description)

Reference: Use Case: Write Update

Preconditions:

* Event exists.
* User has a write permission.

Postconditions:

* Update created.
* Update association to the event created.
* Update association to the publisher created**.**

1. **עריכת עדכון:**
   1. Operation: editUpdate(userID, update, newDetails)

Reference: Use Case: Edit Update

Preconditions:

* Update exists.

Postconditions:

* update created.
* update association to the event created.
* update association to the publisher created.
* update association to original update created.

1. **יצירת קטגוריה:**
   1. Operation: createNewCategory(adminUser, name, description)

Reference: Use Case: Create Category

Preconditions:

Postconditions:

* Category created.
* Category association to EOC representative admin created.

1. **דירוג משתמש ושינוי דרגה בהתאם:**
   1. Operation: sendFeedbacksAllUsers (eventID)

Reference: Use Case: Rank Colleagues

Preconditions:

* Event exists
* The Emergency Organization(s) users which fill feedback are associated to the event
* The Emergency Organization(s) users who have feedback on them are associated to an update of the event
* Event status changed to “Treatment is over”

Postconditions:

* all users associated to the event get feedbacks to fill
  1. Operation: Rank(users, event)

Reference: Use Case: Rank Colleagues

Preconditions:

* Event exists
* All users associated to the event get feedbacks to fill

Postconditions:

* All users that participated in the event returned feedbacks
  1. Operation: sendReport(feedbacks, event)

Reference: Use Case: Rank Colleagues

Preconditions:

* All users that participated in the event returned feedbacks

Postconditions:

* All admins associated to the event get summarized report
  1. Operation: AdminRankChanges(Report\*, eventDescription)

Reference: Use Case: Rank Colleagues, Use Case: Change Rank

Preconditions:

* All admins associated to the event got summarized report

Postconditions:

* For all users in admin organization if rank change was needed, rank changed
* For all users in admin organization if account status change was needed, account status