Πανεπιστήμιο Κρήτης
HY359 – Διαδικτυακός Προγραμματισμός (Web Programming)
Χειμερινό Εξάμηνο 2017
Διδάσκων: Παναγιώτης Παπαδάκος
Υπεύθυνος Βοηθός: Πέτρος Ζερβουδάκης / Παναγιώτης Παπαδάκος

4η Σειρά Ασκήσεων

Διάρκεια: 2/12 – 15/12
Αξία: 8% του τελικού σας βαθμού
Θεματική ενότητα : Συνεργατικό git / Εμπλουτισμός Server / REST σχεδίαση

Η άσκηση αυτή αποτελεί μέρος της προετοιμασίας για το code sprint και έχει ως στόχο την εξοικοιωσή σας με το git σε πολυχρηστικό περιβάλλον τον εμπλουτισμό της lq πλατφόρμας με νέα λειτουργικότητα, καθώς και τη σχεδίαση ενός REST API.

Για να υλοποιήσετε τη συγκεκριμένη λειτουργικότητα για την άσκηση 2 θα πρέπει να χρησιμοποιήσετε τον κώδικα που σας έχει δοθεί. Συγκεκριμένα πρέπει να κάνετε pul το repository https://bitbucket.org/papadako/lq.git, του οποίου αντίτυπο του project μπορείτε να χρησιμοποιήσετε σαν βάση για το δικό σας project (είναι maven web-app project). Θα γίνονται αλλαγές στον συγκεκριμένο κώδικα ανάλογα με τις ανάγκες της εργασίας. Σε περίπτωση bugs/issues στον παραπάνω κώδικα, μπορείτε να ανοίγετε issues στον issue tracker του repository ώστε να παίρνετε και το credit!

ΠΡΟΣΟΧΗ! Το συγκεκριμένο repository δεν πρέπει να γίνει clone μέσα στο δικό σας git repository με τις ασκήσεις. Αντί αυτού μπορείτε να το κάνετε clone κάπου εξωτερικά και να πάρετε τα αρχεία που σας δίνονται για να τα χρησιμοποιήσετε στο δικό σας maven project. Αν το αντιγράψετε εσωτερικά στον φάκελο a4, φροντίστε να διαγράψετε το a4/lq/.git directory.

Οι ομάδες 2 ατόμων θα πρέπει να κάνουν και τις 3 ασκήσεις ενώ οι ομάδες 1 ατόμου αρκεί να υλοποιήσουν τη δεύτερη και την τρίτη.

Άσκηση 1. Χρήση git σε πολυχρηστικό περιβάλλον (Για ομάδες 2 ατόμων) [10%]

Στα πλαίσια της συγκεκριμένης άσκησης θα πρέπει να αποφασιστεί μεταξύ των μελών της όμαδας ποιο repository θα χρησιμοποιείται από κοινού για τις ομαδικές ασκήσεις. Ο υπεύθυνος του συγκεκριμένου repository (ας τον ονομάσουμε **user1**) θα πρέπει να το κάνει share και στο δεύτερο μέλος της ομάδας (ας τον ονομάσουμε **user2**). Για να κάνετε share πατάτε στο δεύτερο εικονίδιο απο δεξιά με το ονομα "Share" στην κεντρική οθόνη του συγκεκριμένου repository στο bitbucket. Προσοχή θα πρέπει να δώσετε Write access στο δεύτερο μέλος ώστε να μπορεί να κάνει push αλλαγές στο online repository. Πλέον και ο user1 και user2 μπορούν να κάνουν clone το online repository.

Στην άσκηση αυτή θα χρησιμοποιήσουμε την ιδέα των branches. Ένα branch αποτελεί ένα διαφορετικό παρακλάδι στην ιστορία του κώδικα και χρησιμοποιείται για την ανεξάρτητη ανάπτυξη τμημάτων κώδικα τα οποία συνενώνουμε στη συνέχεια με την κεντρική ιστορία του κωδικά μας (συνήθως το branch master). Για παράδειγμα η επίλυση ενός συγκεκριμένου bug θα μπορούσε να γίνεται στο δικό του branch (π.χ. bug1) όπως και η ανάπτυξη ενός συγκεκριμένου χαρακτηριστικού (π.χ. feature1).

Οπότε στη περίπτωσή μας, στο παραπάνω repository θα πρέπει να υπάρχουν τα παρακάτω 3 branches:

1. **master (το κεντρικό branch της εργασίας σας με το οποίο θα επικοινωνείτε μεταξύ σας και με το μάθημα). Εδώ δεν θα κάνουμε ποτέ commit, παρά μόνο merge τα commits που υπάρχουν στα υπόλοιπα branches**
2. **user1 (το branch εργασίας του φοιτητή user1)**
3. **user2 (το branch εργασίας του φοιτητή user2)**

Για τη δημιουργία των branches καθώς και την ενημέρωση του online repository ο **user1** θα πρέπει να εκτελέσει τις παρακάτω εντολές

`git branch user1`	Δημιουργία branch user1
`git checkout user1`	Ενεργοποίηση branch user1

`git push --set-upstream origin user1`	Ενημέρωση online repository για την ύπαρξη του branch user1
`git branch user2`	Δημιουργία branch user2
`git checkout user2`	Ενεργοποίηση branch user2
`git push --set-upstream origin user2`	Ενημέρωση online repository για την ύπαρξη του branch user2

Πλέον αν ο κάθε χρήστης κάνει git pull θα έχει local όλα τα branches (master, user1 και user2).

Τώρα μπορούμε να ξεκινήσουμε την παράλληλη ανάπτυξη κώδικα. Οπότε καλείστε να εκτελέσετε τα παρακάτω βήματα:

updates to user1

- `git checkout user1` (Ο χρήστης user1 ενεργοποιεί το branch user1)
- Ο χρήστης user1 δημιουργεί το directory a4.1 μέσα στο directory a4 του repository.
- Δημιουργεί το αρχείο TEAM με όνομα του και με το νούμερο της ομάδας.
- Δημιουργεί το αρχείο user1 και γράφει ένα τυχαίο μήνυμα.
- (`git add ...` και `git commit …`) Κάνει commit τις συγκεκριμένη αλλαγές
- `git push` (ανεβάζει τις αλλαγές online – Οι αλλαγές αυτές είναι πλέον στο branch user1 και στο bitbucket)

update online master

- Πλέον θέλει να δώσει τις αλλαγές αυτές και στον χρήστη user2. Οπότε πρέπει να ενημερώσει το branch master. Οπότε εκτελεί τα παρακάτω:
  - `git checkout master` (Ενεργοποίηση branch master)
  - `git pull` (Πριν ενημερώσουμε το branch master θα πρέπει κατεβάσουμε τυχόν αλλαγές που υπάρχουν στο online branch master)
  - `git merge user1` (Ενσωμάτωση του branch user1)
  - `git push` (Ενημέρωση online repository με τις αλλαγές)
  -

updates to user2

- Ο χρήστης user2 πλέον μπορεί να φέρει αυτές τις αλλαγές στο δικό του local repository
  - `git checkout master` (Ενεργοποίηση branch master)
  - `git pull` (Πριν ενημερώσουμε το branch user2 θα πρέπει κατεβάσουμε τυχόν νέες αλλαγές που υπάρχουν στο online branch master)
  - `git checkout user2` (Ενεργοποίηση branch user2)
  - `git merge master` (Ενσωμάτωση του branch master στο user2)
  - `git push (Ενημέρωση online repository με τις αλλαγές)`
- Ο χρήστης user2 πλέον έχει όλες τις αλλαγές που έκανε ο user1.

Καλείστε τώρα να κάνετε τα αντίστοιχα βήματα για τροποποιήσεις που θα κάνει ο user2 και τελικά θα λάβει ο user1. Θα πρέπει δηλαδή αντίστοιχα ο user2 να φτιάξει ένα αρχείο user2 στο directory user2 που θα κάνετε commit και να κάνει update το αρχείο TEAM κ.ο.κ.

Να θυμάστε τα παρακάτω:

- ότι κάθε στιγμή μόνο ένα branch είναι ενεργό, αυτό που αναφέρει η εντολή `git branch`
- για να αλλάξουμε το active branch γραφουμε `git checkout branchName`
- μην ξεχνάτε να κάνετε συχνά merge τις αλλαγές που γίνονται στο master για να αποφύγετε τυχόν conflicts

Αφιερώστε χρόνο να κατανοήσετε το τι συμβαίνει παραπάνω αφού λίγο πολύ θα αποτελεί τον τρόπο με τον οποίο θα συνεργάζεστε. Παραπάνω πληροφορίες για το git στο http://papadako.github.io/git-a-little-tale/ καθώς και στο τεράστιο πλήθος από online πληροφορίες (π.χ. https://git-scm.com/book/en/v2/Git-Branching-Branches-in-a-Nutshell)

Άσκηση 2. LQ - Ψηφοφορία [50%]

Στην άσκηση αυτή θα επεκτείνουμε την A3 έτσι ώστε να μπορούμε να δημιουργήσουμε policy initiatives προς ψήφιση και να μπορούν οι χρήστες της πλατφόρμας μας να τα ψηφίσουν. Καντε git pull στο repository που σας έχει δοθεί για να μπορείτε να χρησιμοποιήσετε τις νέες κλάσεις.

Θέλουμε να προσφέρουμε την παρακάτω λειτουργικότητα:

- Κάθε χρήστης μπορεί να δημιουργήσει ένα ή παραπάνω policy initiatives (θέματα προς ψήφιση). Κάθε policy initiative έχει ένα δημιουργό, ένα description (free-text), μια κατηγορία (π.χ. οικολογία) και ένα τίτλο **(non-active policy initiatives).**
- Κάθε χρήστης θα μπορεί να δει όλα τα policy initiatives που έχει δημιουργήσει
- Ένα policy initiative μπορεί να τεθεί προς ψηφοφορία μέχρι κάποια χρονική στιγμή που ορίζει ο δημιουργός του **(active policy initiatives)**. Όταν περάσει ο χρόνος αυτός τότε δεν είναι πλέον ενεργό για ψηφοφορία
- Κάθε χρήστης θα πρέπει να μπορεί να κάνει update τα δικά του non-active policy initiatives
- Κάθε χρήστης θα μπορεί να δει όλα τα active policy initiatives τη συγκεκριμένη χρονική στιγμή
- Κάθε χρήστης μπορεί να κάνει upvote/downvote τα active policy initiatives (κάθε χρήστης ψηφίζει μόνο μία φορά)
- Ένας χρήστης μπορεί να κάνει update τη ψήφο του εφόσον δεν έχει λήξει ο χρόνος της ψηφοφορίας
- Ένας χρήστης μπορεί να δεί όλα τα policy initiatives που έχουν λήξει **(ended policy initiatives)** και τα συγκεντρωτικα αποτελέσματα για το καθένα

Δημιουργήστε πάνω από 2 χρήστες για να δείτε ότι όλα δουλεύουν σωστά. Μπορείτε να κάνετε login με διαφορετικούς χρήστες χρησιμοποιώντας διαφορετικούς browsers ή το incognito mode.

Χρησιμοποιήστε τη λειτουργικότητα που σας έχει δοθεί για την υποστήριξη των παραπάνω. Μπορείτε να συνεχίζετε να χτίζετε πάνω στο πρότυπο UI που σας έχει δοθεί.

Άσκηση 3. LQ – REST [25%]

Στη συγκεκριμένη άσκηση καλείστε να σχεδιάσετε και να περιγράψετε ένα REST API το οποίο να μας δίνει τη δυνατότητα να δημιουργήσουμε, τροποποιήσουμε, διαγράψουμε και να ανακτήσουμε πληροφορία που σχετίζεται με τα resources που χρησιμοποιούμε στην lq πλατφόρμα μας. Τα resources που μας ενδιαφέρουν είναι τα εξής: user, policy initiatives (μπορούμε να τα χωρίσουμε σε κατηγορίες όπως active, non-active και ended), καθώς και τα votes. Περιγράψτε τις HTTP μεθόδους που θα χρησιμοποιούσατε καθώς και την πληροφορία (ιδανικά σε JSON) που θα έπρεπε να συμπλεριλαμβάνεται στα requests/responses μαζί με τα κατάλληλα status codes. Για τη συγκεκριμένη άσκηση δεν μας ενδιαφέρουν θέματα access/authentication. Η περιγραφή του μπορεί να γίνει σε ένα αρχείο που θα κάνετε commit μέσα στο lq directory.

Το υπόλοιπο 15% του βαθμού θα κατανεμηθεί βάσει των παρακάτω 3 κριτηρίων:

- **jshint, html validator, code quality – 5%:** θα κρίνεται από το αν η σελίδα σας δεν εμφανίζει λάθη/warnings στον jshint, στον validator, καθώς και στη γενική ποιότητα του κώδικά σας
- **ελκυστικότητα εμφάνισης σελίδων (στυλιστική συνέπεια) – 5%**
- **git – 5%:** θα κρίνεται από τη σωστή χρήση του git (π.χ. να υπάρχουν αρκετά commits που να περιγράφουν με σαφήνεια πως κάνατε την άσκηση, με κατανοητή περιγραφή, καθαρό ιστορικό, κτλ.)

Σημειώσεις:

- Μη ξεχνάτε τη χρήση του "use strict"; για την JavaScript.
- Το parse ενός JSON string γίνεται με χρήση της JSON.parse(str) που επιστρέφει το javascript object που αντιστοιχεί στο str, δεδομένου ότι το str είναι μια σωστή αναπαράσταση JSON .
- Μπορείτε να χρησιμοποιήσετε κάποιον online linter όπως ο jshint http://jshint.com/ για να βελτιώσετε την ποιότητα του js κώδικά σας.
- Προτείνεται η χρήση του netbeans σαν IDE. Επίσης προσπαθήστε να οργανώσετε με όμορφο τρόπο τα servlets που θα χρησιμοποιήσετε.
- Είναι ευκαιρία να διορθώσετε τυχόν προβλήματα που έχετε με τον κώδικα προηγούμενων ασκήσεων. Άλλωστε η συγκεκριμένη άσκηση είναι ένα βήμα πριν από το sprint, οπότε δείτε που μπορείτε να κάνετε βελτιώσεις. Επίσης μπορείτε όπου φαίνεται χρήσιμο να κάνετε χρήση της JSP αντί των servlets.

- Η εφαρμογή μας πρέπει να χρησιμοποιεί μόνο ajax requests για την ανανέωσή της και όχι φόρμες. Επίσης εξακολουθούμε να μιλάμε για μια SPA εφαρμογή.

Τρόπος Παράδοσης

Οι ασκήσεις θα παραδίδονται μόνο μέσω git, σύμφωνα με τις οδηγίες που σας έχουν δοθεί. Συγκεκριμένα στο repository σας στο bitbucket το οποίο θα πρέπει να έχει γίνει ήδη share στο hy359, στο folder a4 θα πρέπει να υπάρχει ένα subfolder όπου θα περιέχεται το maven project της εργασίας σας με όνομα lq (όμορφη δομή packages/κώδικα). **Θα πρέπει να φροντίσετε ότι όλα όσα έχετε κάνει έχουν γίνει σωστά commit και βρίσκοντα online στο bitbucket. Αφού όλα είναι ok θα πρέπει να κάνετε tag την άσκηση σας "git tag a4" και να ανεβάσετε το tag στο bitbucket μέσω της εντολής "git push --tags".**

Προγραμματίστε καλά το χρόνο σας και αποφύγετε να ασχοληθείτε με την εργασία τελευταία στιγμή!

Καλό είναι σαν τελευταίο έλεγχο πριν σταματήσετε να ασχολείστε με την εργασία να κάνετε clone το repository σας κάπου τοπικά και να ελέγξετε αν υπάρχει κάποιο πρόβλημα. Με αυτόν τον τρόπο άλλωστε θα βαθμολογηθείτε/εξεταστείτε.

Στις 23:55 της 15/12 θα γίνει αυτόματο pull από όλα τα repositories που έχουν γίνει share στο hy359 και βάσει αυτών θα βαθμολογηθείτε. Εκπρόθεσμες ασκήσεις **δεν θα γίνονται δεκτές**.

Αντιγραφή

Σε περίπτωση αντιγραφής θα μηδενίζονται άμεσα οι εργασίες όλων των εμπλεκόμενων.

Και λίγα λόγια για το μεράκι....

Θέσετε ως στόχο να υλοποιήσετε τις εργασίες σας με μεράκι και δημιουργικότητα και να αναδείξετε με όμορφο τρόπο το χρόνο που αφιερώσατε! Μην αφήνετε τα πράγματα στην τύχη και δώστε κάτι από τον εαυτό σας και το χαρακτήρα σας!

Καλή συνεργασία