Assignment Replacing Course Exam

עדן אפרים 316429307

מרחב הבעיה (Problem domain)

המטרה היא לתת מענה למשתמשים בעלי עסק או מוסד מסוים שבעזרת התוכנה יוכלו לייעל , לקצר ולשפר את עבודתם. פעולות ידניות שלקחו זמן וכוח אדם נוסף בעבר יוכלו להתבצע "לייב" במערכת בה המשתמש יוכל לקבל באופן ישיר ומקוון נתונים ועדכונים שוטפים ממקורות רבים במקום המבוקש בצורה מסודרת וויזואלית, מה שייצור נוחיות רבה למשתמש.

בנוסף המערכת תנסה להגביל את המשתמש כמה שפחות בחופש תנועתו בתוכנה ,ככה שגם יעילות השימוש במערכת לצרכים שונים מוגבלת כמה שפחות אך גם אינה פוגעת באותה יעילות.

המערכת המוצעת היא מערכת של ארגון ובקרה דינאמי המתקשרת עם המשתמש באמצעות אינטרפייס פשוט ונותנת מבט ויזואלי נגיש על אזורים (או Area) בעסק או מוסד שבו נמצא/ים המשתמש/ים.

בשלב הראשון המשתמש במערכת יתבקש לדמות ולעצב את האזור הרצוי בתוך המערכת, אזור רצוי יוכל לדוגמא להיות מחלקה מסוימת בבית חולים בו המערכת תציג ויזואלית את המחלקה , אזור האחיות וחדרי המיטות של המטופלים או מסעדה בעלת שולחנות ובר בה המערכת תציג את אזורי המסעדה המחולקים לפי אזורים שונים לפי צורך המקום , האובייקטים יכולים להיבחר באופן שרירותי וניתנת להם משמעות והרשאות לפעולות מסוימות על פי תפקידים על ידי המשתמש כך שהמערכת היא רב שימושית ולא מחייב שימוש במקומות ספציפיים.

אזורים שונים יוכלו להיות מקושרים בתוך המערכת כדי לדמות קומות וחדרים סמוכים או מחולקים.

למערכת יועברו ב "live" עדכונים לגבי האובייקטים והמידע יעודכן בענן כך ששחקן חיצוני או מודולים במערכת יוכלו לראות את העדכונים האלו על גבי צג.

עדכונים לגבי אובייקטים יבוצעו באמצעות המשתמש ובנוסף תוצע אפשרות לקבל עדכונים ממערכת חיצונית בה יוכלו לקבל עדכונים אוטומטים.

לדוגמא, כאשר יש מיטה תפוסה בחדר בבית חולים האחות תידרש לשנות את אובייקט המיטה בחדר הספציפי לסטטוס "תפוס" או לשנות את אובייקט המיטה במערכת לצבע שונה כדי לתאר שהמיטה תפוסה על ידי מטופל ויהיה ניתן לראות זאת ויזואלית מאחר והחדר עוצב במערכת ומוצג.

לדוגמא נוספת על אותו עיקרון אך במימוש המערכת במסעדה, מלצר יסמן שולחן פנוי במערכת במיקום השולחן המתאים והמידע הזה יעודכן אצל כל משתמשי המערכת, כלומר העובדים כך שיוכלו להיות מסונכרנים לגבי איזה השולחנות בכל רגע נתון. כך למשל המאחרת תוכל להביט בתוכנה ולראות ויזואלית איפה יש מקום פנוי או אפילו להציג ללקוחות מקומות פנויים בצג ועל פיהם יוכלו לבקש היכן לשבת או לאיזה שולחן הם מעוניינים להמתין .

מכיוון שלא רק הבקרה דינאמית אלא גם הארגון או עיצוב החדר הינו דינאמי אז במידה ויש שינוי, כמו לדוגמא הוספה/הוסרה של שולחן יהיה ניתן לעדכן או לשנות פעולות אלו בקלות.

במקרה של עדכון חדר נוכל לבחור לתת רק למשתמש "המנהל" את האפשרות לערוך חדר ולעומת זאת למשתמש "עובד" לא יהיה את האפשרות הזו, בדוגמא הזו אפשר לראות שבין סוגי המשתמשים יש היררכיה, פעולות שעושה מלצר המנהל כמובן יוכל לעשות אך לא להפך.

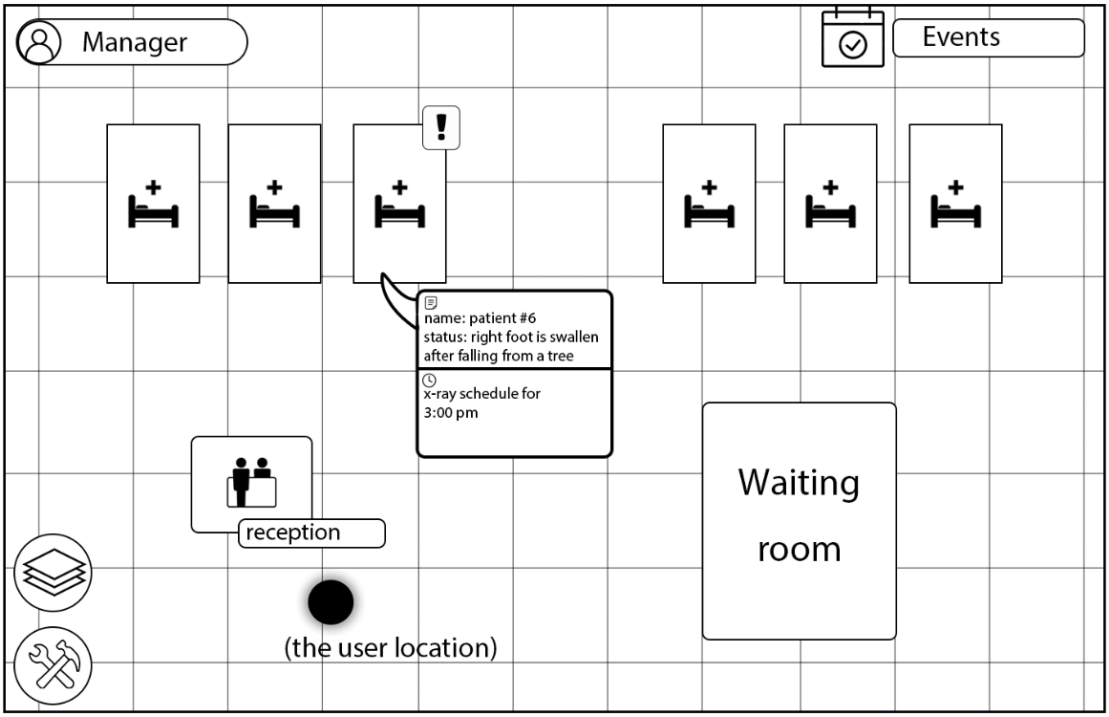
אפשר ליצור רשימת סוגי events (גם בשם אירועים או אוונטים) שיש להם משמעות הניתנת בהתאמה אישית ע"י המשתמש ולשמור אותם בתוכנה על פי דרישותיו.

את האירועים יהיה ניתן לקשר לאובייקטים בחדר ומכיוון שהם נוצרו בהתאמה אישית לא תהיה הגבלה על מה הם ייצגו. לדוגמא, במסעדה יהיה אפשר ליצור איוונט שניקרא "הזמנה", לקשר אותו לאובייקט או שולחן מתאים ובתור הערה בתוך האיוונט לרשום את פרטי ההזמנה כך ששאר העובדים יוכלו לראות את פרטי ההזמנה של אותו שולחן כדי להיות בבקרה ומתואמים ביניהם.

בתוך התוכנה יהיה ניתן לבחור איזה איוונטים להציג באופן אינדיווידואלי למשתמש. למשל בדוגמת המסעדה, המארחת לא בהכרח תצטרך לראות את טיב ההזמנה של אותו שולחן אך כן תצטרך

לראות האם השולחן מאויש או לא , לכן על פי התאמה ועיצוב המשתמש יוכל לבחור איזה משתמש יקבל גישה והרשאה לאיזה פעולות על פי ראות עיניו ונוחיותו.

UI concept (Hospital room example)



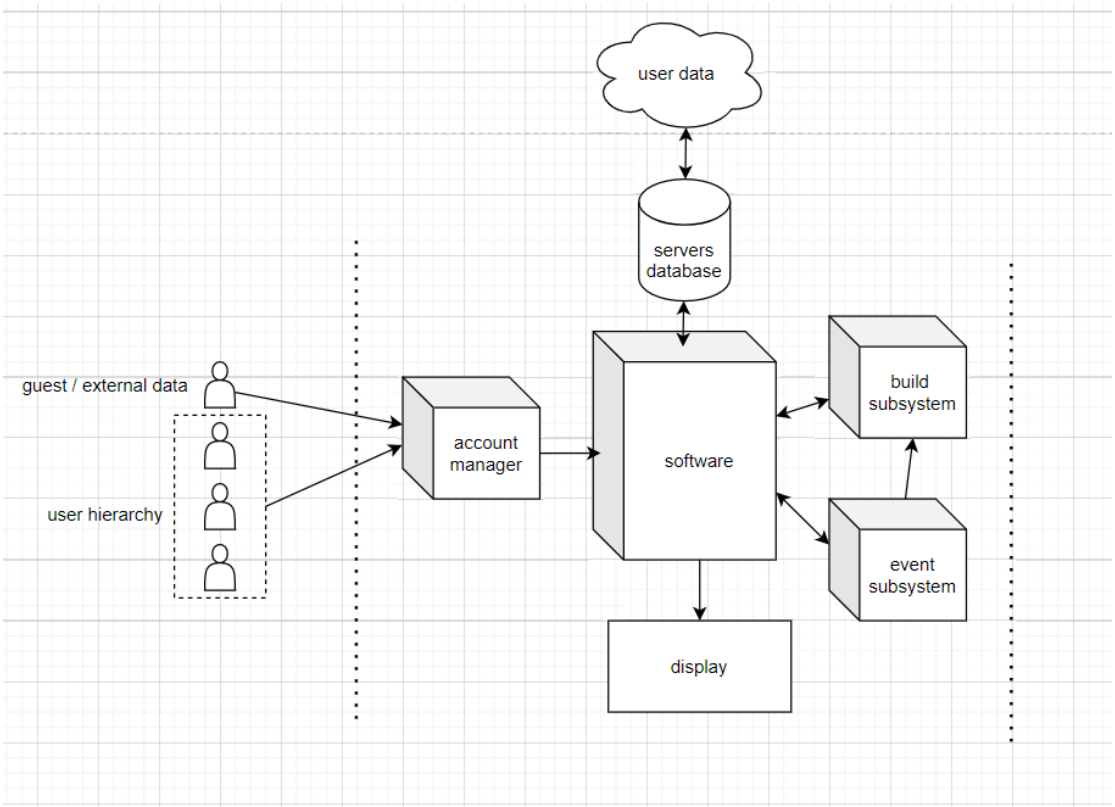
Functional requirements

Requirement	priority	Identifier
המערכת תאפשר מיפוי של אזור בצורה פשוטה וגנרית.	5	REQ1
המערכת תאפשר קישור בין אזורים מרובים לייצוג אזור גדול יותר שמחולק לאזורים מרובים.	2	REQ2
האינטרפייס יהיה נגיש לכמות משתתפים ותעדכן את המידע שנערך באופן שותף ודינאמי.	5	REQ3
המערכת תאפשר יצירת סוגי אירועים כלליים ולתת להם משמעות (ותאפשר פעולות פשוטות).	5	REQ4
המערכת תאפשר שיוך סוגי אירועים שונים לאובייקט מתוך אובייקטים מאופשרים במפה וקבלת מידע על אירועים בעת גישה לאובייקט.	4	REQ5
המערכת תאפשר סוגי גישה בדרגות שונות למשתמשים למטרת עריכה והוספת מידע.	3	REQ6
המערכת תאפשר הצגה או הסתרה של אירועים לדרגות שונות של משתתפים	2	REQ7
המערכת תוכל לשייך גורם חיצוני לאובייקט ולאפשר לו לעדכן מידע באירוע המשויך אליו (למשל משתמש אורח או חיישן המעביר מידע).	2	REQ8

Quality requirements

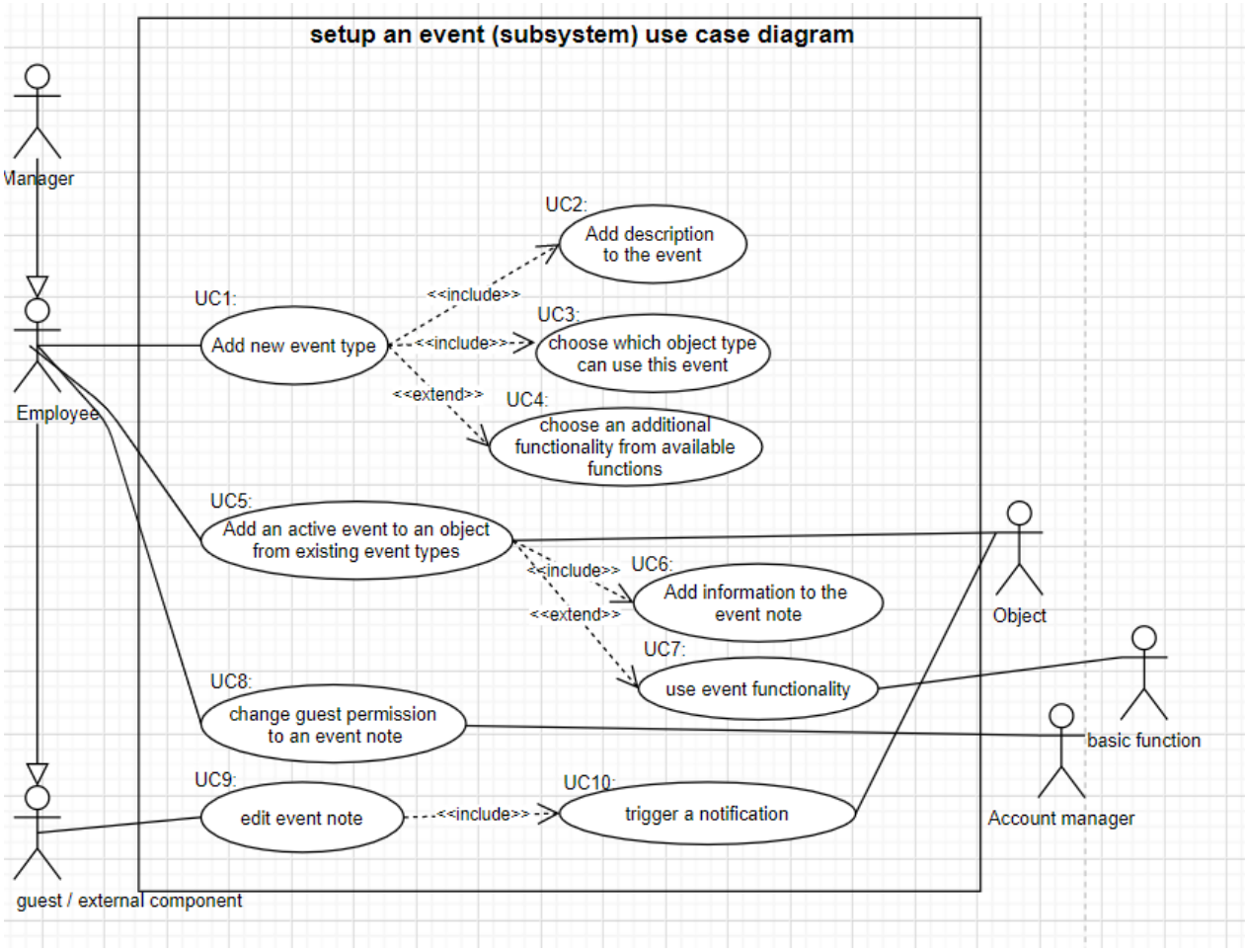
Requirements
המערכת צריכה להיות בעלת יכולת לעדכן את המידע אצל המשתמש באופן מהיר של שניות בודדות או פחות מכך.
המערכת צריכה להיות מסוגלת להכיל הרבה מידע בו זמנית.
המערכת צריכה להיות מסוגלת לעמוד במספר מרובה של משתמשים המחוברים בבת אחת מבלי לקרוס.
המערכת שואפת להיות רב שימושית וללא הרבה הגבלות למשתמש.
עדכון והעברת המידע צריך להיות קל ,מהיר ופשוט למשתמש.
האינטרפייס צריך להיות ידידותי ולא מסורבל למשתמש חדש בשימוש היום יומי.
המערכת צריכה להיות מאובטחת כדי לא לאפשר שינויים לא רצויים במידע, יכולות העריכה מוגבלות בדרגות שונות של משתמשים.
המערכת צריכה גישה יציבה לאינטרנט כדי לעדכן אינפורמציה באופן שותף בלייב.

System Architecture diagram

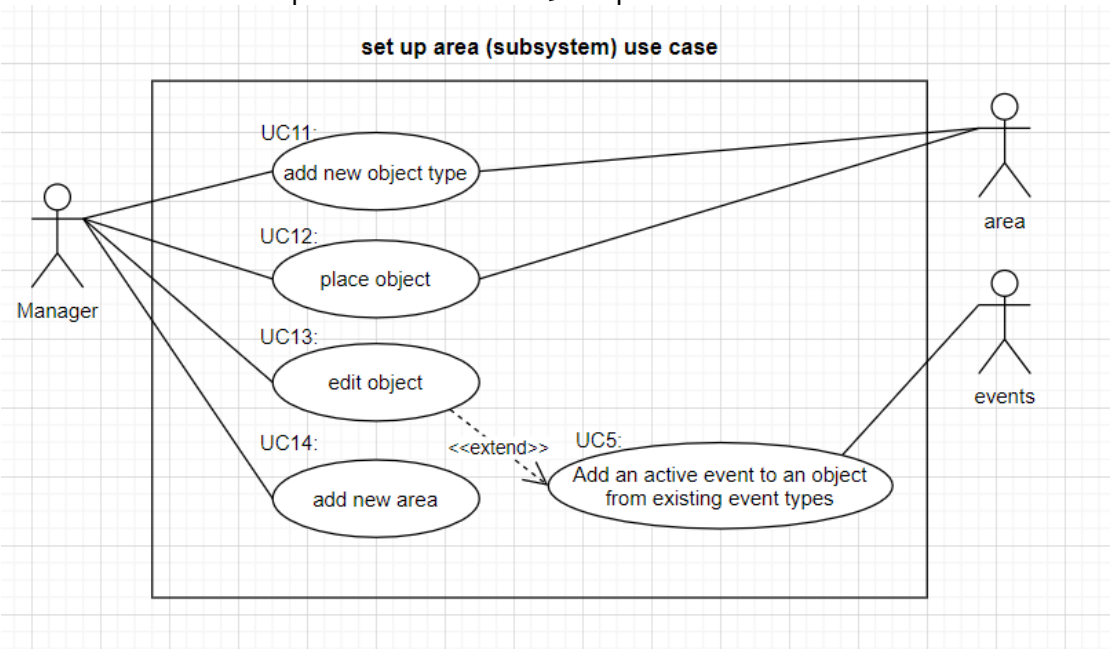


Requirements specifications

בדיאגרמה הבאה מתואר תהליך הוספת איוונט למערכת, עריכה ושימוש שלו



בדיאגרמה הבאה מתואר תהליך של עריכת אזור והוספת אובייקטים לאזור.



UC Diagram Glossary

Use Case UC-1: Add new event type
דרישות קשורות: REQ1,4,5 **שחקנים:** Manager, Employee **מטרתם:** המנהל והעובד יוכלו לבחור ליצור סוג חדש של אוונט אשר יתווסף לרשימת האוונטים של המערכת באזור עליו קבוצת המשתמשים משתמשים בו. **מצבים מקדימים:** למנהל צריך להיות לפחות אזור אחד פתוח. **מצבים הבאים:** הוספת משמעות לאוונט בUC2 **זרימת אירועים למקרה הצלחה:** 1. המנהל/עובד יבחרו באפשרות יצירת אוונט. 2. המשך לעריכת האוונט.

Use Case UC-2: Add description to the event (sub use case)
דרישות קשורות: REQ3,4,6 **שחקנים:** Manager, Employee **מטרתם:** לאחר יצירת האוונט המשתמש ייתן לאוונט שם שיתאר אותו ותיאור נוסף שיתאר את המטרה שלו. **מצבים מקדימים:** המשתמש יהיה חייב להוסיף תיאור בעת יצירת האוונט. **מצבים הבאים:** מרגע יצירת האוונט במלואו, המשתמשים יוכלו להוסיף אותו על אוביקט במפה ולרשום בו מידע בתוך Note. **זרימת אירועים למקרה הצלחה:** 1. המנהל/עובד יבחרו באפשרות יצירת אוונט. 2. המשתמש יבחר שם שייצג את האוונט הזה ויתן לו משמעות. 3. המשתמש יוסיף לאוונט תיאור שיסביר מה מטרתו. 4. המשך הגדרת האוונט.

Use Case UC-3: choose which object type can use this event (sub use case)
דרישות קשורות: REQ4,5,6 **שחקנים:** Manager, Employee **מטרתם:** לבחור לאיזה מן האויבקטים הקיימים יהיה ניתן להוסיף סוג אוונט זה. **מצבים מקדימים:** המשתמש יהיה חייב לבחור את האובייקטים הרצויים בעת יצירת האוונט. **מצבים הבאים:** מרגע יצירת האוונט במלואו, המשתמשים יוכלו להוסיף אותו על אובייקט במפה ולרשום בו מידע, ניתן יהיה לערוך את הרשימה לאחר סיום ההגדרה. **זרימת אירועים למקרה הצלחה:** **1.** המנהל/עובד יבחרו באפשרות יצירת אוונט. **2.** המשתמש יבחר שם שייצג את האוונט הזה ויתן לו משמעות. **3.** המשתמש יוסיף לאוונט תיאור שיסביר מה מטרתו. **4.** המשתמש יבחר לאיזה סוג של אוביקטים מרשימת האוביקטים הקיימים באזור שהוגדרו על ידי שחקן מסוג מנהל יוכלו להשתמש באוונט זה. **5.** האוונט יתווסף לרשימת האוונטים השייכים לאזור וניתן יהיה להשתמש בו כעת.

Use Case UC-4: choose an additional functionality from available functions (sub use case)
דרישות קשורות: REQ4,6 **שחקנים:** Manager, Employee **מטרתם:** מנהל או עובד יוכלו לבחור אם להוסיף לאוונט פונקציה נוספת מפעולות בסיסיות אשר כבר קיימות בו. **מצבים מקדימים:** חובה שיהיה אוונט קיים עליו ירצו להוסיף פונקציונליות. **מצבים הבאים:** משתמשים יוכלו להשתמש בפונקציה בתוך האוונט, פונקציות אפשריות יתווספו לרשימה . פונקציות אפשרויות לדוג' הם :טיימר (במקרה זה נוסיף שחקן משתתף), מחשבון ופעולות נוספות אחרות. **זרימת אירועים למקרה מורחב:** **1.** המנהל/עובד יבחרו בהגדרת אוונט. **2.** יבחרו להוסיף לו פונקציונליות נוספת. **3.** יבחר בפונקצית הטיימר. **4.** לאוונט הנוכחי יהיה אפשרות להשתמש בפונקציית הטיימר.

Use Case UC-5: Add an active event to an object from existing event types
+Use Case UC-6 Add information to the event note (subcase)
+Use Case UC-7 use event functionality (subcase)

דרישות קשורות: REQ3,4.5,6,7

שחקנים: Manager, Employee

שחקנים משניים: Object, basic function(for UC-7 only)

מטרתם: מנהל או עובד יוכלו לבחור אובייקט במפה ולהוסיף אירוע פעיל מהסוג של אותו אירוע ,בו ירשמו מידע הקשור לאובייקט הנבחר. במידה ולאוונט נוסף פונקציונאליות נוספת המשתמש יוכל להשתמש גם בה.

מצבים מקדימים: חובה שיהיה קיים אובייקט בו האוונט מאפשר להשתמש בו.

מצבים הבאים: משתמשים יוכלו לראות את אותו מידע ועבור חלק מהמשתמשים יהיה אפשרות גם לערוך אותו.

זרימת אירועים למקרה שימוש מורחב באוונט (לדוגמא אוונט עם פונקציית הטיימר):

1. המנהל/עובד יבחרו באוביקט בו האוונט מאופשר.
2. המהל/עובד יוסיף אירוע פעיל מסוג אותו האוונט.
3. יכתבו בו תיאור ב note ויוסיפו התראה לשעה ספציפית.
   (במקרה זה השחקן basic function = timer).
4. כשיגיע השעה הנתונה שהוכנסה בטיימר, על האובייקט תופיע התראה מתאימה. (שגם את ההתראה יגדירו באותה פונקציה).

Use Case UC-8: change guest permission to an event note

דרישות קשורות: REQ6,7,8

שחקנים: Manager, Employee

שחקנים משניים: Account manager

מטרתם: מנהל או עובד יוכלו לבחור אם אורח ספציפי יהיה בעל גישה לשנות את המידע שנמצא ב -Note של האוונט הנמצא בו, או אם כל גורם חיצוני אחר יוכל לערוך את אותו המידע.
האורח לא יוכל לשנות את משמעות האוונט, אלא רק את המידע שהוא מכיל באובייקט שבו הוא מחובר אליו.

מצבים מקדימים: חובה שיהיה אוונט פעיל שמחובר לאובייקט.

מצבים הבאים: משתמשים יוכלו לראות את אותו מידע, אורח או גורם חיצוני אשר יוכלו לשנות את המידע בnote בנוסף למנהל או עובד.

זרימת אירועים למקרה הצלחה:

1. המנהל/עובד יבחרו באובייקט בו האוונט מאופשר.
2. המהל/עובד יוסיף לאובייקט משתמש/ים אורח.
3. המשתמש עכשיו מקושר לאובייקט.

Use Case UC-9: edit event note

דרישות קשורות: REQ3,4,5,6,8

שחקנים: Manager, Employee, guest

מטרתם: מנהל/עובד/אורח עם הרשאות יוכלו לשנות מידע הרשום ב Note שמחובר לאובייקט באוונט מסויים.

שאר המשתמשים יוכלו לראות מידע זה בUI כאשר יבחרו באובייקט.

מצבים מקדימים: חובה שיהיה אוונט פעיל שמחובר לאובייקט.

מצבים הבאים: משתמשים יוכלו לראות את אותו מידע, אורח או גורם חיצוני בעלי אפשרות לשנות את המידע בnote בנוסף למנהל או עובד.

זרימת אירועים למקרה הצלחה:

1. המנהל/עובד או אורח עם הרשאות יבחרו באובייקט בו האוונט מאופשר.
2. יבחרו באוונט מהאוונטים המקושרים לאותו אוונט.
3. יבחרו לערוך את ה note וישנו בו מידע על פי צורכם.
4. כל שאר המשתמשים יוכלו לראות את השינוי.

Use Case UC-10: trigger a notification (subcase)

דרישות קשורות: REQ3,6

שחקנים: Manager, Employee, guest

שחקנים משניים: Object

מטרתם: שינוי מידע באוונט, יתריע על שינוי באמצעות notification על הצג .

מצבים מקדימים: חובה שיהיה אוונט פעיל שמחובר לאובייקט.

מצבים הבאים: משתמשים יוכלו לראות את אותו מידע אם יש להם גישה לכך.

זרימת אירועים למקרה הצלחה:

1. משתמש כלשהו שינה מידע באוונט.
2. כל המשתמשים יקבלו התראה על האובייקט אליו מקושר האוונט.
3. המשתמשים האחרים יוכלו לפעול בהתאם לאותה הערה.

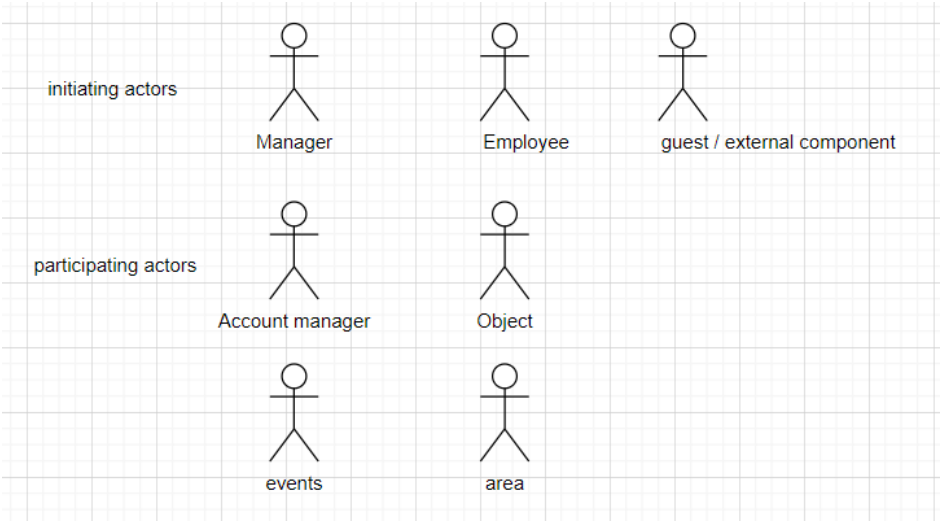
Use Case UC-11: add new object type
דרישות קשורות: REQ1,3,6,8 **שחקנים:** Manager **שחקנים משניים:** Area **מטרתם:** מנהל יוכל להוסיף סוג חדש של אובייקט ולתת לו משמעות על פי דרישותיו וצרכיו. (לבחור שם, צורה, אייקון וכו'..) **מצבים מקדימים:** חובה שיהיה אזור פתוח שמייצג את האזור שהמשתמש רוצה ליצור. **מצבים הבאים:** האובייקט החדש ייצג משהו מהעולם האמיתי והמנהל יוכל להניח את אותו אובייקט באזור הווירטואלי כדי לתאר אותו טוב ככל הניתן. **זרימת אירועים למקרה הצלחה:** 1. מנהל בוחר ליצור סוג אובייקט חדש. 2. מעניק לו שם ותיאור. 3. האובייקט החדש זמין ברשימת האובייקטים להתאמה וסידור באזור.

Use Case UC-12: place object
דרישות קשורות: REQ1,3 **שחקנים:** Manager **שחקנים משניים:** Area **מטרתם:** מנהל יוכל לבחור אובייקט מרשימת האובייקטים שנוצרו ולבחור איפה להניח אותו באזור המבוקש. **מצבים מקדימים:** חובה שסוג האובייקט הרצוי יהיה קיים במערכת. **מצבים הבאים:** עובד יוכל להוסיף לאובייקט אירועים אם ירצה , משתמשים יוכלו לבחור באותו אובייקט כדי לראות מידע עליו. **זרימת אירועים למקרה הצלחה:** 1. מנהל בוחר אובייקט מן הרשימה. 2. מנהל מניח וממקם אותו באזור במקום המתאים.

Use Case UC-13: edit object
דרישות קשורות: REQ1,3,8 **שחקנים:** Manager **מטרתם:** כל אפשרות שהוצעה בעת יצירת האובייקט תאופשר גם לשינוי בעת עריכה. השינוי יעדכן ויופיע על כל האובייקטים מסוג זה שהונחו כבר כך שתישאר אחידות לעיצוב האזור ובנוסף יופיע גם על אובייקטים חדשים שייווצרו בעתיד. **מצבים מקדימים:** חובה שיהיה אובייקט קיים. **מצבים הבאים:** השינוי יעשה על כל האובייקטים מאותו סוג שקיימים וגם לאלה שיונחו בעתיד. **זרימת אירועים למקרה הצלחה:** 1. המנהל יוכל לערוך אובייקטים שיצר מתוך רשימת האובייקטים הנתונים. 2. ישמור את השינויים ויעדכנם אם נדרש במקומות הנדרשים.

Use Case UC-14: add new area
דרישות קשורות: REQ1,REQ2,REQ5 **שחקנים:** Manager **שחקנים משניים:** Events **מטרתם:** הוספת אזור חדש שייצג אזור וירטואלי (בד"כ יתאר את העולם האמיתי), אזור זה יהווה את המפה שעליה יהיה אפשר להניח את האובייקטים ולהגדיר בהם אירועים. בנוסף אפשר להוסיף אזור מקושר לאזור שנוצר שיחלוק את סוגי האובייקטים ואירועים. **מצבים מקדימים:** משתמש מנהל מחובר. **מצבים הבאים:** דרוש משתמש מנהל כדי להתחיל לערוך את המפה ולהוסיף משמעות לאזור. **זרימת אירועים למקרה הצלחה:** 1. מנהל מתחבר למערכת. 2. יוצר אזור חדש. 3. מקשר אזור אחד או יותר לאזור החדש על פי צרכיו. 4. ממשיך ביצירת המפה.

Actors Table



Actor table for Event subsystem UC diagram

Actors	Goal	Use Case
Manager	המטרה הינה יצירת סוג חדש של אובייקט שידמה אובייקט בעל משמעות אמיתית.	UC11: add new object type
Manager	הנחת אובייקט באזור שיתאר אובייקט רצוי בעל משמעות.	UC12: place object
Manager	שינוי אחד מן האובייקטים הקיימים ועדכונו.	UC13: edit object
Manager	יצירת האזור עליו יניחו את כל האובייקטים הנדרשים ותיאור המפה הווירטואלית של האזור.	UC14: add new area
Manager	הוספת סוג חדש של אירוע כאשר מחויבים להוסיף לו תיאור ובחירה איזה מן האובייקטים באזור יוכלו להשתמש בו.	UC1: add new event type UC2: Add description to the event UC3: choose which object type can use this event
Manager	להשתמש באירוע שהוגדר, להוסיף שימוש פעיל שלו באובייקט באזור עליו עובדים. חובה להוסיף אינפורמציה בעת הוספת האירוע לאובייקט.	UC5: Add an active event to an object from existing event types UC6: Add information to the event note
Manager	המטרה להשתמש בפונקציה הנבחרת לאירוע על מנת לבצע פעולה ולעדכן את האירוע.	UC7: use event functionality
Manager	המטרה היא לשנות הרשאות למי שיכול לעדכן את הפרטים בפרטי האירוע המחובר לאובייקט.	UC8: change guest permissions to an event note
Manager	המטרה היא לערוך את פרטי האוונט הרצוי באובייקט הנבחר. שינוי בפרטי האוונט יגרום להתראה על האובייקט.	UC9: edit event note UC10: trigger a notification
Employee	המטרה להוסיף סוג חדש של אירוע, יהיה מחויב להוסיף לו תיאור ולבחור איזה מן האובייקטים באזור יוכלו להשתמש בו.	UC1: add new event type UC2: Add description to the event UC3: choose which object type can use this event
Employee	המטרה להשתמש באירוע שהוגדר, להוסיף שימוש פעיל שלו באובייקט באזור עליו עובדים. יש חיוב להוסיף אינפורמציה בעת הוספת האירוע לאובייקט.	UC5: Add an active event to an object from existing event types : Add information to the event 6UC note
Employee	המטרה להשתמש בפונקציה הנבחרת לאירוע כדי לבצע פעולה ולעדכן את האירוע.	UC7: use event functionality
Employee	המטרה היא לשנות הרשאות למי יכול לעדכן פרטים בפרטי האירוע המחובר לאובייקט.	UC8: change guest permissions to an event note

Employee	המטרה היא לערוך את פרטי האוונט הרצוי באובייקט הנבחר. שינוי בפרטי אוונט יגרום להתראה על האובייקט.	UC9: edit event note UC10: trigger a notification
guest	המטרה היא לערוך את פרטי האוונט הרצוי באובייקט הנבחר. שינוי בפרטי אוונט יגרום להתראה על האובייקט.	UC9: edit event note UC10: trigger a notification

Traceability Matrix

Req	PW	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8	UC9	UC10	UC11	UC12	UC13	UC14
REQ1	5	X										X	X	X	X
REQ2	2														X
REQ3	5		X			X	X	X		X	X	X	X	X	
REQ4	5	X	X	X	X	X	X	X		X					X
REQ5	4	X		X		X	X	X		X					
REQ6	3		X	X	X	X	X	X	X	X	X	X			
REQ7	2					X			X						
REQ8	2								X	X		X		X	
Max PW		5	5	5	5	5	5	5	3	5	5	5	5	5	5
Total PW		14	13	12	8	19	17	17	7	19	8	15	10	10	12

Traceability Matrix Conclusions

כפי שניתן לראות ה- Requirements עם העדיפות הגבוהה ביותר הן REQ4, REQ3,REQ1.

REQ1 קיבלה עדיפות גבוהה מבין היתר וזאת מפני שחלק מיעילות המערכת היא למפות את האזור בצורה פשוטה וגנרית, זהו בעצם "מרחב הבעיה" אותו המערכת באה לפתור.

REQ3 גם דרישה זו בעלת תיעדוף גבוה במערכת וזה מכיוון שיש צורך וחשיבות לכך שהמערכת תהיה נגישה לכמות משתתפים ותעדכן את המידע שנערך באינטרפייס באופן דינאמי כך שלא יוצרו אי התאמות בין משתתפים שונים במערכת לכן יש עדיפות לנגישות ועדכון המידע בה .

REQ4 דרישה זו קיבלה גם היא תיעדוף גבוה וזאת מפני שכדי שהמשתמש יוכל בכלל להתחיל להשתמש במערכת, להתייעל ממנה ולקצור את פירות השימוש בה, הוא צריך ליצור את האירועים השונים שישמשו לנוחיותו במערכת.

כל הדרישות הללו הן אבני הבניין של המערכת לתפקודה ולשימוש בה ולכן קיבלו גם תיעדוף מגבוה מפני יתר הדרישות.

לאחר בניית ה – Traceability Matrix ושקלול משקלי העדיפות ניתן לראות כי ה - Use Cases החשובים ביותר הינם : UC5,UC6, UC7, UC9.

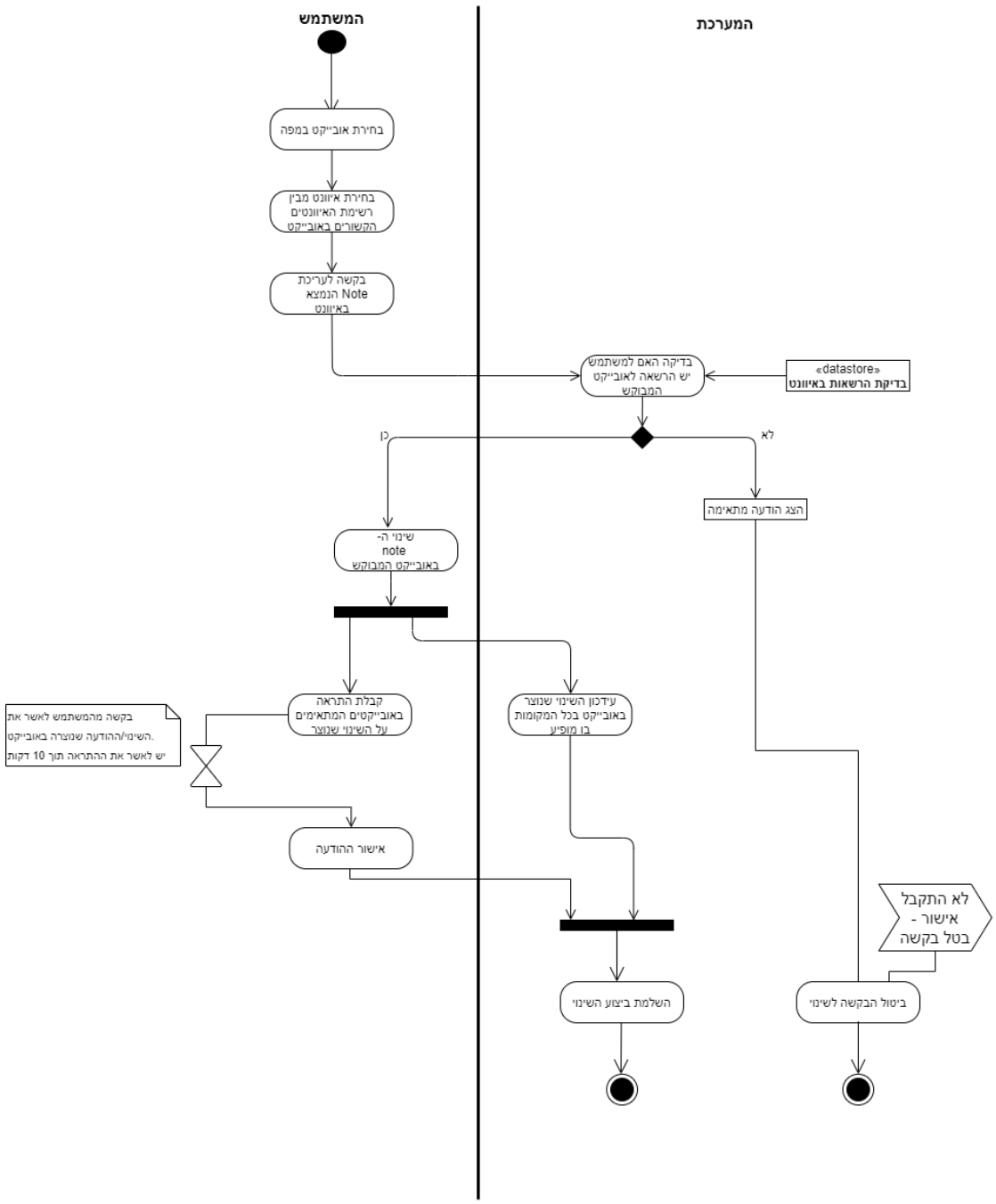
אלו הם ה- Use Case's בעלי משקל עדיפות גבוה ביותר מפני שהינם עוסקים באבני הבניין של יצירת המערכת בצורה השימושית ביותר, הם עוסקים בבחירת והוספת אירוע לאובייקטים והכך הוספת פונקציונליות למערכת וכל מה שלשמה המערכת נוצרה.

בנוסף חשוב להזכיר את UC1,UC11 שגם קיבלו PW גבוהה והם תורמים למערכת להיות גנרית ולנתינת משמעות לבנית האזור.

ה- Requirements עם העדיפות הנמוכה ביותר הן REQ2,REQ7,REQ8.

דרישות אלו קיבלו תיעדוף נמוך משל הדרישות האחרות במערכת וזה מפני שדרישות אלו נועדו "לשדרוג" המערכת ואינן מהוות בסיסי הכרחי להצלחה ושימוש בה. דרישות אלו יהוו דרישות שהינן "nice to have" במערכת כי הן ישפרו את השימוש ואת הנוחיות שבה.

Activity Diagram for UC9 and UC10



Formal Requirements Model

בנדבך זה נתאר יצירת אירוע ושיוכו לאובייקט.
Using UC1, UC3.

[event,object]

eventToObject

newEvent: event⇸object
Events: ℙ event
Objects: ℙ object

Dom newEvent ⊆ Events
Ran newEvent ⊆ Objects

newEvent ∈event ⇸ object
HospitalObjects::= bed| nurseDesk| reception| doctorDesk

HospitalObjects

hospitalObjects: ℙ object
hObjects: HospitalObjects
newEvent: event ⇸ object

Dom newEvent ⊆ Events
Ran newEvent ⊆ Objects

ObjectsForHospital ≙ eventToObject ∧ HospitalObjects

operation

Events': ℙ event
newEvent': event ⇸ object

Dom newEvent' ⊆ Events

addedEvent0

Δ eventToObject
e?: event
o!: object

---

e? ∈ Events\ dom newEvent
newEvent'= newEvent ∪ {e? ↦ o! }

noPremission

Ξ eventToObject
e?: event

---

e? $\in$ Events
event'= event $\cup$ {e?}
Events'= Events
newEvents'= {event $\cup$ {e?}} $\rightarrow\!\!\!\!\rightarrow$ objects

Access::= approved| denied

success

x!: Access

---

x!= approved

failure

x!: Access

---

x!= denied

addedEvent $\widehat{=}$ (addedEvent0 $\wedge$ success) $\vee$ (noPremission $\wedge$ failure)

newEventType0

Ξ eventToObject
e?: event

---

e? $\in$ Events
event'= event $\cup$ {e?}
Events'= Events
newEvent'= {event $\cup$ {e?} $\mapsto$ object}

notInEven

Ξ eventToObject
e?: event

---

e? $\notin$ Events

newEventType $\widehat{=}$ (newEventType0 $\wedge$ success) $\vee$ (notInEvents $\wedge$ failure)

priorityEvents

Ξ eventToObject
priorityEvent: $\mathbb{P}$ event
e?: event
o: object

priorityEvent= { e:Event, o:object| newEvent "000"= SOS • {e?}$\in$ dom newEvent

//למשתמש תוצג הערה במערכת לשימוש בקוד "000" כקוד לevent בעל עדיפות גבוהה, איזשהו מקרה חירום ולכן יומר במחרוזת הנקראת SOS ובכך שאר המשתמשים יוכלו להבחין ולתת עדיפות לאירועים מסוג זה.

EventsCounts

Ξ eventToObject
Ecount!: $\mathbb{N}$

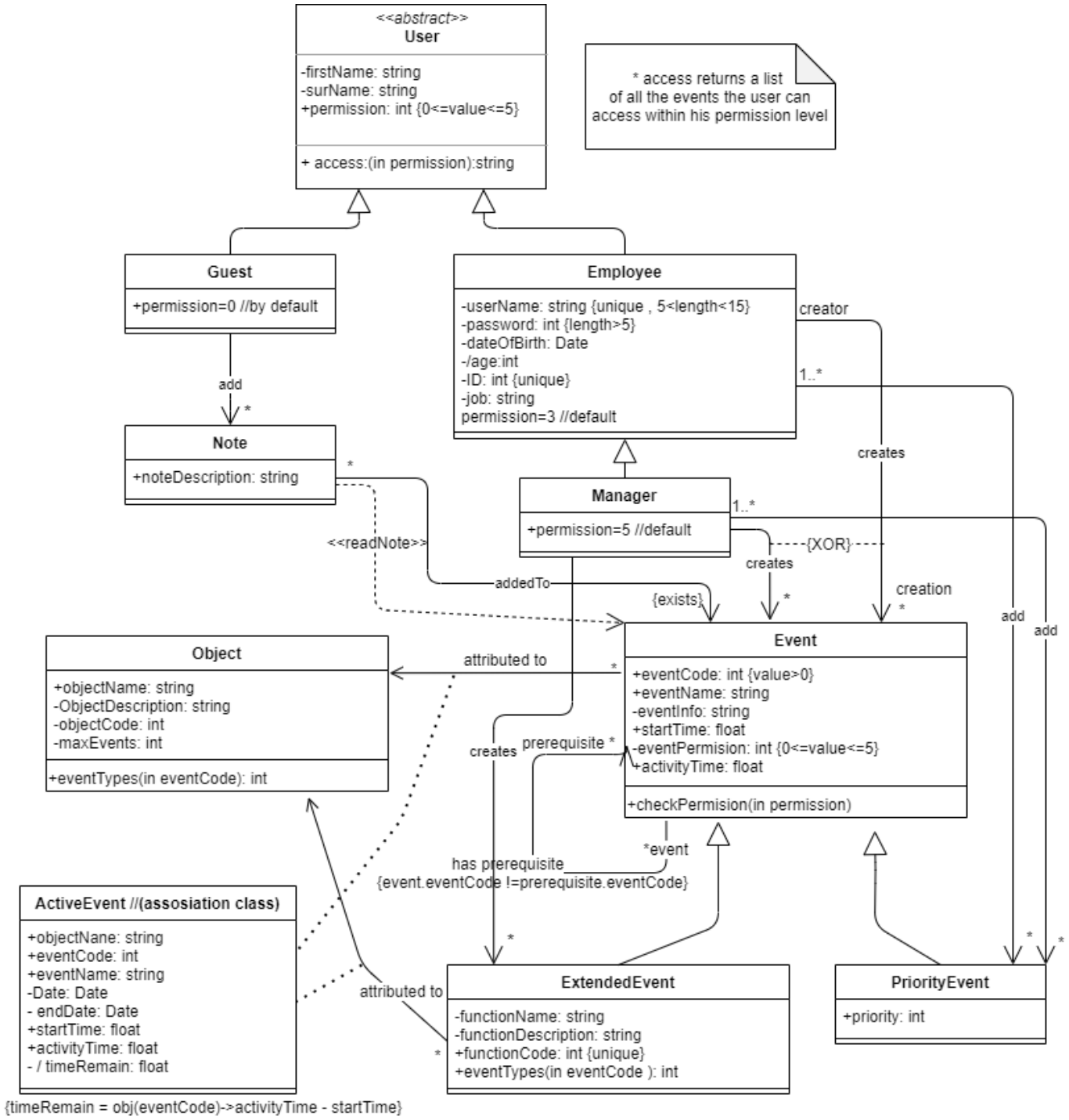
Ecount!= #(Events\ dom NewEvent)

eventsFromObject

Ξ eventToObject
oEvents!: event
e?: event
objectCheck?: object

oEvents!= {{e: event} $\rhd$ {objeckcheck?}}

Design


<<abstract>>
User
-firstName: string
-surName: string
+permission: int {0<=value<=5}
+ access:(in permission):string
* access returns a list of all the events the user can access within his permission level
Guest
+permission=0 //by default
Employee
-userName: string {unique , 5<length<15}
-password: int {length>5}
-dateOfBirth: Date
-/age:int
-ID: int {unique}
-job: string
permission=3 //default
creator
1..*
add
*
Note
+noteDescription: string
creates
Manager
+permission=5 //default
1..*
{XOR}
<<readNote>>
addedTo
creates
{exists}
creation
add
add
Object
+objectName: string
-ObjectDescription: string
-objectCode: int
-maxEvents: int
+eventTypes(in eventCode): int
attributed to
Event
+eventCode: int {value>0}
+eventName: string
-eventInfo: string
+startTime: float
-eventPermision: int {0<=value<=5}
+activityTime: float
+checkPermision(in permission)
creates
prerequisite *
*event
has prerequisite
{event.eventCode !=prerequisite.eventCode}
ActiveEvent //(assosiation class)
+objectNane: string
+eventCode: int
+eventName: string
-Date: Date
- endDate: Date
+startTime: float
+activityTime: float
- / timeRemain: float
attributed to
ExtendedEvent
-functionName: string
-functionDescription: string
+functionCode: int {unique}
+eventTypes(in eventCode): int
PriorityEvent
+priority: int
{timeRemain = obj(eventCode)->activityTime - startTime}