UNIVERSITÀ
DEGLI STUDI DI MILANO-BICOCCA

COURSE SYLLABUS

Software Engineering

2526-3-E3101Q119

Obiettivi

Acquisire conoscenze più avanzate di sviluppo del software rispetto a quelle acquisite durante il corso del II anno di analisi e progettazione del software. Conoscere ed applicare i pattern architetturali durante lo sviluppo del software. Identificare e rimuovere violazioni del codice attraverso l'utilizzo ed il supporto fornito da alcuni tool come SonarQube e SonarCloud . Introduzione a DevOps e alla Continuous Integration.
l percorso formativo include esercitazioni e attività di laboratorio.

Conoscenza e capacità di comprensione
Al termine del corso, lo studente avrà acquisito solida conoscenza e capacità nello sviluppo di progetti software complessi, utilizzo di pattern architetturali e sulla valutazione della qualità del codice.

Conoscenza e capacità di comprensione applicata
Lo studente sarà in grado di applicare tutto quello che ha appreso nel corso del II anno di Analisi e Progettazione del Software ed in quello di Ingegneria del Software del III anno per l'analisi e sviluppo di un progetto complesso, nell'applicazione di design pattern e pattern architetturali, nella valutazione della qualità del software prodotto attraverso metriche e riconoscimento di anomalie e problemi nel software e la lor risoluzione. Ogni studente dovrà conoscere ed utilizzare strumenti /tools molto noti nelle aziende per la valutazione della qualità del codice.

Autonomia di giudizio
Il corso promuove lo sviluppo del pensiero critico attraverso la valutazione comparativa di soluzioni progettuali e metodologiche, attraverso il confronto di strumenti e attraverso il contributo individuale nel progetto d'esame. Gli studenti avranno anche la possibilità di seguire uno o due seminari di aziende su tematiche di grande interesse per l'ingegneria del software, che stimoleranno ulteriormente le capacità di giudizio.

Abilità comunicative
Lo studente sarà in grado di comunicare efficacemente attraverso una presentazione individuale di un pattern architetturale in aula, lo sviluppo del progetto d'esame, l'elaborato prodotto sul progetto d'esame e la prova individuale orale.

Capacità di apprendere
Il corso promuove lo sviluppo dell'autonomia nello studio attraverso un approccio didattico attivo. Al termine del percorso, lo studente sarà in grado di approfondire in modo indipendente tematiche avanzate di ingegneria del software, che possono sfociare anche in uno stage/tesi, consolidando le basi per un apprendimento continuo anche oltre il contesto curriculare.

Contenuti sintetici

Principi, tecniche e strumenti per lo sviluppo del software. Pattern architetturali ed esempi della loro applicazione nello sviluppo del software. Best practices in Java. Valutazione della qualità del codice attraverso SonarQube e altri strumenti. Esempi di progetti software complessi e valutazione della loro qualità e delle principali problematiche che possono emergere.

Programma esteso

1 Presentazione del corso. Obiettivi e contenuti.
Ingegneria del software: introduzione alla model-driven software engineering, component-based software engineering, microservice-oriented software engineering, distributed software engineering.
2 Applicazione dei design pattern nello sviluppo del software. Modellazione di un framework di persistenza con i design pattern.
3 Architetture software. Progettazione dell'architetture software. Pattern architetturali per le applicazioni enterprise. Diagramma dei package, dei componenti e di deployment.
4 Best practices in Java. Reflessivitá in Java. (Lab).
6 Service-oriented software engineering: concetti fondamentali. Migrazione verso microservizi.
7 Valutazione della qualita' del software. Metriche per la valutazione del software. Design Principles di Martin.
8. Identificazione e rimozione di violazioni nel codice attraverso SonarQube (Lab).
9. Utilizzo di Git e GitHub actions, cooperazione in team durante lo sviluppo di un progetto (Lab).
10. Gestione dei progetti software: concetti base. Pianificazione dei progetti: diagrammi Gantt. Gestione dei rischi, gestione della qualitá.
11. Utilizzo di SonarCloud (Lab)
12. Utilizzo del tool Understand.
14. Introduzione al DevOps e al Continuous Integration: utilizzo di Github Actions

Prerequisiti

Analisi e Progettazione orientata agli oggetti.

Programmazione in Java.

Modalità didattica

Le lezioni si dovrebbero svolgere tutte in presenza e sono erogate in italiano.

Lezioni ed esercitazioni in aula. Il materiale didattico sarà disponibile nella piattaforma moodle. Attività ed esercizi di sviluppo software in laboratorio. Uso di Git, GitHub, Maven, JUnit, SonarQube, Understand e Github Actions in laboratorio.

Sono previsti seminari da parte di alcuni esperti del settore e di esponenti di aziende.
Possibilità di stage interni su tematiche dell'insegnamento o stage in aziende che intervengono durante il corso.

6 lezioni di 2 ore in presenza in modalità erogativa.
10 lezioni di 2 ore in presenza in modalità erogativa ed interattiva
10 attività di esercitazione di 2 ore in presenza in modalità erogativa ed interattiva
8 attività di laboratorio di 3 ore in modalità erogativa ed interattiva

Materiale didattico

Sommerville, Ingegneria del Software, Pearson, 8° ed, 2007.

C. Larman, Applicare UML e i Pattern – analisi e progettazione orientata agli oggetti, Pearson, 3° ed, 2005. (capitoli non visti nell'insegnamento del II anno di Analisi e Progettazone).

Slide, articoli , capitoli di libri e tutorial online di approfondimento sui vari argomenti del corso disponibili su moodle.

La maggior parte del materiale su cui dovrete studiare sarà disponibile on line.

Periodo di erogazione dell'insegnamento

I semestre

Modalità di verifica del profitto e valutazione

Progetto d'esame: La verifica dell'apprendimento comprende lo sviluppo di un sistema software in gruppo o anche individualmente ed un orale individuale con domande e discussione sul progetto e su alcuni argomenti del corso correlati al progetto d'esame assegnato. Il progetto d'esame prevede l'analisi, la progettazione e l'implementazione completa del sistema assegnato e l'utilizzo di SonarQube e di Understand per valutare la qualità del codice.

Valutazione nel range 0-22.

Valutazione dell'attività di laboratorio, molto importante in quanto gli studenti potranno imparare ad usare diversi strumenti e tool.
Valutazione nel range 0-4.

Task individule in aula. Presentazione di un pattern architetturale. Valutazione nel range 0-2

Orale. Valutazione nel range 0 - 5, si ricorda però che l'orale potrebbe anche abbasare il voto.

Orario di ricevimento

su appuntamento via email

Sustainable Development Goals