CarMechanic

Έχει υλοποιηθεί η περισσότερη λειτουργικότητα από αυτήν που γράφτηκε στο εγχειρίδιο χρήσης. Πιο συγκεκριμένα, έχουν υλοποιηθεί:

- Σε κάθε σελίδα:
  - Υπάρχει η Navigation Bar, με 6 επιλογές για μη συνδεδεμένους χρήστες:
    - Αρχική, Πληροφορίες, Επικοινωνία, Σύνδεση Χρήστη, Εγγραφή Χρήστη (Οδηγούν σε αντίστοιχες σελίδες)
  - Για συνδεδεμένους χρήστες:
    - Όπως για μη συνδεδεμένους χρήστες με τη διαφορά ότι δεν υπάρχουν οι επιλογές σύνδεσης και εγγραφής και στη θέση τους υπάρχει μήνυμα με το όνομα του χρήστη το οποίο είναι ένα dropdown button, με δύο επιλογές: Προφίλ και Αποσύνδεση

- Αρχική Σελίδα (index.jsp – servlets/searchservlet.java):
  - Αναζήτηση αυτοκινήτου με κείμενο ή με επιλογές. Από το κείμενο χρησιμοποιούνται μόνο οι 2 πρώτες λέξεις και αγνοούνται όλες οι υπόλοιπες. Στις επιλογές, πρέπει να επιλεχτεί τουλάχιστον ο «Κατασκευαστής».
  - Τα δεδομένα των επιλογών προέρχονται από τη βάση δεδομένων. Κατά τη φόρτωση του index.jsp, γίνεται αναζήτηση στη βάση δεδομένων με χρήση της JSTL για SQL queries. Συγκεκριμένα αναζητεί για τους κατασκευαστές, και «γεμίζει» τις επιλογές για τους κατασκευαστές. Αφού επιλεγεί ένας κατασκευαστής, καλείτε μια JavaScript συνάρτηση, η οποία με τη σειρά της καλεί το servlets/UpdateSearchPageSerlvet.java δυναμικά, προκειμένου να «γεμίσουν» οι επιλογές του μοντέλου, σύμφωνα με τον κατασκευαστή. Αντίστοιχη ενέργεια γίνεται και για τις άλλες επιλογές.
  - Καλείτε μέσα από το searchservlet μια κατάλληλη μέθοδος από τη database/DatabaseManager.java, ανάλογα με την μέθοδο αναζήτησης που επιλέχτηκε, και εκτελείται αναζήτηση στη βάση δεδομένων. Τα αποτελέσματα επιστρέφονται πίσω στο index.jsp, και εμφανίζονται στο κάτω μέρος «Αποτελέσματα Αναζήτησης». Αν δεν βρεθούν αποτελέσματα, εμφανίζεται κενός πίνακας.
  - Δίπλα από κάθε αυτοκίνητο στη λίστα αποτελεσμάτων υπάρχει ένας υπερ-σύνδεσμος «Problems» που οδηγεί στη σελίδα «Προβολή Αυτοκινήτου» για το συγκεκριμένο αυτοκίνητο. Συγκεκριμένα καλείτε το carview servlet με όρισμα το model id , του αυτοκινήτου.

- Σελίδα Προβολής αυτοκινήτου (carview.jsp – servlets/CarView.java):

- Η σελίδα καλείτε μέσω του αντίστοιχου servlet με όρισμα “s” το αναγνωριστικού του αυτοκινήτου. Το servlet κάνει αναζήτηση στη βάση δεδομένων για τις απαραίτητες πληροφορίες (Πληροφορίες μοντέλου, κατηγορίες προβλημάτων και τα προβλήματα για το αντίστοιχο μοντέλο) και επιστρέφει το αποτέλεσμα στο carview.jsp.
- Οι πληροφορίες για τα προβλήματα και το μοντέλο μεταφέρονται με χρήση Java Beans (model/Car.java, model/Problem.java, model/Category.java) και διαβάζονται με χρήση JSTL Core (for each tag) και Expression Language (${beanname.attribute}).
- Εμφανίζει τις πληροφορίες για το μοντέλο, μια εικόνα placeholder για μελλοντική χρήση, ένα κουμπί που επιστρέφει πίσω στην αρχική και ένα κουμπί «Προσθήκη Προβλήματος», το οποίο εμφανίζεται μόνο σε συνδεδεμένους χρήστες.
- Από κάτω εμφανίζονται τα καταχωρημένα προβλήματα για το συγκεκριμένο μοντέλο αυτοκινήτου. Είναι χωρισμένα σε καρτέλες ανά κατηγορία.
- Δίπλα από κάθε πρόβλημα, υπάρχει ένας υπερσύνδεσμος ο οποίος οδηγεί στη σελίδα «Προβολή προβλήματος». Συγκεκριμένα, καλείτε με μέθοδο GET μέσω του servlets/ProblewView.java και με όρισμα “s” το ID του προβλήματος που επιλέχτηκε.

- <u>Σελίδα προβολής προβλήματος (problemview.jsp - servlets/ProblemView.java):</u>
  - Καλείτε μέσω του servlet ProblemView.java, το οποίο κάνει αναζήτηση στη βάση δεδομένων για τις πληροφορίες του συγκεκριμένου προβλήματος, σύμφωνα με την παράμετρο (problem id) με την οποία καλέστηκε.
  - Εμφανίζει ένα κείμενο ως την περιγραφή του προβλήματος, διάφορες φωτογραφίες (σε αυτήν την έκδοση εμφανίζει μόνο placeholders), κουμπιά για προσθήκη φωτογραφιών και λύσης (μόνο για συνδεδεμένους χρήστες), τη λύση του προβλήματος αν υπάρχει και τέλος σχόλια των χρηστών (εμφανίζονται μόνο για συνδεδεμένους χρήστες).
  - Η προσθήκη λύσης δεν έχει υλοποιηθεί σε αυτήν την έκδοση.
- <u>Σελίδα εισαγωγής προβλήματος για ένα μοντέλο (insertproblem.jsp – serverlets/inserproblem.java):</u>
  - Έχει δύο πεδία εισαγωγής κειμένου, για περιγραφή και λύση προβλήματος και κουμπί για εισαγωγή (μιας μόνο σε αυτήν την έκδοση) φωτογραφίας.
  - Ο χρήστης επιλογή από μια λίστα την κατηγορία του προβλήματος.
  - Αφού πατηθεί το κουμπί «Αποθήκευση», αποστέλλονται οι πληροφορίες στο insertproblem.java, το οποίο αποθηκεύει το πρόβλημα στη βάση δεδομένων, και επιστρέφει το χρήστη στη σελίδα προβολής αυτοκινήτου.
- <u>Σελίδα προβολής προφίλ χρήστη (userprofile.jsp – servlets/UserProfile.java):</u>
  - Καλείτε μέσω του servlet, με όρισμα το “id” του χρήστη. Το servlet αναζητεί τις πληροφορίες του χρήστη στη βάση, και επιστρέφει το αποτέλεσμα στο userprofile.jsp.

    - Εμφανίζει τις πληροφορίες του χρήστη και μια εικόνα placeholder (για μελλοντική χρήση). Δίπλα από κάθε πεδίο υπάρχει ένα κουμπί επεξεργασίας. Όταν πατηθεί το κουμπί επεξεργασίας, από στατικό κείμενο, μετατρέπεται σε πεδίο εισαγωγής κειμένου, για να εισαχθεί η αλλαγή.
    - Το κουμπί αποθήκευση, στέλνει τις αλλαγές στο servlet, για να αποθηκευτούν στη βάση.
    - Τα κουμπιά κάτω από την εικόνα (Επεξεργασία & διαγραφή) είναι ανενεργά.
    - Στο κάτω μέρος, φαίνονται τα προβλήματα που έχει αποστείλει ο συγκεκριμένος χρήστη, με τον αντίστοιχο σύνδεσμο προς αυτά.
- Σελίδες λάθους και 404,401 (errorPage.jsp, error-404.jsp,error-401jsp):
    - Αν τυχόν εισάγουμε στη URL Address Bar κάποια διεύθυνση που δεν υπάρχει, θα οδηγηθούμε σε κατάλληλη σελίδα λάθους.
    - Αν συμβεί κάποιο λάθος στις JSP σελίδες, οδηγούμαστε σε κατάλληλη σελίδα.
    - Για τις περιπτώσεις άρνησης πρόσβασης (401 pages) έχει υλοποιηθεί η αντίστοιχη σελίδα, αλλά δεν χρησιμοποιείται σε αυτήν την έκδοση.
- Σελίδα επικοινωνίας (contact.jsp – servlets/SendMessage.java):
    - Με χρήση έτοιμων βιβλιοθηκών έχει υλοποιηθεί αποστολή μηνυμάτων από τους χρήστες προς το διαχειριστή της εφαρμογής, μέσω email. Συγκεκριμένα, μέσω μια φόρμας που συμπληρώνει ο χρήστης, στέλνετε σε μια προκαθορισμένη διεύθυνση ηλεκτρονικού ταχυδρομείου, το μήνυμα του χρήστη.

Αυτά που δεν έχουν υλοποιηθεί, μέχρι και τη στιγμή της συγγραφής αυτής της αναφοράς:

- ~~Επιλογή κατηγορίας κατά την εισαγωγή ενός προβλήματος. Τώρα γίνεται χειροκίνητα στη βάση δεδομένων.~~
- Δε γίνεται αποθήκευση των αλλαγών στο προφίλ χρήστη (βγάζει μήνυμα λάθους).
- Βελτίωση των γραφικών και της αισθητικής.
- Χρήση φίλτρων για σκοπούς καταγραφής και ασφάλειας σχετικά με τη σύνδεση/εγγραφή χρηστών.
- Η υποστήριξη φωτογραφιών στα προβλήματα και στο προφίλ χρήστη δεν είναι ολοκληρωμένη. Υπάρχει κώδικας, αλλά δεν έχει ολοκληρωθεί
- ~~Η χρήση παραμέτρων αρχικοποίησης του Container. Κυρίως για τη σύνδεση με τη βάση δεδομένων, η οποία καλείται σε κάθε servlet. Και για τη ρύθμιση της διαδρομής φακέλου στον οποίο θα αποθηκεύονται οι φωτογραφίες.~~
- Η χρήση των κατηγοριών των προβλημάτων στην προβολή τους στο προφίλ χρήστη.
- Η σύνδεση με μέσα κοινωνικής δικτύωσης.
- Η δημιουργία λογαριασμού διαχειριστή, για χειρισμό του περιεχομένου μέσω της εφαρμογής (διαγραφή προβλημάτων, σχολίων κτλ) και η αντίστοιχη λειτουργικότητα.
- ~~Χρήση listeners.~~
- ~~Η αποστολή λύσης σε ένα μη λυμένο πρόβλημα.~~

Για την υλοποίηση της εφαρμογής χρησιμοποιήθηκαν διάφορες πηγές στο διαδίκτυο και το βιβλίο του μαθήματος. Πιο συγκεκριμένα, η πλειοψηφία των πληροφοριών και των παραδειγμάτων κώδικα προήρθαν από:

- http://stackoverflow.com
- http://w3schools.com
- http://www.tutorialspoint.com/jsp
- http://theopentutorials.com/tutorials/java/design-patterns/post-redirect-get-prg-pattern-in-servlet-jsp/ (τρόπος για αποφυγή σφαλμάτων πολλαπλών αποστολών – δεν εφαρμόστηκε ακόμα [βλ. servlets/searchresultsservlet.java])
- http://stackoverflow.com/questions/19448316/responsive-images-inline-block-in-a-div-with-bootstrap (τρόπος για εμφάνιση των φωτογραφιών στα προβλήματα)
- http://www.oracle.com/technetwork/java/javase/jaf-136260.html (για την αποστολή των emails)
- https://java.net/projects/javamail/pages/Home (για την αποστολή των emails)
- http://getbootstrap.com
- http://stackoverflow.com/questions/10598409/initializing-a-database-connection-object-once-and-reusing-it-jboss

Κάποια από τα βασικά εργαλεία που χρησιμοποιήθηκαν:

- JSTL - MySQL and Core libraries
- Expression Language
- Java Beans, used with EL
- JQuery (JavaScript Library)
- Bootstrap Framework (για το αισθητικό κομμάτι)
- JAF & JavaMail (για υποστήριξη αποστολής email)
- UTH LDAP (UserManagement από 5η Εργασία)
- MD5 Hash Function (για αποθήκευση των τοπικών κωδικών)
- Context Listener για τη σύνδεση με τη βάση δεδομένω

Σημειώσεις υλοποίησης:

- Όλες οι λειτουργίες για τη βάση δεδομένων, βρίσκονται στο database/DatabaseManager.java. Η βάση είναι MySQL. Η σύνδεση ανοίγει με τη δημιουργία του Context, με τη χρήση ServletContextListener. Και κλείνει με καταστροφή του Context. Τα στοιχεία για τη σύνδεση στην DB εισάγονται στο web.xml.
- Σε κάθε JSP, γίνεται static include το κομμάτι του header (navigation bar etc) και τα JavaScript που χρειάζονται, στο τέλος του body. Βρίσκονται στο web/static/.
- Υπάρχουν 3 JSP στο web/static/ που χρησιμοποιούνται από το index.jsp, μέσω της JavaScript συνάρτησης, για την παραγωγή του περιεχομένου για τις επιλογές αναζήτησης, από το UpdateSearchPageServlet.