סדנה בתכנות מתקדם בשפת JAVA

מגיש: עוז הררי

ת.ז: 300173804

הפרוייקט מציג אתר לשירות למסירה ואימוץ של חיות בית. המערכת מאפשר להציע חיות בית לאימוץ וכן לחפש חיות בית לאימוץ.

**הוראות התקנה**

את המערכת אפשר להריץ בעזרת תוכנת NetBeans והורדת שרת Glassfish 4.1.

יש לפתוח פרויקט חדש מסוג Web application שמתשמש בJSF 2.2

קבצי הxhtml צריכים להיות תחת תיקית web

קובץ הcss צריך להיות תחת תיקיית resources שהיא תהיה תחת תיקיית web.

כל קבצי הjava צריכים להיות תחת תיקיית animalsBeans שהיא תהיה תחת תיקיית java שהיא תהיה תחת תיקיית src.

קונפיגורציה של השרת Glassfish

יש להכנס לאתר

<http://localhost:4848/>

Configuration > HTTP Service > Virtual Servers

יש להגדיר property בשם alternatedocroot\_1 והערך שלו יהיה from=/images/\* dir=/path/to

זה חיוני על מנת שאפשר יהיה להציג תמונות.

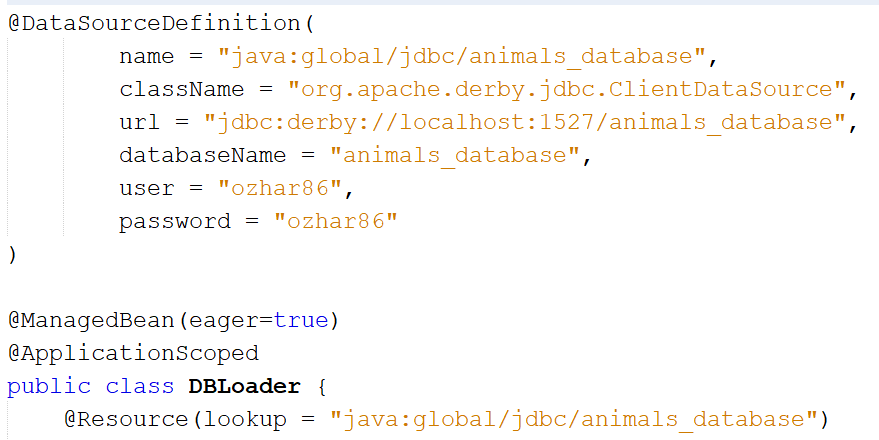
יש ליצור תיקיה path תחת התיקייה domain1 שהותקנה במהלך ההתקנה של netbeans. תחת התיקיה path יש ליצור תיקייה to ותחתיה תיקייה images. בתיקייה זו יש לשים את כל התמונות הרלונטיות. שם ישמרו בעתיד כל התמונות של האפליקציה.

יש להעתיק את הכתובת המלאה של התיקייה images ולהדביק אותה בערך המחרוזת של המשתנה PATH\_TO\_IMAGES אשר נמצא בקובץ UtilsFunc.java.

קונפיגורציה של מסד הנתונים

בתוכנה netbeans תחת לשונית services יש לבחור databases. יש ללחוץ לחיצה ימנית בעכבר על Java DB. ולבחור בcreate database

יש לתת שם למסד הנתונים, למשל animals\_database. יש לבחור שם משתמש וסיסמה. את הפרטים האלה יש לעדכן בקובץ DBLoader.



את הטבלאות של מסד הנתונים יש למלא כפי שיפורט בהמשך.

יש למלא את הטבלאות Category שבה יש את רשימת סוגי בעלי החיים.

יש למלא את הטבלה Area שבה יש אזורים בארץ (דרום, מרכז וכו')

יש למלא את טבלת City שבה יש את רשימת כל הערים בארץ. כל עיר צריכה להיות משויכת לאזור.

יש למלא את הטבלה MAX\_ANIMAL\_ID בערך אחד בלבד. שזה יהיה הערך המינימלי שממנו יתחילו להיות מספרי הזהות של בעלי החיים.

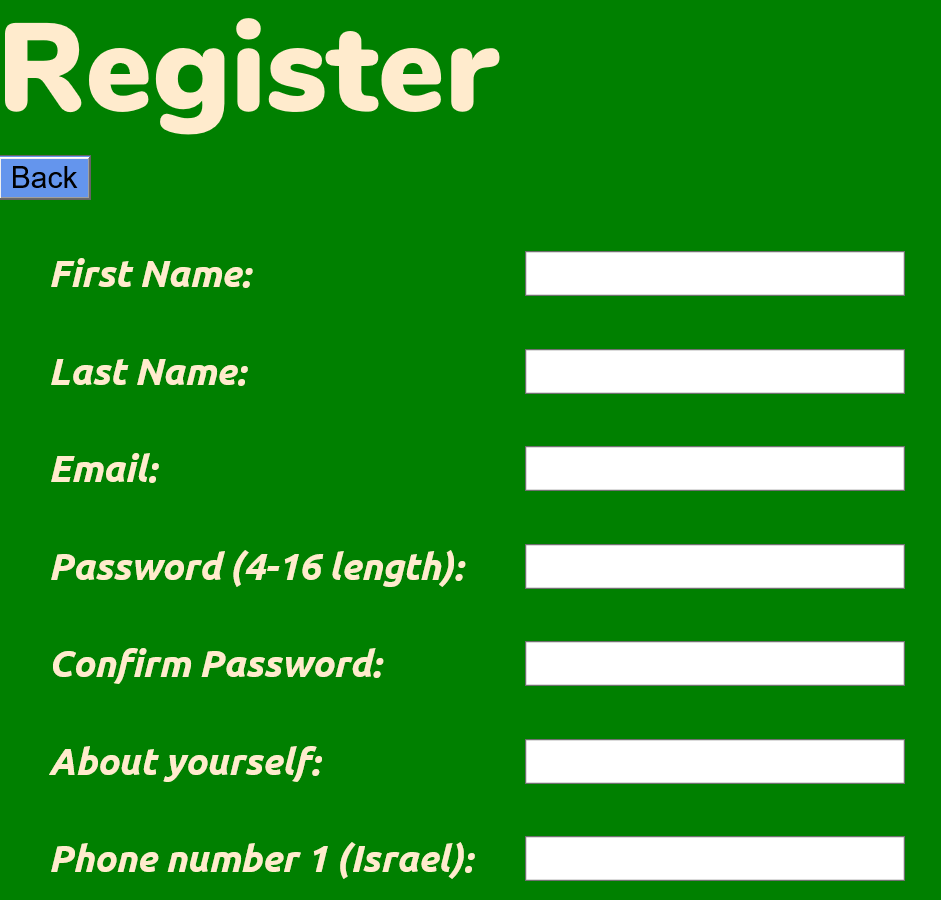
את הטבלאות האלה יש למלא לפני שמריצים את התוכנית.

**תאור המערכת מנקודת מבט של המשתמש**

תחילה המשתמש צריך להתחבר עם שם משתמש וסיסמה או להרשם.

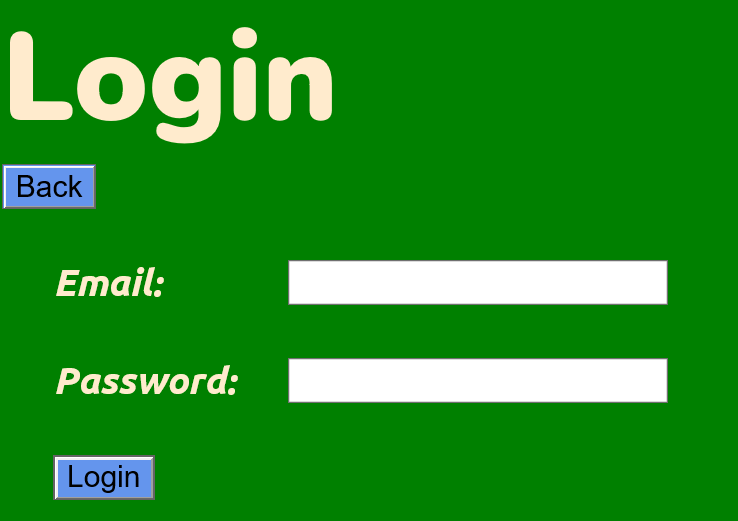
מסך הרשמה (register\_user):

המשתמש ממלא את פרטיו. בין היתר הוא בוחר סיסמה. הוא יכול לכתוב כמה מילים על עצמו. והוא ממלא עד שני מספרי טלפון. לאחר מכן הוא עובר למסך שבו הוא מעלה תמונה שלו.



מסך login:

המשתמש מכניס מייל (כשם משתמש) וסיסמה.

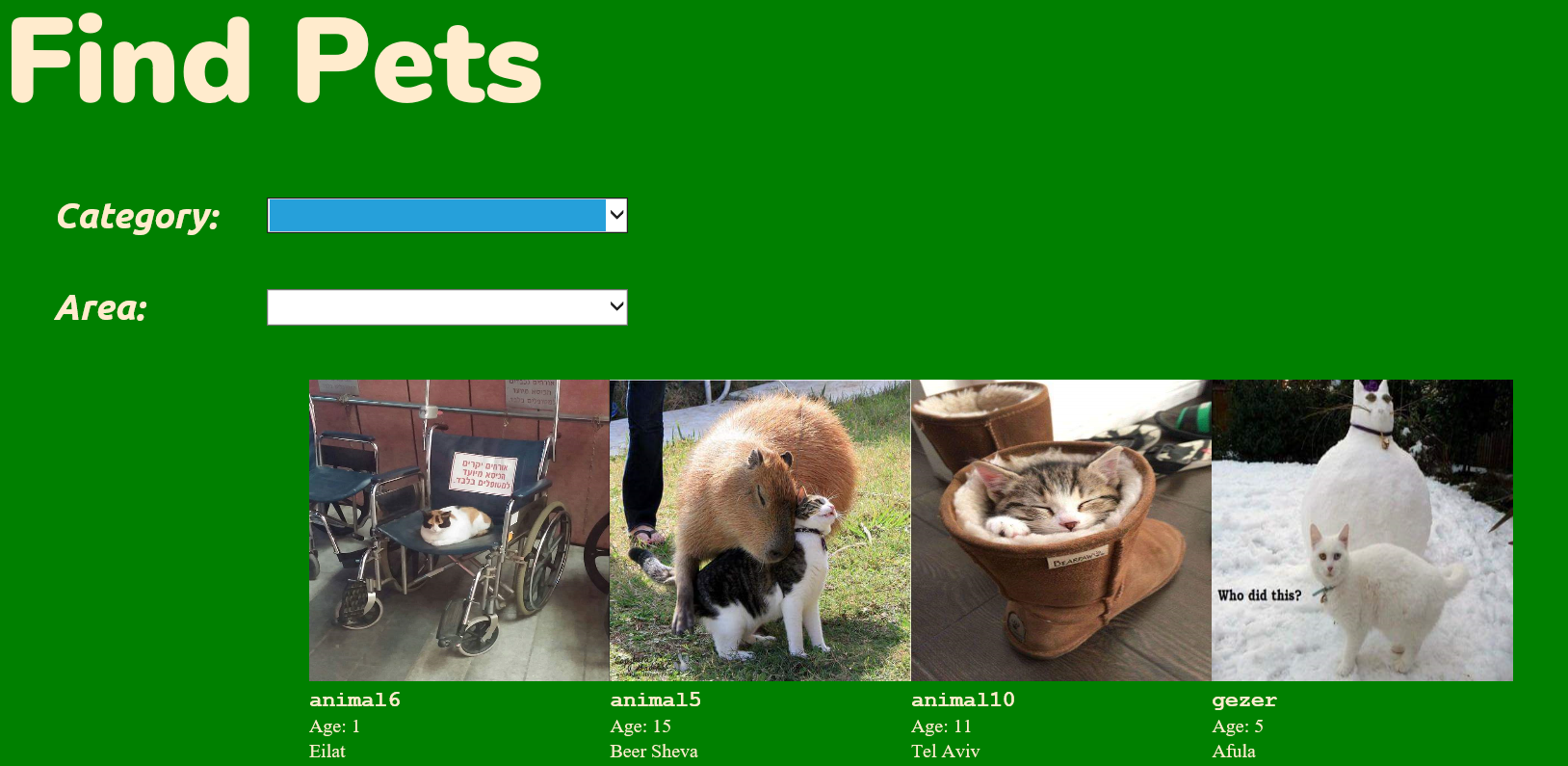


אם שהמשתמש נכנס למערכת, הוא מגיע למסך menu. הוא יכול לחפש בעלי חיים לאימוץ או להציע בעל חיים בעצמו. הוא יכול לחזור אחורה. והוא יכול גם לשנות פרטי חיב לאימוץ שהוא הציע בעבר.





במסך החיפוש המשתמש יכול לראות את כל החיות המוצעות לבחירה. הוא יכול לסנן את לפי קטגוריה (כלב, חתול וכו'). הוא יכול לסנן לפי אזור בארץ. החיות ממוינות לפי תאריך הפרסום שלהן. ליד כל תמונה יש את שם החיה, תאור קצר שלה, גיל, והיישוב בה היא נמצאת



לאחר שהמשתמש לוחץ על תמונה מסוימת הוא מגיע למסך פרטים. במסך הזה הוא יכול לראות את כל פרטי החיה. כולל כל התמונות ששייכות לה. כולל גם פרטים על מוסר החיה, כולל תמונת הפרופיל שלו והמידע שהוא סיפק על עצמו. כמו כן פרטי יצירת קשר אליו. בכותרת הוא יראה תאור קצר של החיה.

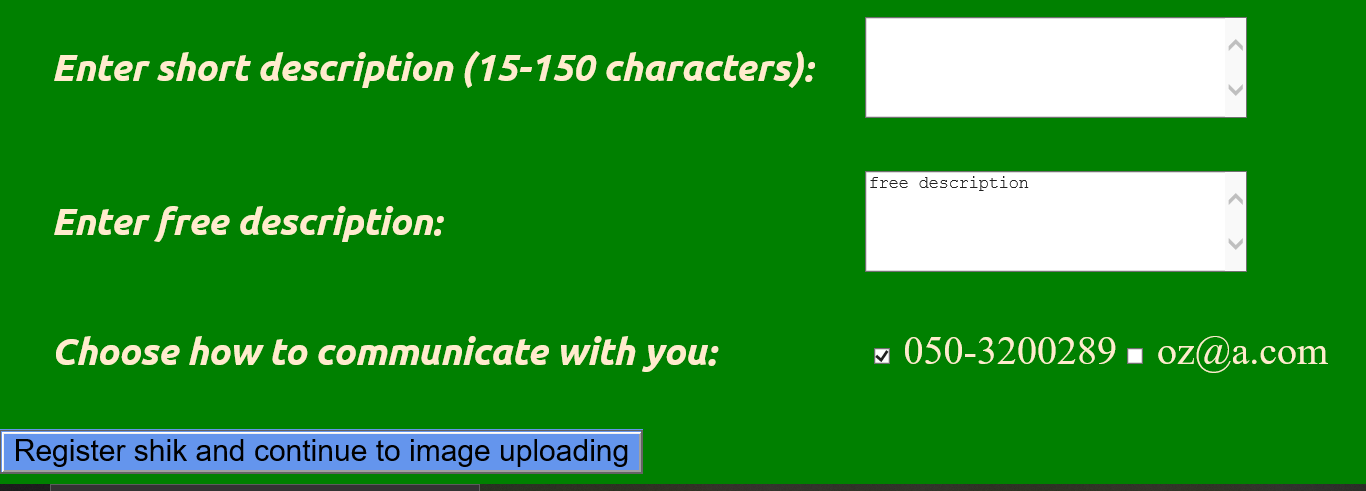




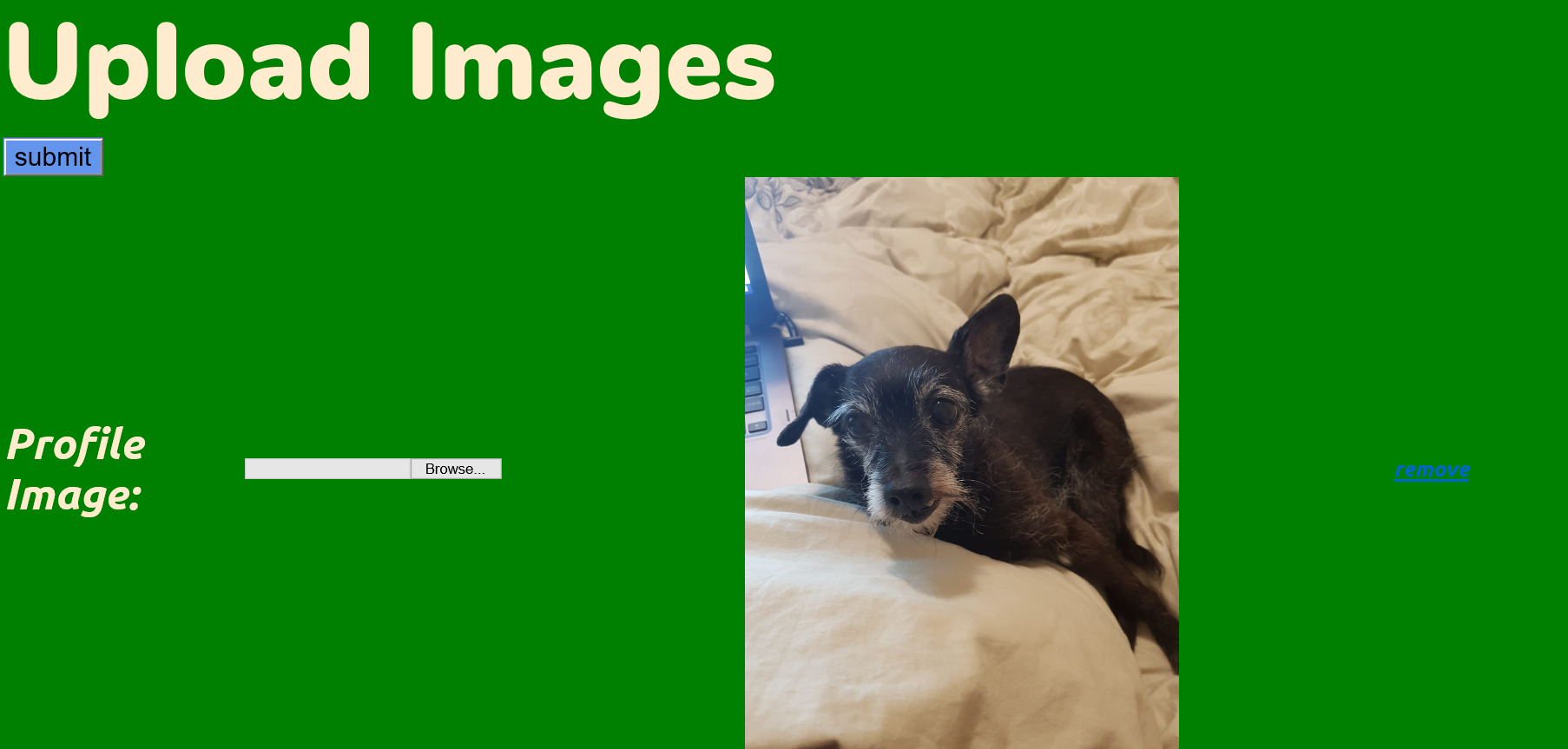


אפשרות אחרת היא להציע חיה לאימוץ. המשתמש ממלא את פרטי החיה. ובוחר איזה פרטי התקשרות הוא רוצה שיופיעו במודעה, מבין המייל שלו ושני מספרי הטלפון שהוא סיפק.

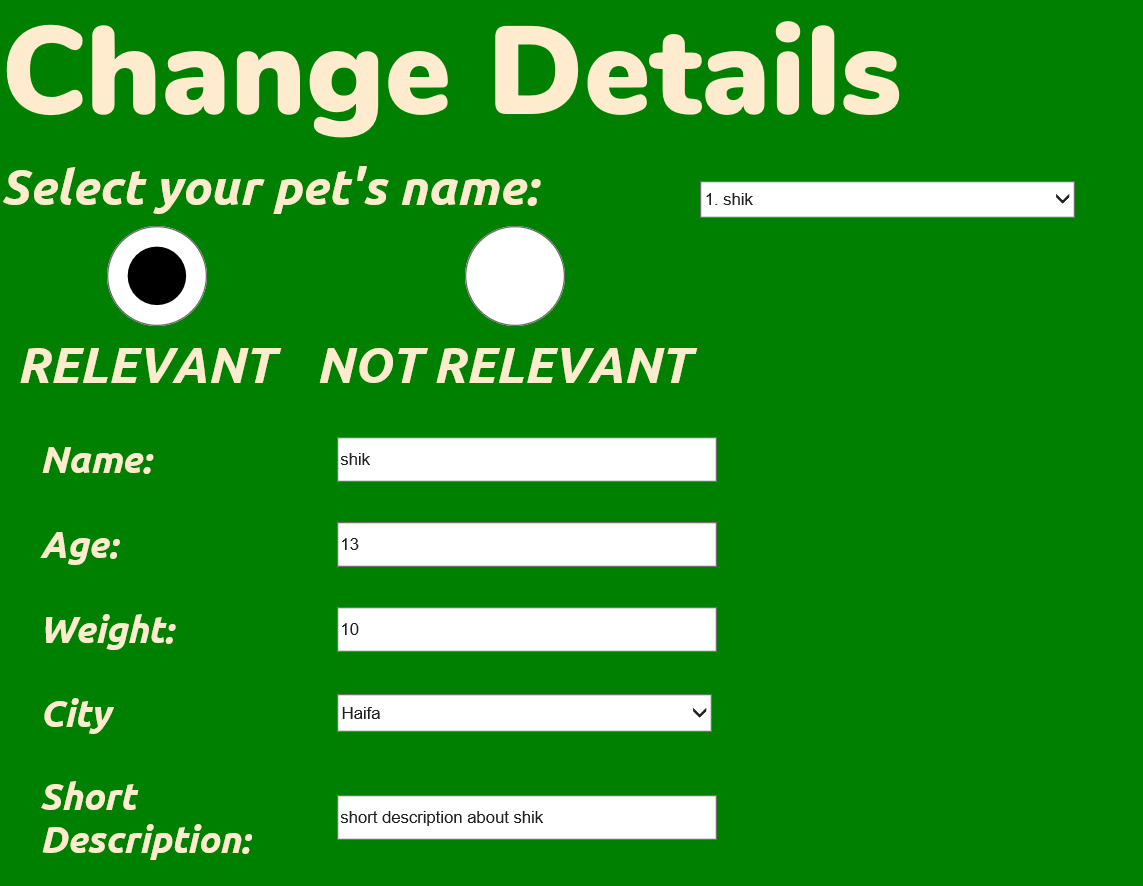




כשהמשתמש לוחץ submit החיה כבר רשומה והמשתמש רק צריך לבחור תמונות. במסך הזה המשתמש מעלה עד 5 תמונות. כשהתמונה הראשונה היא הראשית שמופיע בטבלת החיפוש (אם אין כזאת תופיע תמונה אפורה). המשתמש יכול להסיר את התמונה כדי להעלות חדשה.



אפשרות אחרונה של המשתמש היא לשנות פרטים במודעה שהוא סיפק. בין היתר הוא יכול לקבוע שהמודעה כבר לא רלונטית ולכן היא כבר לא תופיע במסך החיפוש אצל אנשים.



המשתמש יכול להתקדם ולעבור את מסך בחירת התמונות כך שהוא יכול לשנות את התמונות שהוא העלה.

**פירוט מסד הנתונים**

USERS

* EMAIL, varchar(100) primary-key
* FIRST\_NAME varchar(50)
* LAST\_NAME varchar(50)
* PHONE1 varchar(11)
* PHONE2 varchar(11)
* IMAGE\_NAME varchar(104)
* PASSWORD varchar(16)
* ABOUT\_ME varchar(250)

הערה: NAME\_IMAGE משמעותו השם של תמונת הפרופיל כולל סיומת. למשל [oz@gmail.com.jpg](mailto:oz@gmail.com.jpg) כאשר שם הקובץ הוא האימייל של המשתמש.

AREA

* NAME varchar(50) primary key

זה שם של אזור, למשל North או South

CITY

* CITY\_NAME varchar(100) primary key
* AREA\_NAME varchar(50) foreign key from AREA

CATEGORY

* NAME varchar(100) primary key

MAX\_ANIMAL\_ID

* NUMBER bigint primary key

זה מספר חיובי שרירותי כלשהו שיסמן את ההתחלה של כל "תעודת זהות" של חיה

ANIMAL

* ANIMAL\_ID varchar(10) primary key
* ANIMAL\_NAME varchar(100)
* CATEGORY varchar(100) foreign key from CATEGORY
* CITY varchar(100) foreign key from CITY
* AGE int
* WEIGHT int

OFFER

* EMAIL varchar(100) foreign key from USERS part of primary key
* ANIMAL\_ID varchar(10) foreign key from ANIMAL part of primary key
* DATE\_PUBLICATON date
* SHORT\_DESCRIPTION varchar(250)
* LONG\_DESCRIPTION long varchar
* IS\_RELEVANT boolean
* CONTACT1 varchar(100)
* CONTACT2 varchar(100)
* CONTACT3 varchar(100)

שלושת הCONTACT האחרונים מייצגים את הפרטים שמוסר החיה הסכים לפרסם כשמדובר בחיה הזאת

IMAGE

* IMAGE\_NAME varchar(50)
* ANIMAL\_ID varchar(10) foreign key from ANIMAL and part of primary key
* SERIAL int ,part of primary key

מתאר תמונה. מספר סידורי אפס מסמן תמונת פרופיל של החיה. שם התמונה מורכב ממספר הזהות של החיב ואז מקף תחתון ואז מספר הסידורי של התמונה וכולל הסיומת של הקובץ. למשל 1001\_2.jpg

**תאור המחלקות השונות**

**DBLoader.java**

מחלקה זו מייצגת אתחול של מסד הנתונים. היא מכילה פונקציה אחת בשם getStatement שמקבלת מחרוזת של פקודת sql ומחזירה אובייקט מסוג PreparedStatement.

**Validators.java**

מחלקה זו מכילה פונקציות עזר שקשורות לvalidation של נתונים שהמשתמש מכניס. למשל פונקציה validateEmail שבודקת אם המייל תקין.

פונקציה isEmailExist שבודקת אם המייל קיים במערכת

פונקציה validateConfirmPassword בודקת אם הסיסמה שהמשתמש הכניס בפעם השנייה תואמת לסיסמה שהוא הכניס בפעם הראשונה (כשעושים confirm email)

פונקציה validatePhone בודקת אם מספר הטלפון הוא תקין

פונקציה validateImage בודקת אם התמונה היא בסיומת נכונה כמו jpg

פונקציה validateLogin בודקת אם הסיסמה שהמשתמש הכניס בlogin תואמת למייל שהוא הכניס.

פונקציה findArea מחפשת את האזור ששייך לעיר מסוימת

**מחלקה NavigationParameters**

המחלקה משמשת להעברת פרמטרים בין עמוד JSF אחד לשני. בפועל לא השתמשתי בה הרבה אלא רק עבור העברת ID של חיה כשהמשתמש בוחר אותה במסך החיפוש

**מחלקה Area**

מחלקה זו מכילה פונקציה getAreas שמעבירה רשימה של כל האזורים הקיימים במערכת. הרשימה תמיד מכילה מחרוזת ריקה כדי שהמשתמש יוכל לבחור גם במחרוזת ריקה (ובעצם לציין שהוא לא רוצה לציין אזור)

**מחלקה City**

מחלקה שמטפלת בכל מה שקשור לערים. מכילה משתנה selectedArea שהוא "אזור נבחר" ועל פיו היא יכולה להוציא רשימה של כל הערים שנמצאות באותו אזור. כמו כן מכילה גם פונקציה allCities שמוציאה רשימה של כל הערים מבין כל האזורים.

**מחלקה UtilsFunc**

מחלקה זו מכילה קבועים עבור כל האפליקציה. וכן גם פונקציות סטטיות לחישובים שונים.

למשל PATH\_TO\_IMAGES שמציין איפה בכונן של המחשב לשמור את כל התמונות

RELATIVE\_IMAGE\_PATH מציין את התיקייה הספציפית של התמונות

NO\_PICTUR\_NAME מציין את שם התמונה האפורה שאומרת שלחיה מסוימת אין תמונה ראשית

פונקציה getExtension שמקבלת אובייקט Part ומחזירה את סוג הקובץ

פונקציה saveImage שומרת תמונה במחשב תחת שם מסוים

פונקציה removeImage מוחקת תמונה מהמחשב

**מחלקת User**

מחלקה זו מייצגת משתמש. היא מתקשרת עם העמוד register\_user. מכילה את כל הproperties של אותו יוזר. מחלקה זו היא בסקופ של session כך שפרטי היוזר נשמרים לאורך כל הגלישה.

פונקציה addEntry מכניסה את כל המידע למסד הנתונים

פונקציה fillUser ממלאת את שאר פרטי המשתמש על פי המייל שלו בלבד. נועד למלא את כל פרטי המשתמש כאשר המשתמש נכנס בlogin ומכניס רק את המייל שלו.

פונקציה communications מחזירה רשימה של כל פרטי ההתקשרות של המשתמש, כאלה שהם לא null או מחרוזות ריקות.

**מחלקה Animal**

המחלקה הגדולה ביותר באפליקציה. מחזיקה מידע מלא על חיה מסוימת. כולל פרטי ההצעה לאימוץ שלה. מתקשרת עם העמוד register\_animal וגם עם מחלקות java אחרות.

לשים לב שגיל ומשקל שערכם הוא 1- משמע שהמשתמש לא ציין אותם. ולכן בget המחלקה תחזיר null.

המערך communications זה פרטי ההתקשרות למי שמוסר את החיה. אלה פרטי ההתקשרות שהוא הסכים לחשוף.

המערת imagesNames מציין את שמות הקבצים. אם במקום הסידורי המסוים המשתמש לא העלה תמונה אז הערך יהיה NULL.

מכיון שיש עד 5 תמונות אז יש 5 פונקציות של get ו5 פונקציות של set. בתוך כל אחת מהפונקציות האלה היא פונה לפונקציה private לפי אינדקס מתאים.

הפונקציה register רושמת את החיה למאגר.

הפונקציה clearImagesOfID מוחקת את כל התמונות ששייכות לחיה מסוימת ממסד הנתונים. זאת על מנת שלפני שהמשתמש ישנה את התמונות, שכל מה שהיה לפני ימחק.

הפונקציה submitImages מכניסה את התמונות למסד הנתונים.

המחלקה מזריקה את המחלקה AnimalIDGenerator על מנת ליצור תעודת זהות לכל חיה שנכנסת. מחלקה זו בודקת במסד הנתונים מה היה תעודת הזהות האחרונה שהוכנסה ואז מגדילה אותה ב1 ומספקת אותה לחיה החדשה שנכנסה.

המחלקה AnimalItemSearch מייצגת מידע על חיה שיופיע בטבלת חיפוש של החיות, כלומר תמונה ראשית, גיל, קטגוריה, תאור קצר וכו'. זאת על מנת ליצור רשימה של אובייקטים מסוג AnimalItemSearch שמייצגים מידע חלקי.

**המחלקה ImagesUploadHandle**

מחלקה זו עובדת מול הדף images\_upload על מנת להעלות תמונות של חיה מסוימת.

כל פעם שעושים remove או מעלים תמונה, התמונה נשמרת או נמחקת במסד הנתונים

על מנת שהרכיב graphicImage לא יפנה לcache ויציג תמונה שהיא כבר לא עדכנית, המתודה getImageName תמיד שומרת עותק של התמונה הנכונה עם מספר סטטי שמשתנה. העותקים האלה שמותיהם מאוחסנים במערך imagesNamesPreventCache. כשהמחלקה נהרסת, כל העותקים האלה נמחקים על ידי פונקציה finalize.

**המחלקה ImagesList**

מייצגת רשימה של חיות שעונות לקריטריונים של חיפוש. מחזיקה רשימה של אובייקטים מסוג AnimalItemSearch. היא יכולה לעשות פילטר לפי קטגוריה ולפי אזור. היא מחזיקה דגל שנקרא arranged שמציין האם כבר יש רשימה של החיות או לא.

הרשימה צריכה להיות רשימה של כל החיות הרלוונטיות. וממיונות לפי סדר פרסומן.

הפונקציה getAnimals מחזירה את הרשימה הזאת. אם יש פילטרים של קטגוריה ושל אזור אז היא עושה את הפילטרים האלה.

הפונקציה loadAnimals טוענת את החיות ממסד הנתונים. רק החיות הרלוונטיות

פונקציה filterCategory עושה פילטר לפי קטגוריה. מה שלא עונה על הקטגוריה היא מוציאה אותו מהרשימה.

פונקציה chooseAnimal נקראת כשהמשתמש בוחר לראות פרטים של חיה מסוימת. היא מעבירה פרמטר לאובייקט NavigationParameters. עמוד הבא יקרא את הפרמטר מהאובייקט הזה.

**מחלקה OfferShow**

נועדה להציג את כל הפרטים של החיה ושל המוסר של החיה כשמגיעים לעמוד של פרטים על חיה. מגיעים לעמוד הזה מטבלת החיפוש של החיות.

המחלקה מחזיקה אובייקט מסוג Animal שמייצד מידע מלא על חיה, וכמו כן פרטים על המוסר של החיה כמו about me, השם שלו וכו'. המשתנה loaded מציין אם הפרטים כבר עלו ממסד הנתונים או לא.

הפונקציה loadAnimalFromDB מעלה את כל הפרטים האלה על פי המשתנה שעבר דרך אובייקט NavigationParameters. הפרמטר הוא תעודת הזהות של החיה. דרכה אפשר להגיע למי שמוסר את החיה, לתמונות שלה, לאזור שלה וכו'.

**המחלקה AnimalForUpdate**

נועדה על מנת לשנות פרטים של חיה. היא מחזיקה משתנה מסוג Animal. היא מתקשרת עם עמוד שבו המשתמש משנה את הפרטים.

המשתנה relevantOptions הוא רשימה עם שני פריטים: "רלוונטי" או "לא רלוונטי" שזה מה שיוצג למשתמש בקשר לחיה. אם הוא יבחר "לא רלוונטי" אז היא תפסיק להיות מוצגת.

בעמוד שמוצג למשתמש יש לו אפשרות לראות את כל החיות שהוא מסר לאימוץ ולבחור אחת מהן. ברגע שהוא בוחר אז כל הפרטים שהוא הכניס עולים. והוא יכול לשנות אותם.

הפונקציה setUserChoise מקבלת את הבחירה של המשתמש, איזה חיה הוא בחר לשנות (לפי תעודת זהות שלה). והוא מעלה את הפרטים שלה בהתאם.

המשתנה animalsID הוא מפה שמקשרת בין הLabels שיוצגו למשתמש שאלה השמות של החיות שלו, לבין תעודות הזהות שלהן שהוא כמובן אינו יודע. ברגע שהוא בוחר שם אז המפה תקשר לנו את תעודת הזהות.

פונקציה listAnimals מחזירה רשימה של כל החיות שהוא מסר לאימוץ

פונקציה update מעדכנת את הפרטים בלי להוביל את המשתמש לעמוד שבו משנים גם את התמונות.

פונקציה updateAndChangeImages עושה אותו דבר רק שהיא מובילה את המשתמש גם לעדכון תמונות. אם המשתמש בוחר לעדכן תמונות המערך names שמייצג את שמות התמונות של החיה מתעדכן על מנת שהתמונות יעלו.

**באגים ידועים**

* באג שעושה rotate אוטומטית לתמונות שהאורך שלהן יותר גדול מהרוחב. האפליקציה הופכת אותן ב90 מעלות.
* כאשר המשתמש מעלה תמונה ואז מעלה תמונה נוספת במקומה מבלי לצאת מהעמוד אז התמונה הקודמת מופיעה שוב. זה מכיוון שהרכיב graphicImage פונה לcache. את המניפולציה עם הcache לא יישמתי על עמוד זה.

**מקום לשיפור שלא הספקתי**

* לא עשיתי אפשרות למשתמש לשנות את הפרטים שלו עצמו כמו להחליף תמונה
* אפשר היה לתת למשתמש אפשרות למיין גם לפי גיל
* בתכנון הפרוייקט חשבתי ליצור מועדפים עבור כל משתמש. אם נתקלים במודעה מסוימת של חיה אפשר יהיה לסמן אותה ב"מועדפים" ואז לראות את רשימת המועדפים. זה משהו שלא הספקתי לעשות.