IDEA Progetto

Design Thinking ; Requisiti ; Kanban ; Pitch

D1

Implementazione

Git / GitHub
User Stories
RESTful API
OpenAPI
Web 2.0 JavaScript
WebAPI Node.js
MongoDB
Authentication JWT + GoogleAuth
Frontend
Deplyment & CI-CD
Testing Jest

D3

Analisi e Progettazione

Processi di sviluppo
Agile
Linguaggi di modellazione
Use Case Diagram
Sequence + Activity Diagram
Architetture
Component Diagram
Class Diagram
Class Diagram -> API
Testing

D2

Report finale

D4

Software Engineering

Testing: verifica e validazione

Testing: verifica e validazione

Il Testing

Verifica del codice implementato tramite una serie di **Casi di Test**

Due fasi del Testing:

- verifica del codice da parte del programmatore
- verifica del codice da parte di altre persone

Testing: verifica e validazione

Metodologie di Testing

- Analisi Statica
- Analisi Dinamica

Testing: verifica e validazione

Analisi Statica

Ricerca di eventuali anomalie analizzando il codice, anche usando tool appositi, ma non eseguendo il software

NOTA:

analisi ora non affrontata, sarà trattata in parte in un corso della Laurea Magistrale in Informatica denominato “Security testing”

Testing: verifica e validazione

Analisi Dinamica

Ricerca di eventuali malfunzionamenti del prodotto software tramite l'esecuzione del codice stesso fornendo opportuni dati in ingresso.

Testing: verifica e validazione

Logica invertita rispetto alla fasi di Analisi dei Requisiti, Progettazione ed Implementazione.

Prima dovevo pormi domande di questo tipo:

Com’è possibile implementare questa funzionalità?

Come posso indurre l’utente a compiere queste azioni?

Ora le domande diventano:

Cosa può generare un errore in questa funzionalità?

Cosa accade se l’utente compie queste azioni?

Testing: verifica e validazione

Indicatori chiave Analisi Dinamica

- Quantità dei test effettuati
- Copertura del codice da testare

Testing: verifica e validazione

Elementi chiave Analisi Dinamica

- Analisi Funzionale (Black Box)
- Analisi Strutturale (White Box)

Testing: verifica e validazione

Analisi Funzionale (Black Box)

considera la

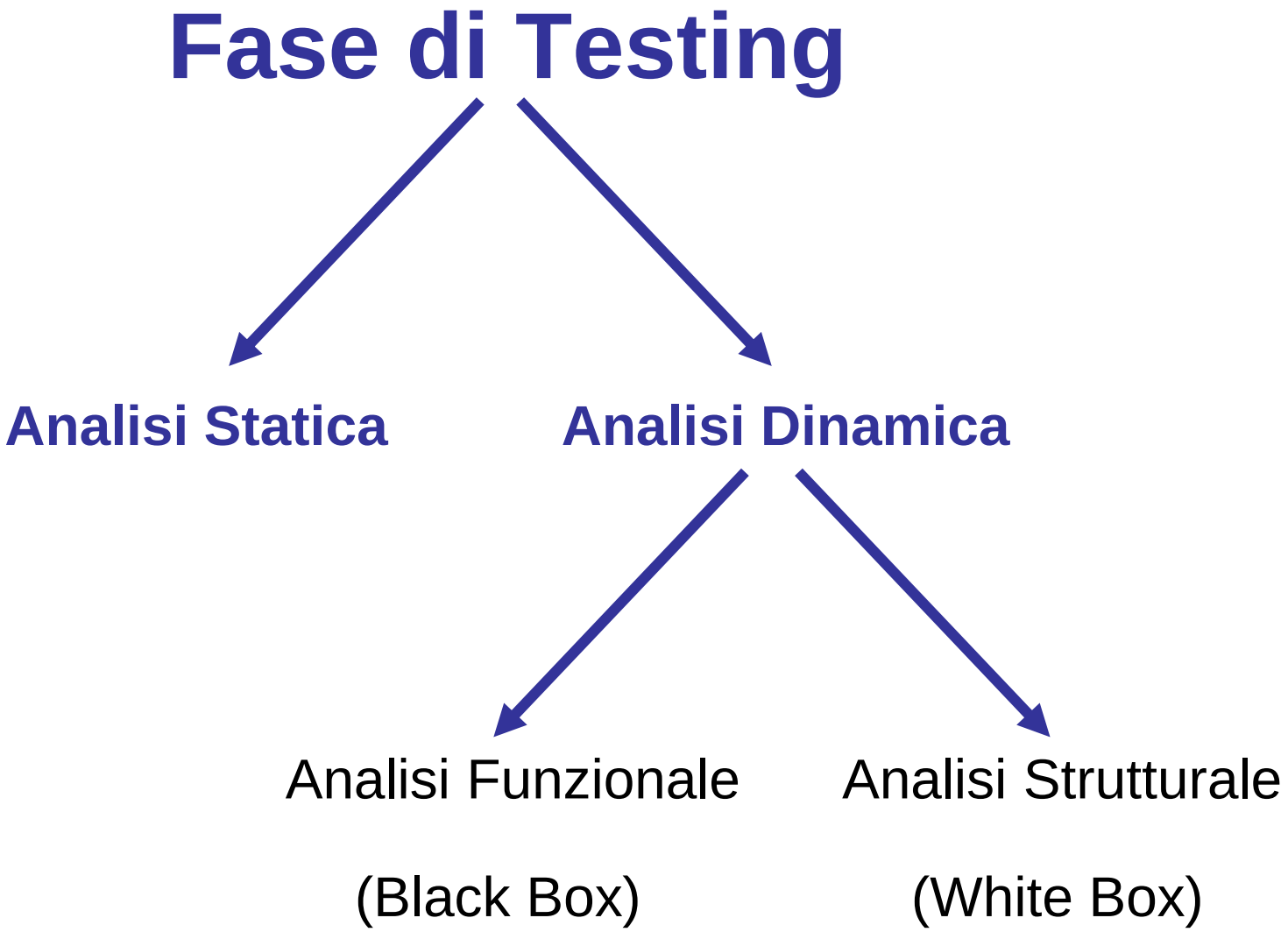
Quantità dei test effettuati

Analisi Strutturale (White Box)

considera la

Copertura del codice da testare

Testing: verifica e validazione



Testing: verifica e validazione

Analisi Funzionale (Black Box)

SCOPO:

sottoporre il software ad una serie di casi di test per verificare i requisiti richiesti (dato un requisito si possono avere 1..n casi di test)

CARATTERISTICHE:

- il codice non viene considerato
- verifica “sul campo” dei requisiti richiesti

Testing: verifica e validazione

Analisi Funzionale (Black Box)

DERIVABLE:

per ogni caso di test l'analisi deve riportare:

- descrizione caso di test
- dati di input utlizzati per il test
- eventuali precondizioni necessarie a questo test
- eventuali dipendenze con altri casi di test
- risultato atteso
- risultato riscontrato
- nel caso di risultato riscontrato diverso da quello atteso, descrizione dell'anomalie che ha prodotto il malfunzionamento.

Testing: verifica e validazione

Analisi Funzionale (Black Box)

RISULTATO:

positivo se risultato atteso sempre uguale a quello riscontrato per un alto numero di casi di test

Ma non basta:

è necessario garantire un adeguato set di casi di test (più casi di test per ogni requisito)

Testing: verifica e validazione

Analisi Strutturale (White Box)

SCOPO:

software sottoposto ad una serie di casi di test al fine di eseguire la maggior quantità possibile di righe di codice

CARATTERISTICHE:

- il codice è considerato
- garantisce la correttezza del software anche in casi estremi
- individua eventuali parti del codice non utilizzate

Testing: verifica e validazione

Analisi Strutturale (White Box)

DERIVABLE:

percentuale di copertura del codice utilizzando un’opportuna serie di casi di test. Questa percentuale sarà più alta per le parti del codice critiche. In ogni caso è molto difficile raggiungere il 100% di copertura.

Testing: verifica e validazione

Analisi Strutturale (White Box)

RISULTATO:

più è alta la percentuale di codice coperta più il risultato sarà positivo.

Testing: verifica e validazione

Definizione dei Casi di Test

Per ogni requisito si individuano alcuni casi di test.

In particolare:

- **almeno** un caso di test con il requisito rispettato
- **almeno** un caso di test con il requisito non rispettato
- **almeno** un caso di test in situazioni di frontiera

Ognuno di questi casi di test sarà caratterizzato da particolari dati di input del software da testare

Testing: verifica e validazione

Esempio di casi di Test associati ad un requisito

Requisito:

Il software da sviluppare deve assegnare un'account (nome utente a password) agli utenti che ne facciano richiesta.

Casi di Test:

- richiesta di un'account in modo corretto
- richiesta di un'account non specificando il nome utente
- richiesta di un'account inserendo un nome utente già presente
- richiesta di un'account specificando una password che non rispetta le politiche di sicurezza

Sono questi Casi di Test (quelli di frontiera) che fanno la differenza

Testing: verifica e validazione

Analisi strutturale e funzionale sono da farsi ovviamente dopo la fase di sviluppo del codice ma…

… la definizione dei Casi di Test viene fatta prima, in molti casi nella fase di Analisi dei Requisiti!

Caratteristiche caso di test

Un caso di test deve:

- avere un'alta probabilità di scoprire un errore
- non essere ridondante
- non deve essere né troppo semplice né troppo complesso

Testing: verifica e validazione

NOTE:

- non sarete voi ad eseguire questo caso di test
- chi legge il caso di test deve essere in grado di sapere cosa fare e sapere cosa aspettarsi, senza conoscere nel dettaglio l'applicazione
- chi legge il caso di test deve anche saper valutare se il risultato ottenuto è corretto o meno

Testing del vostro progetto

Fase di testing per il vostro progetto

Nel documento D3, per ogni requisito per cui avete sviluppato del codice, compilare una tabella analoga a quella presentata nella prossima slide.

Testing del vostro progetto

Esempio di tabella dei casi di test per RF 2: Registrare nuovo utente

Numero Test Case	Descrizione Test Case	Test Data	Precondizioni	Dipendenze	Risultato Atteso	Risultato riscontrato	Note:
1	Creazione di un'account in modo corretto	<username> non vuota <password> rispettosa delle politiche di sicurezza	<username> mai inserita prima nel sistema	---	Viene creata l'account specificato. Il sistema risponde con <messaggio_OK>		
1.1	Creazione di un'account specificando uno username già esistente	<username> già inserita nel sistema	<username> già inserita nel sistema	Questo caso di test deve essere fatto dopo il caso di testo numero 1 specificando la stessa <username>	Viene mostrato un messaggio di errore <messaggio errore_user_exists> , l'account non viene creata ed il sistema mostra alcuni username disponibili da utilizzare		
2	Creazione di un'account non specificando lo username	<username> Vuota	---	---	Viene mostrato <messaggio errore_emptyuser> e l'account non viene creato		
3	Creazione di account violando le politiche di sicurezza (password banale)	<password> non rispettosa delle politiche di sicurezza	---	---	Viene mostrato un messaggio di errore e l'account non viene creata		