Κέντρο Αριστείας Ανοιχτού Λογισμικού ΑΠΘ

Σεμινάριο #3

Αρχιτεκτονικές έξυπνων πόλεων:
Έρευνα, Συμπεράσματα και Σχετικά Εργαλεία
Ανάπτυξης Ανοιχτού Λογισμικού

Κακαρόντζας Γεώργιος
Καθηγητής Εφαρμογών
Τμήμα Μηχανικών Πληροφορικής ΤΕ
ΤΕΙ Θεσσαλίας

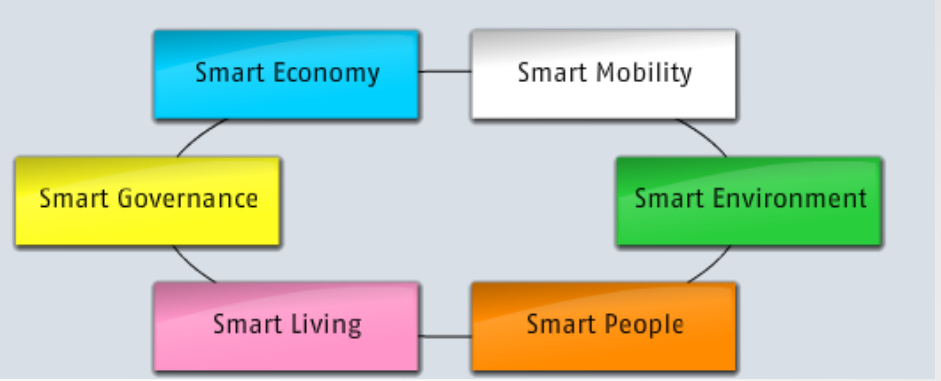
Ατζέντα

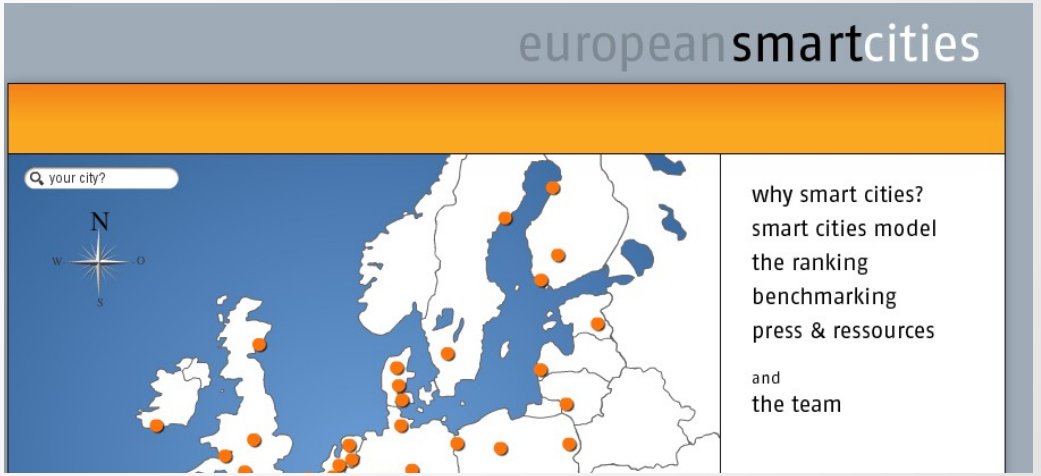
- Έρευνα για τις αρχιτεκτονικές έξυπνων πόλεων
- Συμπεράσματα για τις αρχιτεκτονικές έξυπνων πόλεων: ένα εννοιολογικό αρχιτεκτονικό πλαίσιο
- Εργαλεία ανοιχτού λογισμικού για προγραμματισμό εφαρμογών με βάση το εννοιολογικό αρχιτεκτονικό πλαίσιο
- Σύντομο παράδειγμα εργαλείων ανάπτυξης

Έρευνες για Αρχιτεκτονικές Έξυπνων Πόλεων

Χαρακτηριστικά έξυπνων πόλεων

- Έξυπνη οικονομία: επιχειρηματικό πνεύμα, καινοτομία κ.α.
- Έξυπνη κινητικότητα: διεθνής προσβασιμότητα, διαθεσιμότητα πληροφορικών υποδομών κ.α.
- Έξυπνο περιβάλλον: περιβαλλοντική προστασία κ.α.
- Έξυπνοι άνθρωποι: δημιουργικότητα, δια βίου μάθηση κ.α.
- Έξυπνη διαβίωση: πολιτισμικές υποδομές, συνθήκες υγιεινής κ.α.
- Έξυπνη διακυβέρνηση: συμμετοχή στη λήψη αποφάσεων, δημόσιες και κοινωνικές υπηρεσίες κ.α.





Στόχοι της έρευνας

- ***G. Kakarontzas, L. Anthopoulos, D. Chatzakou, A. Vakali: "An application architecture framework for smart cities", 2014***
- Αναφέρουμε τα αποτελέσματα έρευνας με βάση ερωτηματολόγια και χρησιμοποιώντας αυτά τα αποτελέσματα εντοπίζουμε σημαντικές ποιοτικές και λειτουργικές απαιτήσεις για τις έξυπνες πόλεις.
- Σημαντικές ποιοτικές απαιτήσεις περιλαμβάνουν: τη διαλειτουργικότητα, τη ευχρηστία, την ασφάλεια, τη διαθεσιμότητα, τη δυνατότητα ανάκαμψης και την ευκολία συντήρησης.
- Παρατηρούμε, επίσης, θέματα που αφορούν την επιχειρηματικότητα γύρω από τις έξυπνες πόλεις, π.χ. την έλλειψη επιχειρηματικού σχεδίου στις περισσότερες περιπτώσεις.
- Στο τομέα της αρχιτεκτονικής λογισμικού παρουσιάζουμε ένα εννοιολογικό αρχιτεκτονικό πλαίσιο με βάση τα αρχιτεκτονικά πρότυπα τα οποία αντιμετωπίζουν τα διαπιστωμένα ποιοτικά χαρακτηριστικά. Το εννοιολογικό πλαίσιο μπορεί να χρησιμοποιηθεί ως σημείο εκκίνησης για τα έργα πραγματικών έξυπνων πόλεων.

Χαρακτηριστικά έρευνας

- Πέντε γενικές ομάδες ερωτήσεων: “σχετικές ερωτήσεις με αρχιτεκτονική”, “πηγές δεδομένων”, “διαχείριση-οργάνωση και χρηματοδότηση”, “διαχείριση κρίσιμων ζητημάτων/ορόσημα” και “αντικειμενικοί στόχοι και προσανατολισμός του έργου”.
- 18 μηχανικοί έξυπνων πόλεων: Digital City of Trikala (Greece), Den Haag (The Netherlands), Amsterdam (The Netherlands), Brisbane (Australia), Waterfront Toronto (Canada), Turin (Italy), Thermi (Greece), Thessaloniki (Greece), Heraklion (Greece), Zurich (Switzerland), Geneva MAN (Switzerland), smart city of Melbourne (Australia), green city of Queensland (Australia), green city of Roland Victoria (Australia), smart city of Brisbane (Australia).
- Στη συνέχεια παρουσιάζουμε κάποια βασικά ερωτήματα.

Υπηρεσίες/Υποδομές

- Παρεχόμενες υπηρεσίες
  - Περιβαλλοντικές υπηρεσίες: 67%
  - Επικοινωνία: 56%
  - Υπηρεσίες ηλεκτρονικής διακυβέρνησης: 50%
- Υποδομή:
  - Wired backbone: 67%
  - Wireless backbone: 61%
  - Αφιερωμένη μισθωμένη υποδομή: 0%

Πηγές Δεδομένων #1

- Συλλογή δεδομένων
  - Απευθείας συλλογή δεδομένων: 72%
  - Τοπική αποθήκευση: 44%
  - Έμμεση συλλογή δεδομένων: 33%
  - Εξωτερική αποθήκευση (cloud): 22%
- Πρόσβαση στο προσωπικό του δήμου
  - Με βάση το ρόλο (role-based): 39%
  - Εξαρτάται από το project: 22%

Πηγές δεδομένων #2

- Πρόσβαση των παραγόντων της πόλης
  - Εξαρτάται από το project: 60%
- Επίπεδο πρόσβασης των τελικών χρηστών
  - Ατομική πρόσβαση: 33%
  - Εξαρτάται από το project: 28%
- Σε ποιον ανήκουν τα δεδομένα;
  - Δήμος: 50%
  - Οργανισμό έξυπνης πόλης: 39%
  - Διάφοροι παράγοντες: 28%

Διαχείριση-οργάνωση &χρηματοδότηση #1

- Ποιοι εμπλέκονται στη μέτρηση απόδοσης των υπηρεσιών;
  - Δήμος: 100%
  - Τοπικές υπηρεσίες μεταφοράς: 50%
  - Τοπικές ΜΜΕ: 39%
- Ποιος χρηματοδοτεί το έργο?
  - Δημόσιος και Ιδιωτικός τομέας: 56%
  - Δημόσιος τομέας: 44%
- Από που προέρχεται η χρηματοδότηση;
  - Κυβέρνηση: 50%
  - Διάφοροι: 50%
  - ΕΕ: 39%
  - Πάροχος υπηρεσίας: 28%

Διαχείριση-οργάνωση &χρηματοδότηση #2

- Ποιος είναι υπεύθυνος για το marketing;
  - Δημόσιος οργανισμός: 67%
  - Δήμος: 67%
  - Ιδιωτική εταιρία: 56%
- Ποιος παρέχει υπηρεσίες έναντι αμοιβής;
  - Δεν γνωρίζω: 44%
  - Οι υπηρεσίες είναι ελεύθερες χωρίς αμοιβή: 17%
  - Ιδιωτικές εταιρίες: 17%
- Ποιος συλλέγει τα κέρδη;
  - Δήμος: 22%
  - Διάφοροι φορείς: 22%
- Ποιος διαχειρίζεται τις υπηρεσίες?
  - Δήμος: 72%
  - Οργανισμός έξυπνης πόλης: 67%

Διαχείριση κρίσιμων ζητημάτων/ορόσημα

- Ποιος είναι υπεύθυνος για την επιθεώρηση των υπηρεσιών;
  - Δήμος: 67%
  - Διάφοροι φορείς: 28%
  - Οι εμπλεκόμενοι φορείς στο έργο: 22%
- Κάνετε αξιολόγηση των υπηρεσιών και πόσο συχνά;
  - Τακτικές αξιολογήσεις: 61%
- Πόσο συχνά κάνετε συντήρηση των υπηρεσιών και των υποδομών;
  - Αβέβαιοι: 50%
  - Περισσότερο από μία φορά το χρόνο: 17%
  - Σπάνια: 17%
  - Ετησίως: 11%
  - Εξαρτάται από τις προδιαγραφές της υπηρεσίας: 5%

Αντικειμενικοί στόχοι/προσανατολισμός

- Πιθανοί επιχειρηματικοί στόχοι
  - Ενθάρρυνση Καινοτομίας: 33%
  - Οικονομία πόρων: 33%
  - Βελτίωση του βαθμού εξωστρέφειας της πόλης: 22%
- Εσωτερική οργανωτική δομή:
  - Ασαφής: 61%
  - SOE (State-Owned Enterprise): 28%
  - Οργανισμός έξυπνης πόλης: 17%
- Ποιες ήταν οι αρχικά παρεχόμενες υπηρεσίες;
  - Έξυπνες μεταφορές: 22%
  - Διάλογοι Online: 22%
- Προσθέσατε υπηρεσίες και ποιες;
  - Πάρα πολλές για να αναφερθούν μεμονωμένα: 17%
- Ακυρώσατε υπηρεσίες και ποιες;
  - Έξυπνη μεταφορά: 6%

Ερευνητική προσέγγιση

- Για να επιλέξουμε τα κατάλληλα πρότυπα αρχιτεκτονικής για την αρχιτεκτονική της έξυπνης πόλης θα χρησιμοποιήσουμε πρώτα τα αποτελέσματα του ερωτηματολογίου για να κατανοήσουμε τις πιο σημαντικές λειτουργικές και ποιοτικές προδιαγραφές για έξυπνες πόλεις.
- Στη συνέχεια, θα χρησιμοποιήσουμε αυτές τις απαιτήσεις να προτείνουμε μια αρχιτεκτονική προσέγγιση για έξυπνες πόλεις με βάση τα αρχιτεκτονικά πρότυπα.
- Οι απαιτήσεις για την ποιότητα μπορεί να οριστούν χρησιμοποιώντας το μοντέλο ποιότητας ISO-25010. Παρέχει τις διάφορες κατηγορίες ποιότητας που πρέπει να εξετάσουμε με βάση τα πορίσματα του ερωτηματολογίου.

Διαλειτουργικότητα

- Τα πολλά διαφορετικά είδη εφαρμογών και υπηρεσιών καταδεικνύουν την ανάγκη για μια γενική αρχιτεκτονική για κάθε εφαρμογή, εξασφαλίζοντας συγχρόνως την επικοινωνία μεταξύ αυτών των εφαρμογών με ομοιογενή τρόπο (αν χρειάζεται).
- Ως εκ τούτου, η διασφάλιση της διαλειτουργικότητας μεταξύ αυτών των διαφορετικών εφαρμογών και υπηρεσιών είναι απαραίτητη.

Απόδοση

- Παρέχοντας μόνο ένα μικρό ποσοστό εφαρμογών σχετιζομένων με την ασφάλεια και την υγεία (22% το καθένα), οι έξυπνες πόλεις φαίνεται να έχουν σχετικά χαμηλή ανάγκη (προς το παρόν) για τη υποδομές που εξασφαλίζουν απόκριση σε πραγματικό χρόνο (ISO: time behaviour)
- Η έλλειψη μισθωμένων γραμμών επίσης δείχνει πως η ανάγκη για απόδοση από άποψη ταχύτητας απόκρισης είναι μικρή προς το παρόν. Έτσι η απόδοση ξανά φαίνεται να είναι δευτερεύον ζήτημα.

Έλεγχος Ταυτότητας & Διαλειτουργικότητα

- Η κυρίως τοπική αποθήκευση (44%) σε συνδυασμό με το πλήθος των παρόχων υπηρεσιών δείχνει πως τα δεδομένα μπορούν να αποθηκευτούν σε πολλές διαφορετικές θέσεις. Κάθε εφαρμογή και η οργάνωση που τη φιλοξενεί είναι τελικά υπεύθυνη για την προστασία των δικών της δεδομένων, αλλά την ίδια στιγμή θα πρέπει να είναι δυνατό οι άλλες εφαρμογές να μπορούν να έχουν πρόσβαση σ' αυτά τα δεδομένα εξ αποστάσεως με προβλεπόμενους τρόπους.
- Ως εκ τούτου στο μοντέλο ISO η πτυχή της ασφάλειας (security) που αφορά τον έλεγχο ταυτότητας (authenticity) γίνεται σημαντική, καθώς και επίσης πτυχή της διαλειτουργικότητας (interoperability) του χαρακτηριστικού της συμβατότητας (compatibility) ξανά φαίνεται να είναι ιδιαίτερα σημαντική.

Ευχρηστία

- Η πληθώρα των παρεχόμενων υπηρεσιών δείχνει επίσης ότι η ευχρηστία μπορεί να είναι σημαντικό θέμα, ειδικά αν σκεφτεί κανείς τα διαφορετικά είδη των διασυνδέσεων χρήστη, που μπορεί να απαιτούνται για να υποστηριχθούν όλα αυτά τα διαφορετικά είδη των εφαρμογών.
- Επίσης η πληθώρα των διαφορετικών τύπων χρηστών καταδυκνύει την ανάγκη για αυξημένη ευχρηστία.
- Έτσι στο μοντέλο ISO το χαρακτηριστικό 'Usability' φαίνεται να είναι σημαντικό.

Παροχή εξουσιοδότησης

- Η πρόσβαση στα δεδομένα είναι περίπλοκη, δεδομένου ότι πολλά διαφορετικά μέρη μπορούν να έχουν πρόσβαση σε δεδομένα με πολλούς διαφορετικούς τρόπους.
- Και πάλι αυτό δείχνει ότι κάθε εφαρμογή θα πρέπει να αναλάβει την ευθύνη για την παροχή του δικού της μηχανισμού ελέγχου ταυτότητας (authenticity).
- Επίσης η πρόσβαση με βάση το ρόλο απαιτείται για το διοικητικό προσωπικό. Επομένως και το χαρακτηριστικό της παροχής εξουσιοδότησης στην κατηγορία της ασφάλειας φαίνεται σημαντικό (authorization).

Δυνατότητα ανάκαμψης

- Αν και υπάρχουν αξιολογήσεις των υπηρεσιών, η συντήρηση δεν πραγματοποιείται τακτικά. Αυτό δείχνει η διαθεσιμότητα μπορεί να είναι πρόβλημα αν οι παρεχόμενες υπηρεσίες δε συντηρούνται επαρκώς.
- Επίσης δείχνει ότι υπάρχει ανάγκη για δυνατότητα ανάκαμψης όσον αφορά την ιδιότητα της αξιοπιστίας. Οι υπηρεσίες πρέπει να είναι σε θέση να ανακάμψουν αυτόνομα και γρήγορα σε περιπτώσεις αποτυχίας.

Ευκολία συντήρησης

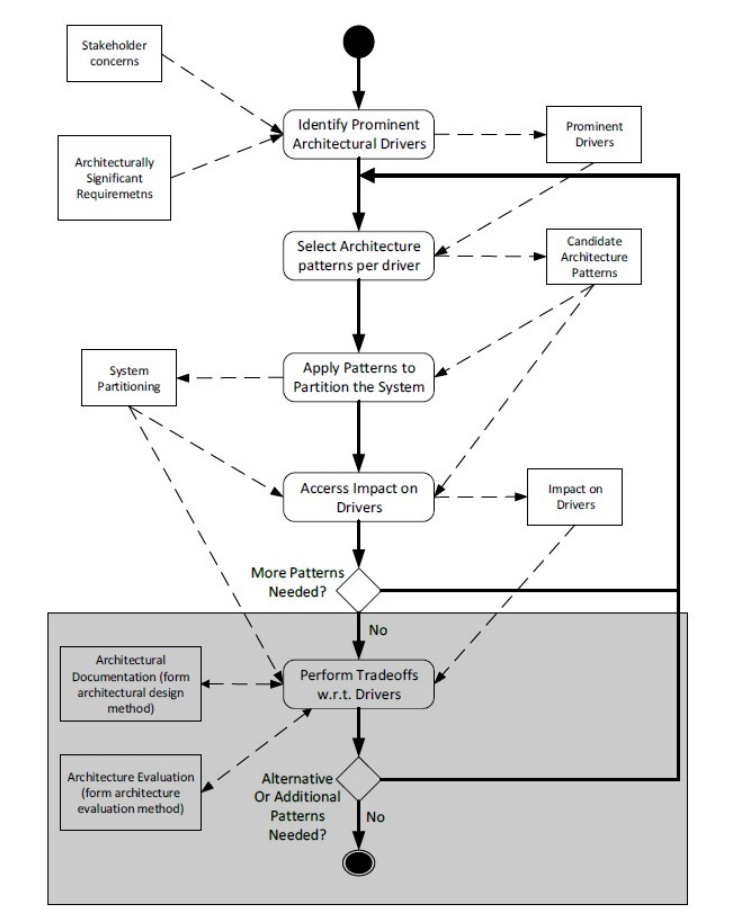
- Η έλλειψη του επιχειρηματικού σχεδίου, σε πολλές περιπτώσεις, καθώς και η ύπαρξη ενός πλήθους πιθανών επιχειρηματικών στόχων, συνεπάγεται ότι τα έργα αυτά μπορούν να θεωρηθούν (προς το παρόν τουλάχιστον) ως μια ομπρέλα κάτω από την οποία εντάσσονται πολλές διαφορετικές εφαρμογές και πως αυτές οι εφαρμογές μπορούν να αναπτυχθούν σε πολλές διαφορετικές κατευθύνσεις στο μέλλον.
- Αυτό δημιουργεί την ανάγκη για τη βελτίωση της συντήρησης, προκειμένου να μπορέσουν οι εν λόγω εξελίξεις να γίνουν με ευκολία. Η ανάγκη αυτή είναι επίσης εμφανής όταν εξετάζει κανείς και την ποικιλομορφία των αρχικά παρεχόμενων υπηρεσιών.

Συμπεράσματα για τις αρχιτεκτονικές έξυπνων πόλεων: ένα εννοιολογικό αρχιτεκτονικό πλαίσιο

Σημαντικές ποιοτικές ιδιότητες

- Διαλειτουργικότητα
- Ευχρηστία
- Πρόσβαση με έλεγχο ταυτότητας και παροχή εξουσιοδότησης
- Διαθεσιμότητα
- Δυνατότητα ανάκαμψης
- Ευκολία συντήρησης

Μέθοδος επιλογής προτύπων

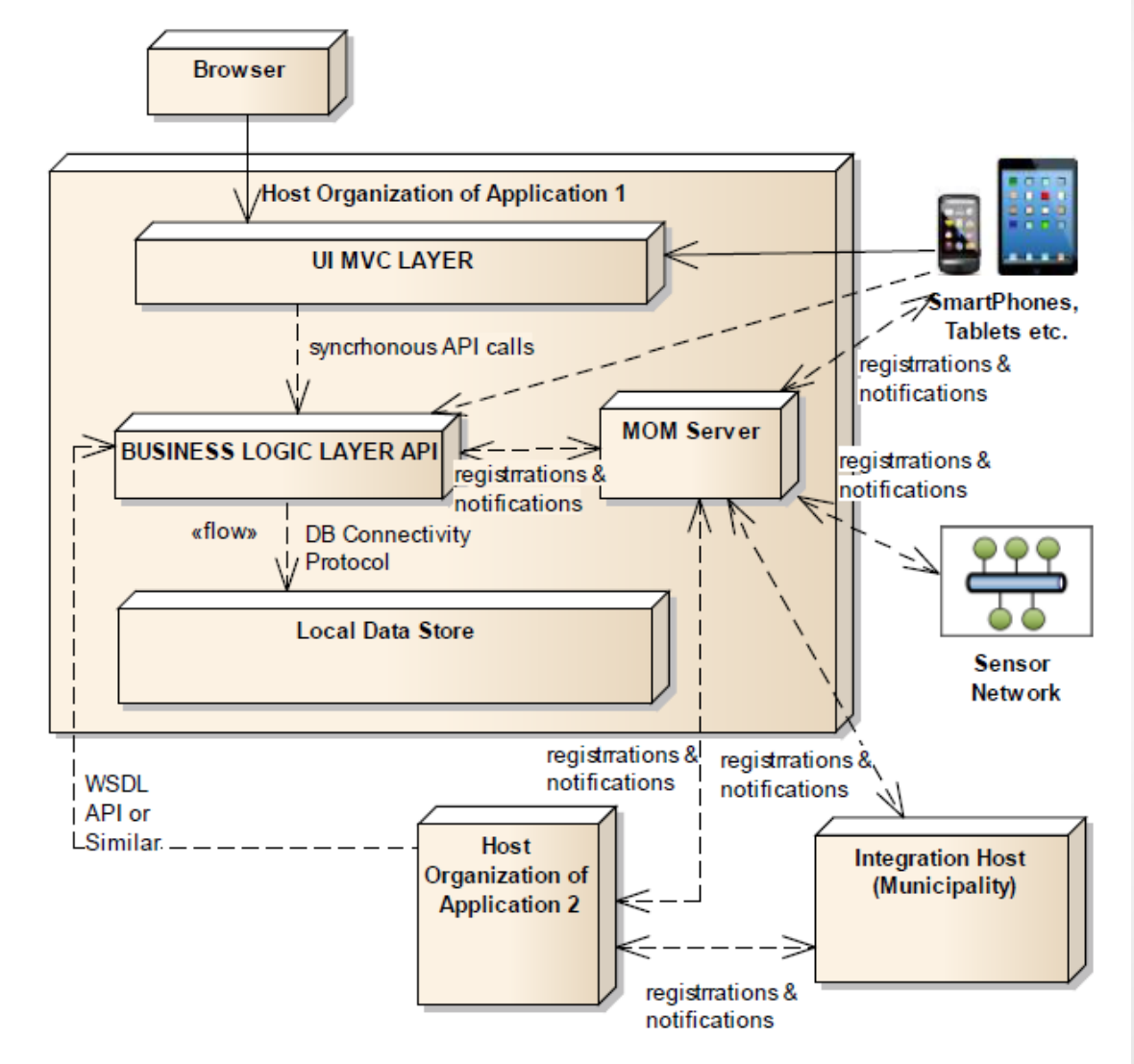


- Χρησιμοποιήσαμε τη μέθοδο Pattern-Driven Architecture Partitioning (PDAP) [Harrison & Avgeriou, 2007] για την επιλογή κατάλληλων αρχιτεκτονικών προτύπων και την εξαγωγή της αρχιτεκτονικής που απαντά στις απαιτήσεις.

Σημαντικά αρχιτεκτονικά πρότυπα

- Διαλειτουργικότητα: API Façade (Gamma et al., 1995.)
- Διαλειτουργικότητα, Ασφάλεια: Layers architectural pattern (Buschmann et al., 1996)
- Ευχρηστία: Model-View Controller (MVC) pattern (Reenskaug et al., 1996), (Buschmann et al., 1996)
- Διαθεσιμότητα και Δυνατότητα ανάκαμψης: Messaging architecture pattern (Hohpe and Woolf, 2003), παροχή failback server.

Εννοιολογικό αρχιτεκτονικό πλαίσιο εφαρμογών για έξυπνες πόλεις



Εργαλεία ανοιχτού λογισμικού για προγραμματισμό εφαρμογών με βάση το εννοιολογικό αρχιτεκτονικό πλαίσιο

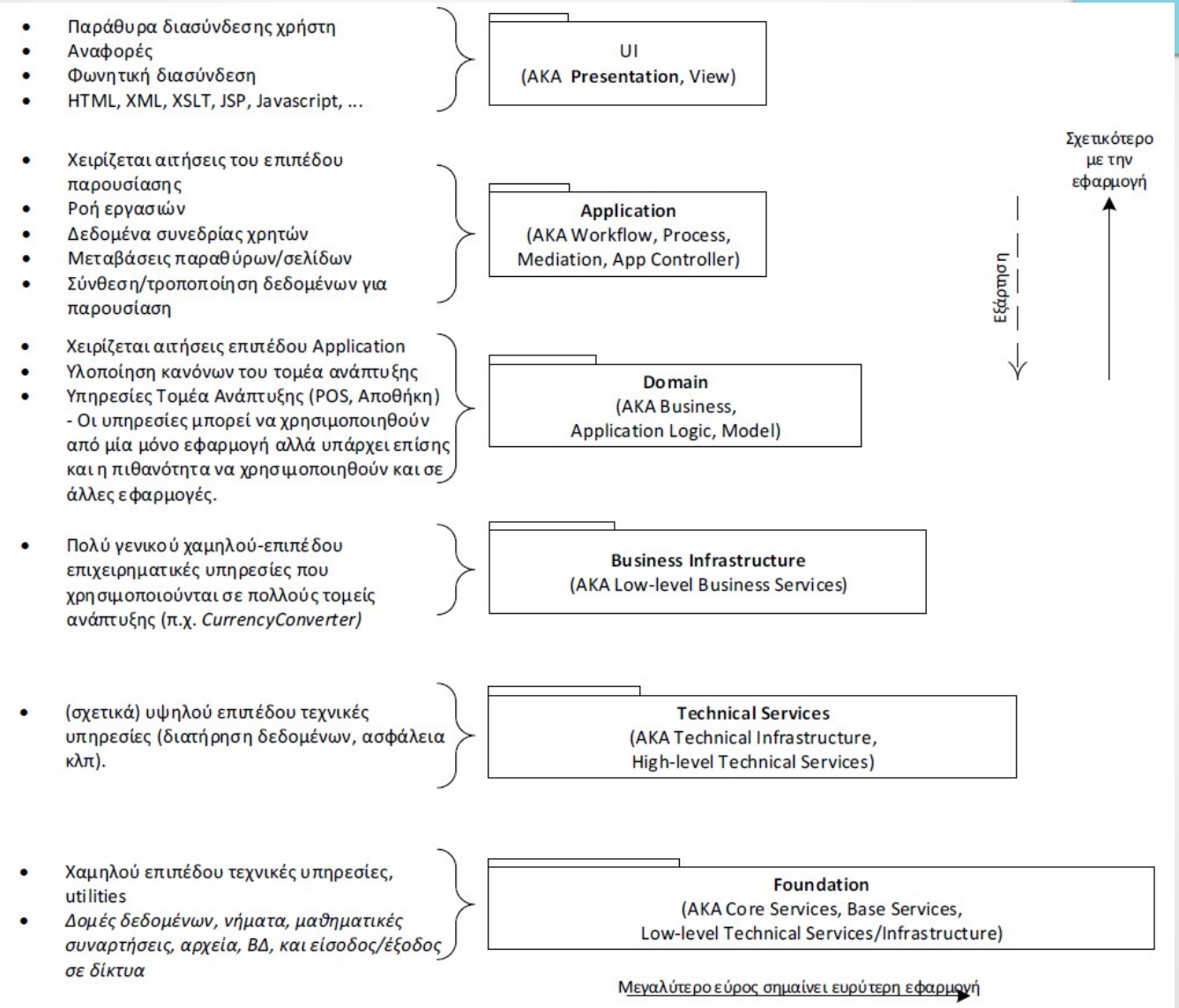
Τι είναι τα επίπεδα (layers)

- Ένα επίπεδο είναι ένα πολύ μεγάλο κομμάτι του συστήματος που αποτελείται από ομάδες κλάσεων, πακέτα ή υποσυστήματα. Παρέχει μία συνεκτική λειτουργία που είναι σχετική με μία μεγάλη σε εμβέλεια άποψη της λειτουργίας του συστήματος. Επίσης τα επίπεδα οργανώνονται με τέτοιο τρόπο ώστε τα υψηλότερα επίπεδα (όπως αυτό της διασύνδεσης χρήστη – UI Layer) να καλούν υπηρεσίες των χαμηλότερων επιπέδων, αλλά όχι το αντίθετο.

Τυπικά επίπεδα

- Τυπικά επίπεδα είναι τα ακόλουθα:
  - Διασύνδεσης χρήστη (User Interface)
  - Λογικής της εφαρμογής και Αντικειμένων του χώρου ανάπτυξης (Application Logic and Domain Objects): Αντικείμενα του συστήματος που αναπαριστούν έννοιες του χώρου ανάπτυξης (π.χ. μία κλάση Sale για τις πωλήσεις) και εκπληρώνουν μία απαίτηση της εφαρμογής, όπως ο υπολογισμός του κόστους μίας πώλησης.
  - Τεχνικές υπηρεσίες (Technical Services): αντικείμενα γενικού σκοπού και υποσυστήματα που παρέχουν τεχνικές υπηρεσίες υποστήριξης, όπως διασύνδεση με βάσεις δεδομένων, καταγραφή λαθών κλπ. Τέτοιες υπηρεσίες είναι συνήθως ανεξάρτητες από την εφαρμογή και επομένως επαναχρησιμοποιήσιμες σε πολλά συστήματα.

Τυπικά επίπεδα (Larmann 2004)



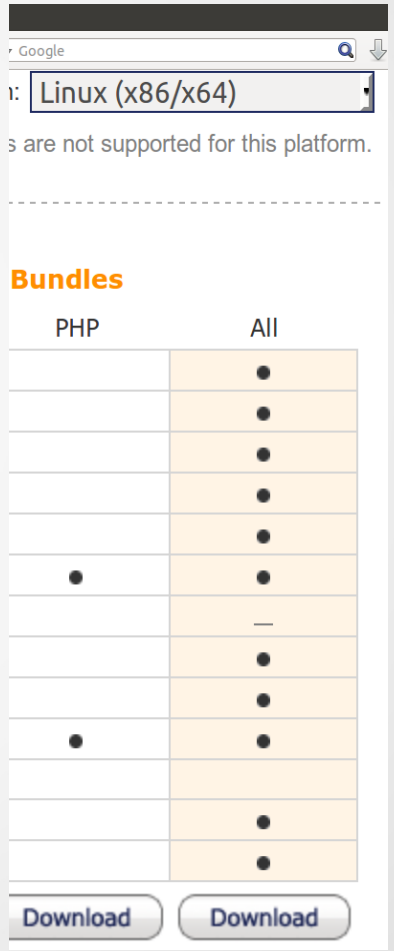
Java Enterprise Edition

- Μπορείτε να αναπτύξετε εφαρμογές επιπέδων με τη Java Enterprise Edition.
- Διαθέσιμα πολλά περιβάλλοντα ανάπτυξης, εξυπηρετητές εφαρμογών, βάσεις δεδομένων ανοιχτού λογισμικού.
- Για παράδειγμα:
  - NetBeans IDE ή Eclipse IDE
  - Glassfish Application Server, Jboss Application Server
  - MySQL, PostgreSQL
  - κ.α.

Σύντομο παράδειγμα εργαλείων ανάπτυξης

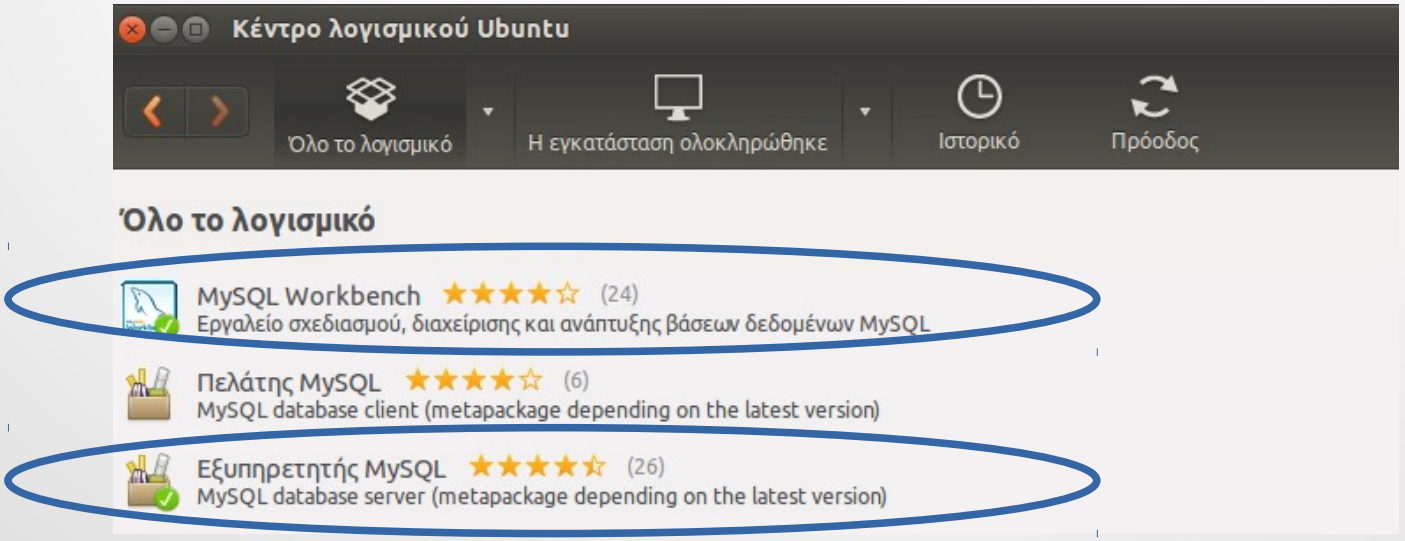
Εγκατάσταση του NetBeans IDE

- Θα επισκεφτείτε την διεύθυνση: http://netbeans.org και θα κατεβάσετε την τελευταία έκδοση του NetBeans
- Επιλέξτε την κατάλληλη γλώσσα και πλατφόρμα (π.χ. English & Linux) και επιλέξτε την έκδοση 'All' γιατί χρειαζόμαστε την πλήρη έκδοση για την ανάπτυξη εφαρμογών Java Enterprise Edition.



Εγκατάσταση της MySQL στο Ubuntu Linux

- Ως βάση δεδομένων θα χρησιμοποιήσουμε τη MySQL.
- Για το Ubuntu Linux μπορούμε να εγκαταστήσουμε την MySQL από το κέντρο λογισμικού. Εγκαταστείστε και τον MySQL εξυπηρετητή αλλά και το MySQL Workbench όπως δείχνει η εικόνα

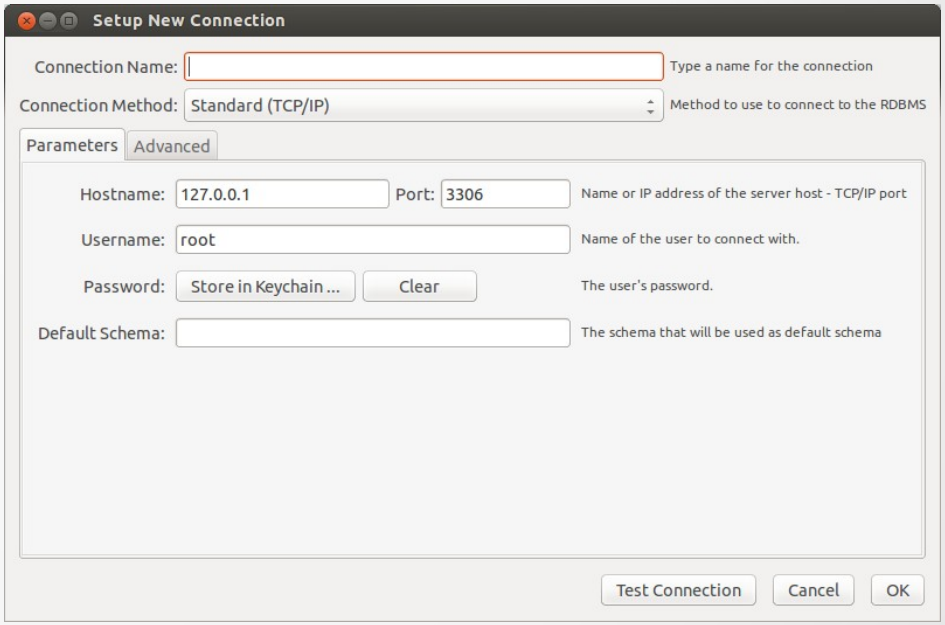


Δοκιμή των προγραμμάτων

- Για να δοκιμάσουμε τα προγράμματα θα κάνουμε μία απλή εφαρμογή διαδικτύου με μία φόρμα στην οποία θα τοποθετούμε κάποια στοιχεία και τα οποία θα εισάγονται σε μία βάση δεδομένων.
- Τα βήματα είναι τα ακόλουθα:
  - Δημιουργία της Βάσης Δεδομένων με το MySQL Workbench
  - Δημιουργία της Εφαρμογής στο NetBeans.
  - Εκτέλεση της εφαρμογής και διαπίστωση της ορθότητάς της.

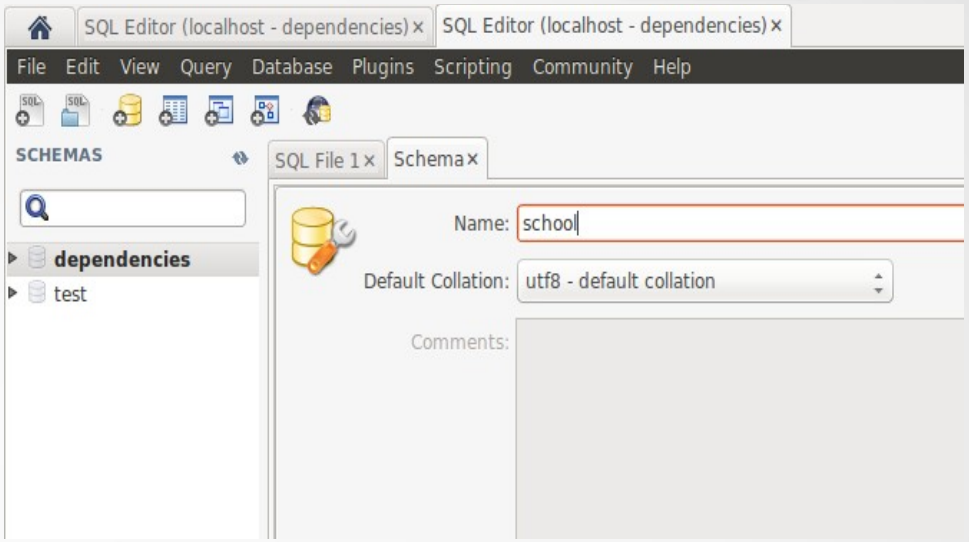
Δημιουργία της Βάσης Δεδομένων
Βήμα 1: δημιουργία τοπικής σύνδεσης

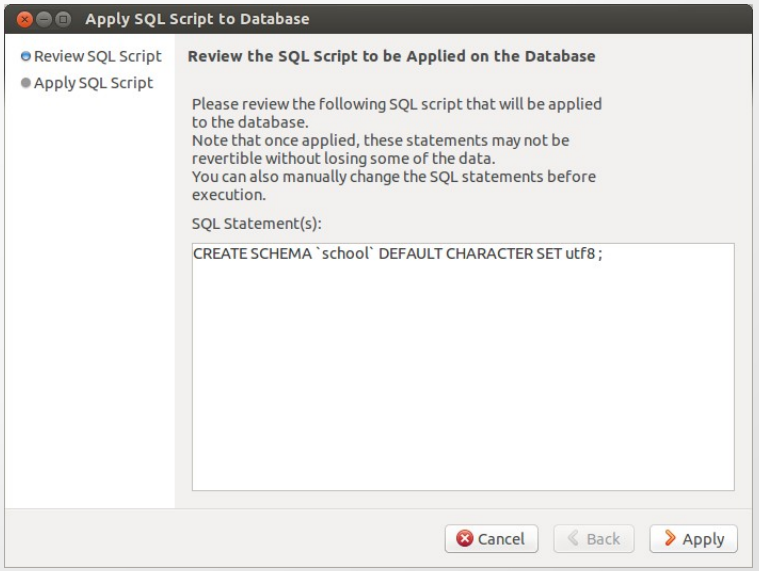
- Ξεκινήστε το MySQL Workbench.
- Στην βασική οθόνη του προγράμματος πατήστε τον σύνδεσμο 'New Connection' για να δημιουργήσετε μία νέα σύνδεση με τον τοπικό εξυπηρετητή (αυτό γίνεται μόνο μία φορά - τις επόμενες απλά επιλέγετε την σύνδεση για να συνδεθείτε).
- Στην καρτέλα της νέας σύνδεσης δίνετε τα στοιχεία που φαίνονται στην εικόνα.
- Επίσης πατήστε το πλήκτρο 'Store in keychain...' για να δώσετε τον κωδικό του χρήστη 'root' της MySQL και Test Connection για να δοκιμάσετε τη σύνδεση.



Δημιουργία της Βάσης Δεδομένων
Βήμα 2: σύνδεση & δημιουργία της ΒΔ

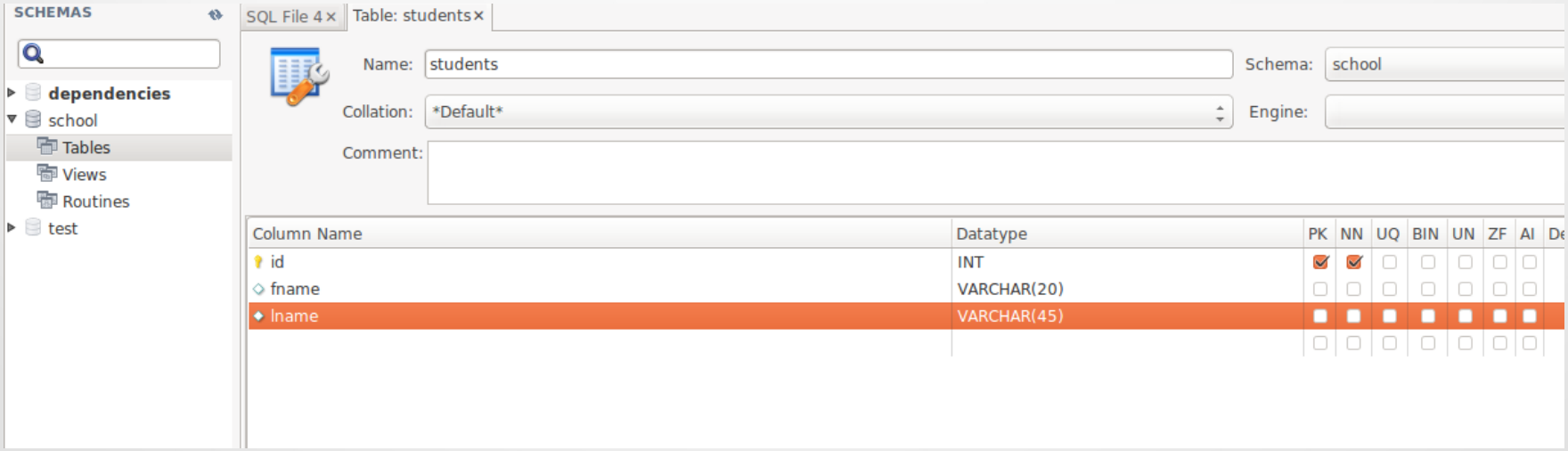
- Μόλις συνδεθείτε θα δείτε στα αριστερά σας τις υπάρχουσες βάσεις δεδομένων. Κάνοντας δεξί κλικ πάνω στο όνομα μιας υπάρχουσας ΒΔ θα εμφανισθεί ένα αναδυόμενο μενού.
- Από το αναδυόμενο μενού μπορείτε να επιλέξετε 'Create Schema...' για να δημιουργήσετε μία νέα ΒΔ.
- Δημιουργείστε μία ΒΔ με το όνομα 'school' και την κωδικοποίηση χαρακτήρων utf8 (όπως φαίνεται στην πάνω εικόνα).
- Μόλις πατήσετε το πλήκτρο 'Apply' θα εμφανισθεί το πλαίσιο διαλόγου που φαίνεται στην κάτω εικόνα και εφαρμόζει τις αλλαγές δημιουργώντας τη ΒΔ.





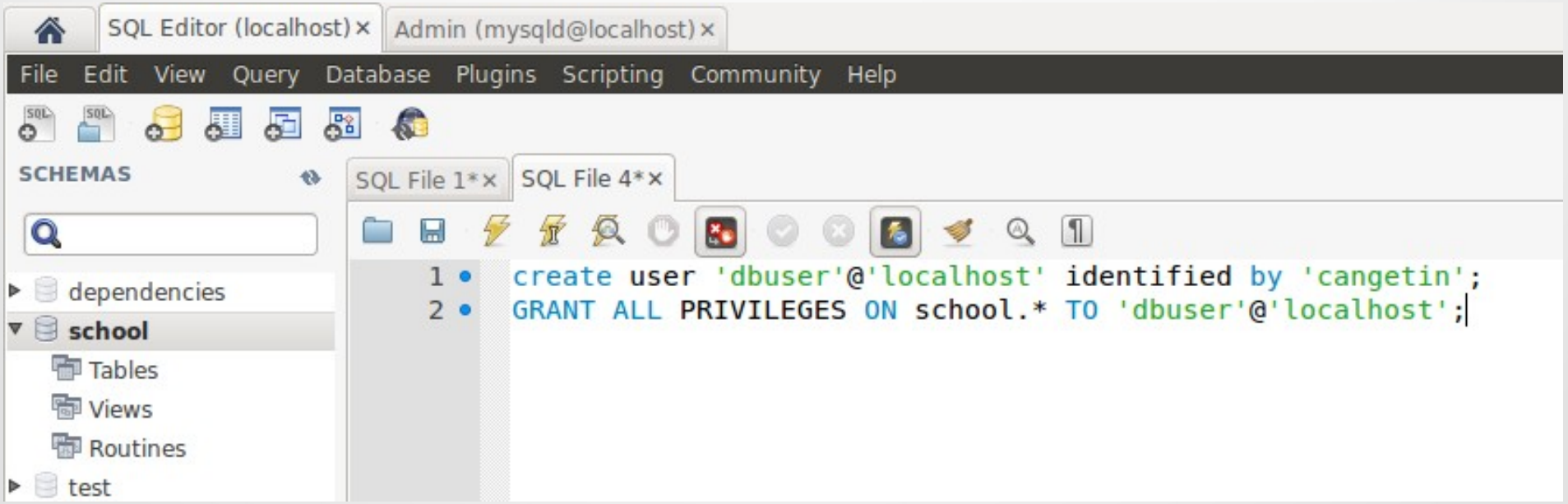
Δημιουργία της Βάσης Δεδομένων
Βήμα 3: δημιουργία πίνακα

- Θα δημιουργήσουμε έναν πίνακα με το όνομα students και τρία πεδία (id, fname, lname) για τον κωδικό, το όνομα και το επώνυμο αντίστοιχα.
- Κάνετε κλικ στο βελάκι δίπλα από το όνομα της ΒΔ school για να εμφανισθούν οι πίνακες (Tables). Προς το παρόν δεν υπάρχει κανείς.
- Κάνετε δεξί κλικ πάνω στο 'Tables' και επιλέξτε 'Create Table...'
- Δώστε τα στοιχεία που φαίνονται στην Εικόνα και πατήστε 'Apply' για να δημιουργήσετε τον πίνακα.



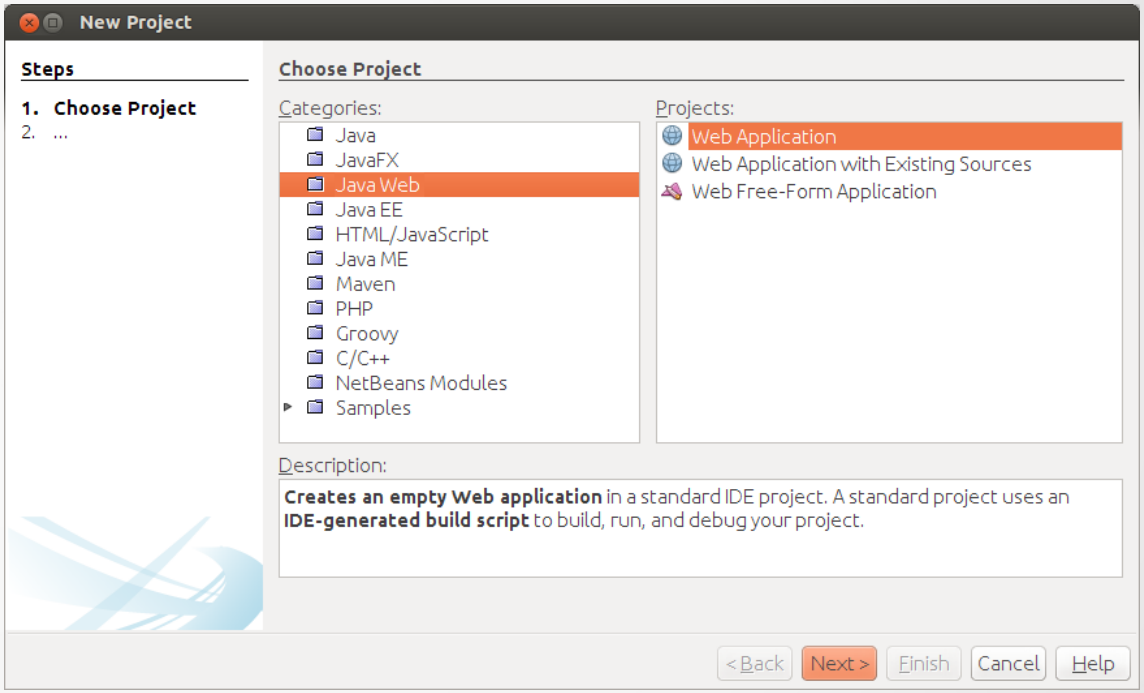
Δημιουργία της Βάσης Δεδομένων
Βήμα 4: δημιουργία χρήστη ΒΔ

- Προκειμένου να προσπελάσουμε την ΒΔ από μία εφαρμογή Java EE θα χρειαστεί να χρησιμοποιήσουμε έναν λογαριασμό της MySQL με δικαιώματα σ' αυτή τη ΒΔ. Τυπικά δεν χρησιμοποιούμε ποτέ τον λογαριασμό 'root' γι' αυτό για λόγους ασφαλείας.
- Στο MySQL Workbench πατήστε τον συνδυασμό πλήκτρων Ctrl+T ή επιλέξτε από το μενού την εντολή 'File-->New Query Tab'
- Στη νέα καρτέλα που θα δημιουργηθεί δώστε τις ακόλουθες εντολές SQL που φαίνονται στην Εικόνα και επιλέξτε το μενού 'Query --> Execute (All or Selection)' για να εκτελεσθούν οι εντολές και να δημιουργηθεί ο χρήστης 'dbuser' με κωδικό 'cangetin' ο οποίος έχει όλα τα δικαιώματα στη ΒΔ 'school' όταν συνδέεται από τον τοπικό υπολογιστή (localhost).



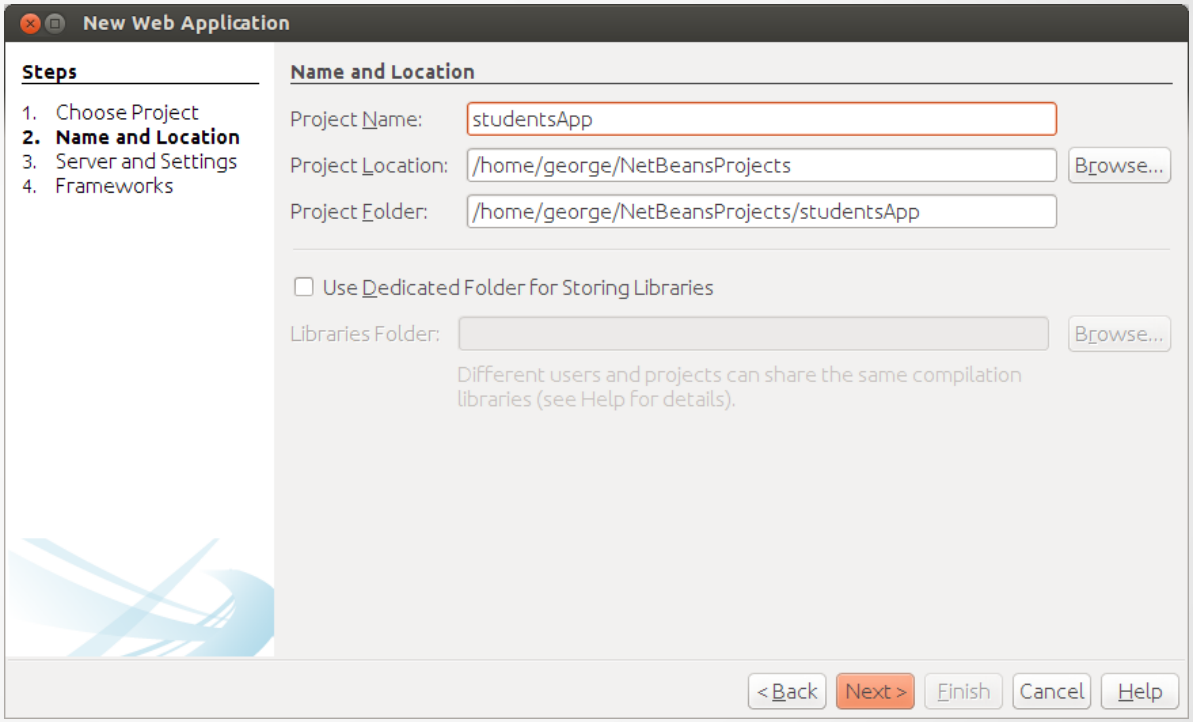
Δημιουργία της Εφαρμογής Διαδικτύου
Βήμα 1: Επιλογή κατηγορίας έργου

- Για να δημιουργήσουμε την εφαρμογή διαδικτύου 'studentsApp' ανοίγουμε το NetBeans και επιλέγουμε File-->New Project.
- Από το πλαίσιο διαλόγου 'New Project' επιλέγουμε στα 'Categories' την κατηγορία έργου 'Java Web' και από τα 'Projects' το 'Web Application' (όπως δείχνει η εικόνα) και πατάμε το πλήκτρο 'Next>'
- Προσέξτε πως για μία πλήρη εφαρμογή Java Enterprise Edition θα έπρεπε να επιλέξουμε τη κατηγορία έργου Java EE και όχι Java Web.



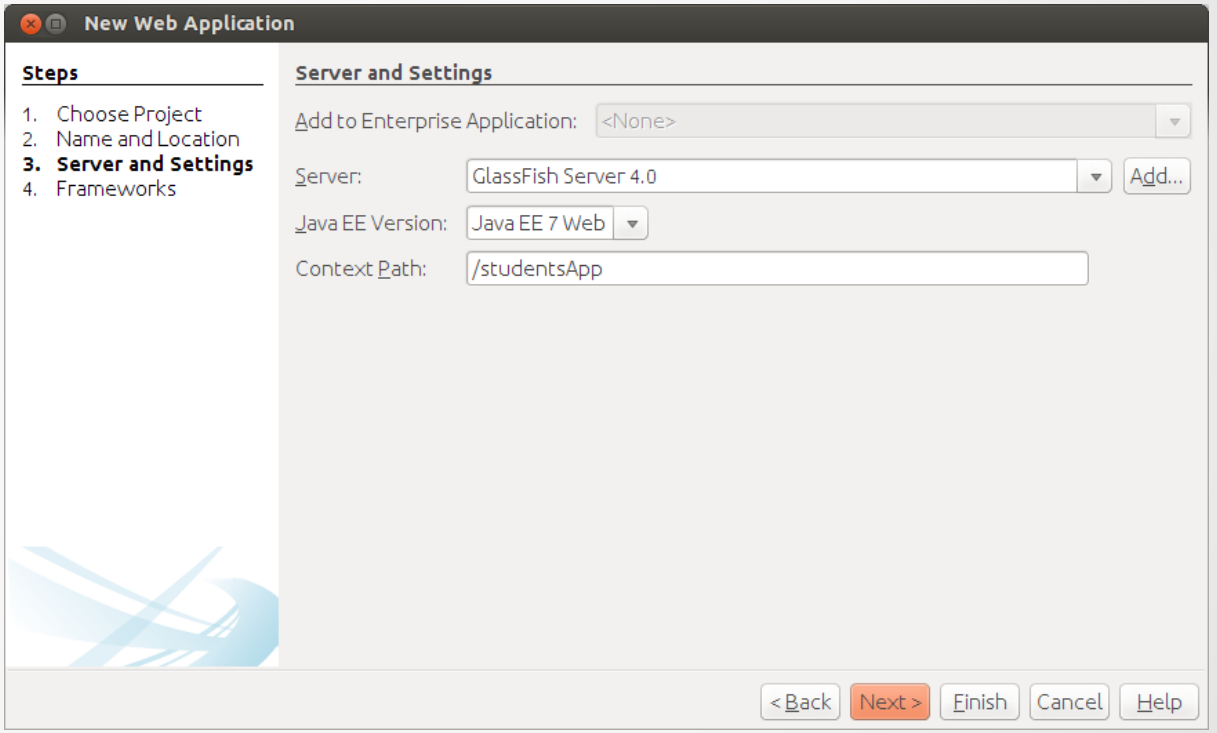
Δημιουργία της Εφαρμογής Διαδικτύου
Βήμα 2: Ονομασία της εφαρμογής

- Στο δεύτερο βήμα του οδηγού αποδίδουμε το όνομα 'studentsApp' στην εφαρμογή.
- Προσέξτε πως για την εφαρμογή θα δημιουργηθεί ένας κατάλογος με το ίδιο όνομα στην θέση 'Project Location' που είναι η ίδια για όλα τα έργα που κάνετε με το NetBeans.
- Ο φάκελος της εφαρμογής είναι το 'Project Folder'. Αν θέλετε να μεταφέρετε την εφαρμογή σε κάποιον άλλο υπολογιστή αυτόν το φάκελο αντιγράφετε.



Δημιουργία της Εφαρμογής Διαδικτύου
Βήμα 3: Εξυπηρετητής και Ρυθμίσεις

- Στο 3ο βήμα αφήνουμε τις εξ ορισμού τιμές για τα ακόλουθα:
- Εξυπηρετητής (Server): Glassfish Server
- Έκδοση Java EE: Java EE 7 Web
- Διαδρομή (context) εφαρμογής: /studentsApp. Αυτό σημαίνει πως η εφαρμογή αυτή θα είναι προσπελάσιμη όταν τρέχει σε έναν εξυπηρετητή με την διεύθυνση: http://<διεύθυνση εξυπηρετητή>/studentsApp π.χ. http://localhst/studentsApp
- Σε αυτό το βήμα πατάμε το 'Finish' για να ολοκληρώσουμε την δημιουργία της εφαρμογής,

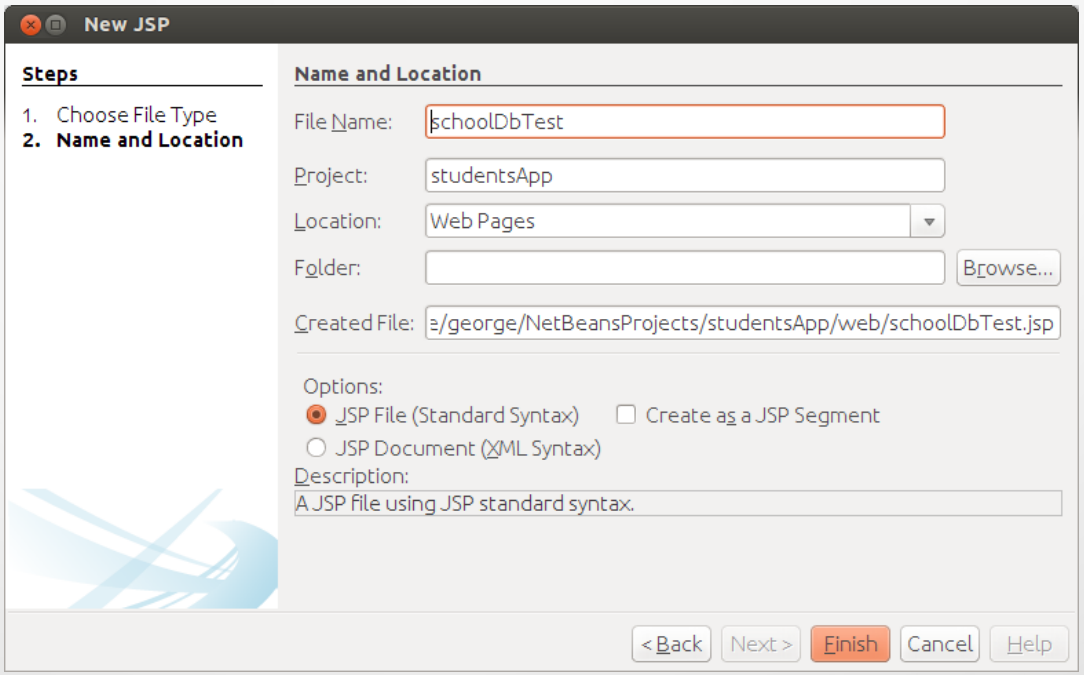


Δοκιμαστική εκτέλεση της εφαρμογής

- Μόλις ανοίξει η εφαρμογή στην καρτέλα 'Projects' θα δείτε το εικονίδιο της εφαρμογής ενώ στον editor θα έχει ανοίξει για επεξεργασία η αρχική σελίδα της εφαρμογής index.html.
- Αυτή η σελίδα αρχικά θα περιέχει το κείμενο 'TODO write content'
- Μπορείτε άμεσα να δοκιμάσετε την εκτέλεση της εφαρμογής χωρίς καμία αλλαγή.
  - Πατήστε το πλήκτρο 'F6' ή επιλέξτε το μενού 'Run--> Run Project (studentsApp)'.
  - Θα πρέπει μετά από λίγη ώρα αφού εγκατασταθεί η εφαρμογή, να ανοίξει ο browser και να δείτε την ιστοσελίδα index.html.
  - Προσέξτε πως η διεύθυνση της ιστοσελίδας είναι http://localhost:8080/studentsApp/ γιατί ο Glassfish εξυπηρετητής που εκτελείται στον τοπικό υπολογιστή χρησιμοποιεί τη θύρα 8080.

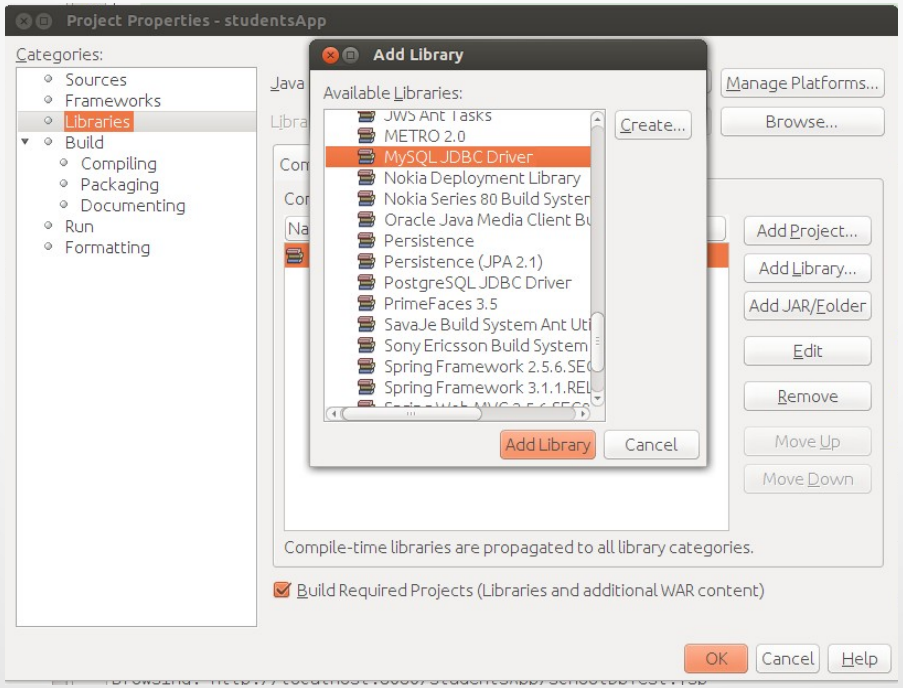
Δημιουργία Java Server Page (JSP)

- Τέλος θα δημιουργήσουμε μία JSP που θα εμφανίσει μία φόρμα για την εισαγωγή φοιτητών. Οι φοιτητές που εισάγουμε θα αποθηκεύονται στη ΒΔ αλλά θα εμφανίζονται και στη JSP σελίδα.
  - Δεν θα καταλάβετε τις λεπτομέρειες λειτουργίας της JSP. Ο σκοπός είναι να βεβαιώσουμε πως η επικοινωνία του εξυπηρετητή εφαρμογών (Glassfish) με τον εξυπηρετητή ΒΔ (MySQL) γίνεται κανονικά.
- Για να δημιουργήσετε τη JSP σελίδα κάνετε δεξί κλικ στο εικονίδιο της εφαρμογής 'studentsApp' και επιλέξτε 'New-->JSP...' από το αναδυόμενο μενού.
- Στο πλαίσιο διαλόγου δώστε το όνομα 'schoolDbTest' και πατήστε Finish για να δημιουργηθεί η JSP σελίδα.



Προσθήκη της βιβλιοθήκης του MySQL JDBC Driver

- Προκειμένου να συνδεθεί μία εφαρμογή Java με μία ΒΔ χρειάζεται μία ειδική βιβλιοθήκη που ονομάζεται JDBC Driver.
- Κάθε σύστημα ΒΔ έχει δικό του JDBC Driver.
- Για να προσθέσετε τη βιβλιοθήκη του MySQL JDBC Driver στην εφαρμογή κάνετε δεξί κλικ στο project και επιλέξτε 'Properties'.
- Στο πλαίσιο διαλόγου 'Project Properties' επιλέγετε 'Libraries' και στην συνέχεια πατάτε το πλήκτρο 'Add Library'. Από τις βιβλιοθήκες που θα εμφανισθούν επιλέγετε 'MySQL JDBC Driver' και πατάτε το πλήκτρο 'Add Library' και στην συνέχεια το 'OK' για να κλείσετε το πλαίσιο διαλόγου Properties.



Κώδικας της schoolDbTest.jsp

Ο κώδικας αυτός δεν είναι και ο καλύτερος μια και εμφανίζονται εντολές SQL, Java & HTML μαζί, η σύνδεση με τη ΒΔ γίνεται απευθείας χωρίς χρήση δεξαμενής συνδέσεων, τα στοιχεία του χρήστη της σύνδεσης με τη ΒΔ περιέχονται στον κώδικα κ.α. Αλλά είναι όλα σε ένα μόνο αρχείο και είναι βολικό για να διαπιστώσουμε πως όλα δουλεύουν καλά.

1

```
<%@page import="java.sql.*"%>
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
      <title>ΒΔ Σχολείου</title>
   </head>
   <body>
      <%
         String id = request.getParameter("id");
         String fname = request.getParameter("fname");
         String lname = request.getParameter("lname");
         try {
            Class.forName("com.mysql.jdbc.Driver");
            Connection con = DriverManager.getConnection(
              "jdbc:mysql://localhost:3306/school",
                  "dbuser", "cangetin");
            if (id!=null) {
               Statement s1 = con.createStatement();
               int result = s1.executeUpdate(
                  "insert into students values("+id+", '"+
                  fname+"','"+lname+"')");
               if (result==1) {
                  out.println("Επιτυχής Εισαγωγή: "+id+", "+fname+", "+lname+"<br />");
               }
            }
            Statement s2 = con.createStatement();
            ResultSet r = s2.executeQuery("Select * from students");
      %>
      <table border="1">
         <tr>
            <td>Κωδικός</td><td>Όνομα</td><td>Επώνυμο</td>
         </tr>
```

2

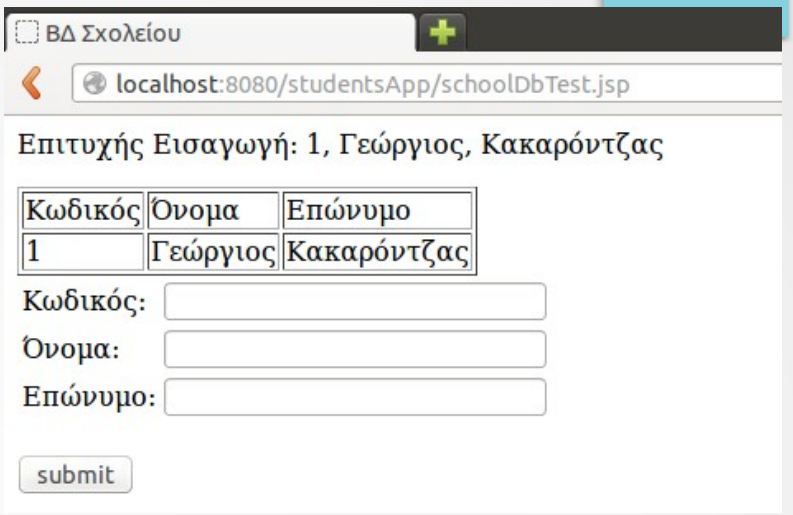
```
      <%
            while (r.next()) {
      %>
               <tr>
                  <td><%=r.getInt(1)%></td>
                  <td><%=r.getString(2)%>
                  </td><td><%=r.getString(3)%></td>
               </tr>
      <%
            }
            con.close();
         } catch (SQLException e) {
            out.println(e.getMessage());
         }
      %>
      </table>
      <p />
      <form method="post">
         <table>
            <tr>
               <td>Κωδικός: </td><td><input type="text" name="id" ></td>
            </tr>
            <tr>
               <td>Όνομα: </td><td><input type="text" name="fname"></td>
            </tr>
            <tr>
               <td>Επώνυμο:</td><td><input type="text" name="lname"></td>
            </tr>
         </table>
         <br>
         <input type="submit" value="submit">
      </form>
   </body>
</html>
```

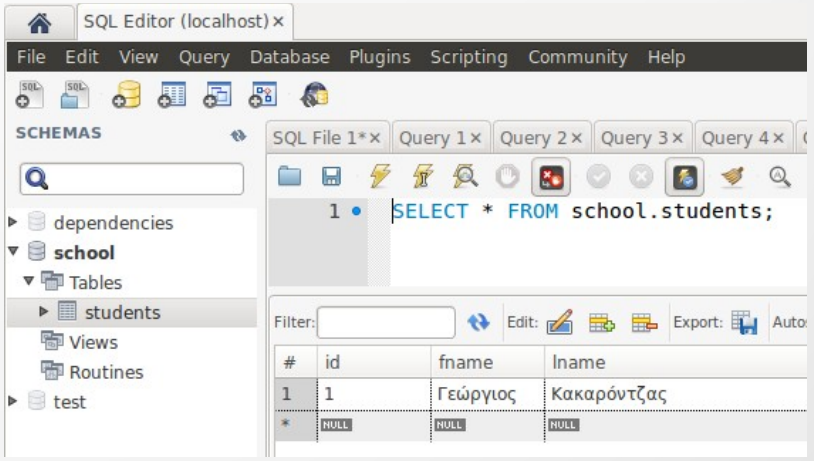
Ρύθμιση του Glassfish για UTF-8

- Προκειμένου να μπορεί ο εξυπηρετητής εφαρμογών Glassfish να χειριστεί Ελληνικά θα πρέπει να κάνουμε επίσης το εξής:
  - Κάνετε δεξί κλικ στο έργο και επιλέξτε 'New-->Other', και από την κατηγορία GlassFish επιλέξτε Glassfish Descriptor.
  - Ο επεξεργαστής του Glassfish Descriptor θα εμφανισθεί. Πηγαίνετε στη καρτέλα 'XML' και προσθέστε την γραμμή
    <parameter-encoding default-charset="UTF-8" />
    στο σημείο που φαίνεται στην εικόνα.

Εκτέλεση εφαρμογής

- Για να εκτελέσετε το αρχείο 'SchoolDbTest.jsp' με ανοιχτή αυτή τη σελίδα πατήστε Shift+F6 ή πατήστε δεξί κλικ πάνω στη σελίδα στην καρτέλα projects και από το αναδυόμενο μενού επιλέξτε 'Run File'
- Μόλις εκτελείται η σελίδα θα εμφανισθεί ένας άδειος πίνακας και μία φόρμα στην οποία μπορείτε να εισάγετε έναν νέο φοιτητή.
- Μόλις πατάτε το πλήκτρο 'submit' θα εισάγεται ο φοιτητής στη ΒΔ και θα βλέπετε ξανά την ίδια σελίδα όπου θα μπορείτε να εισάγετε και τον επόμενο φοιτητή κλπ.





Ερωτήσεις;