Processo di sviluppo, strumenti e metodologie.

Marco Matarazzi

Chi sono

Marco Matarazzi
Sono un ingegnere informatico.

Sono un appassionato di Sviluppo Software

Mi piace lavorare insieme a persone in gamba (smart) che come me hanno il chiodo fisso di realizzare software all'avanguardia usato da milioni di utenti.

Di cosa parleremo?

- ★ Che cosa facciamo a Vendini
- ★ Le figure professionali
- ★ Il workflow in un team distribuito
- ★ Cicli di rilascio del codice

Due parole su chi siamo e cosa facciamo

- ★ Siamo un team di 20 ingegneri informatici tutti molto appassionati di sviluppo software. La nostra sede è a Gualdo Tadino (Perugia).

- ★ Lavoriamo per un'azienda di San Francisco. Sviluppiamo un SaaS (web based) per il Ticketing.

La nostra architettura software

Legacy web apps

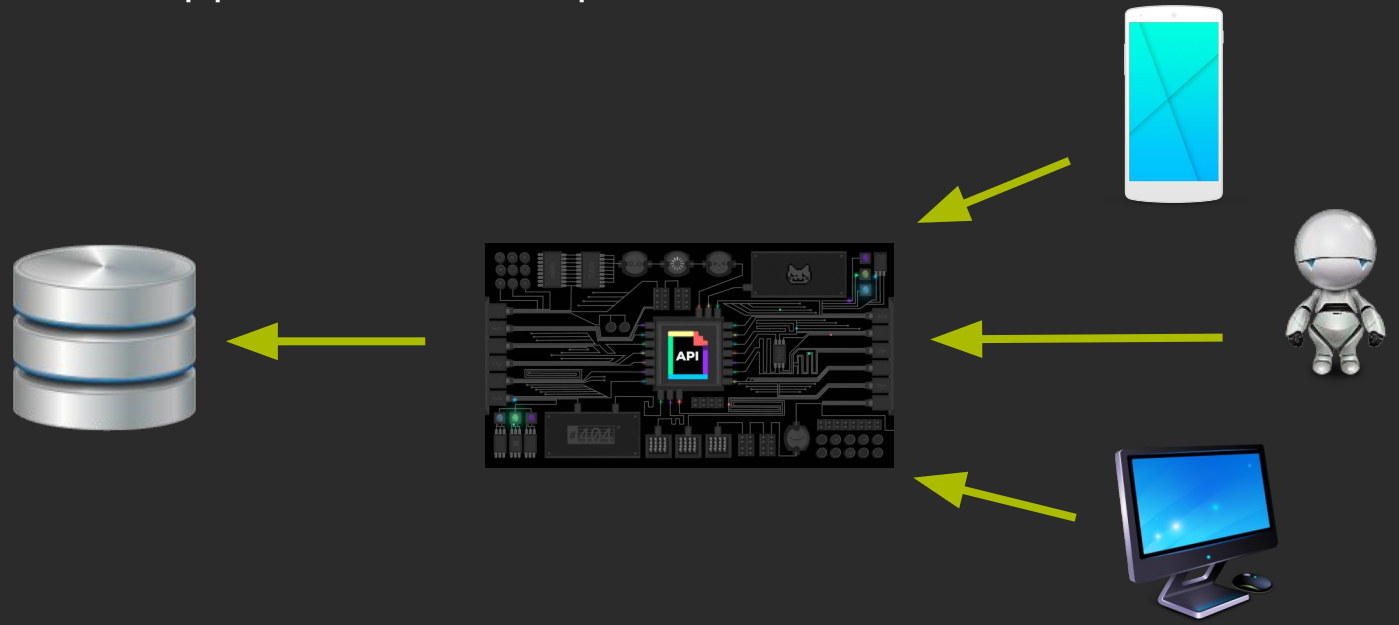
- ★ Logica di business potenzialmente ripetuta in più punti
- ★ Logica di business spesso mischiata con logica applicativa, di visualizzazione o di raccolta dati dall'utente
- ★ Ogni applicazione necessita di più o meno codice ad-hoc per accedere e manipolare dati

API

- ★ Logica di business centralizzata in un solo punto
- ★ Logica di business separata da logica applicativa, di visualizzazione o di raccolta dati dall'utente
- ★ Dati offerti dagli stessi endpoint (URL di servizio) possono servire N applicazioni diverse

Cosa vogliamo fare con le Vendini API?

Esporre le funzionalità della piattaforma Vendini sotto forma di servizi HTTP, agnostici rispetto all'utilizzatore e consumabili da qualsiasi tipo di client (app mobile/desktop, bot, ecc...)



VENDINI

Applicazioni web

- Le nostre applicazioni web di nuova generazione sono basate su Angular JS

- Framework molto popolare (made in Google) per scrivere applicazioni anche molto strutturate, in Javascript, che girano direttamente nel browser

https://angular.io

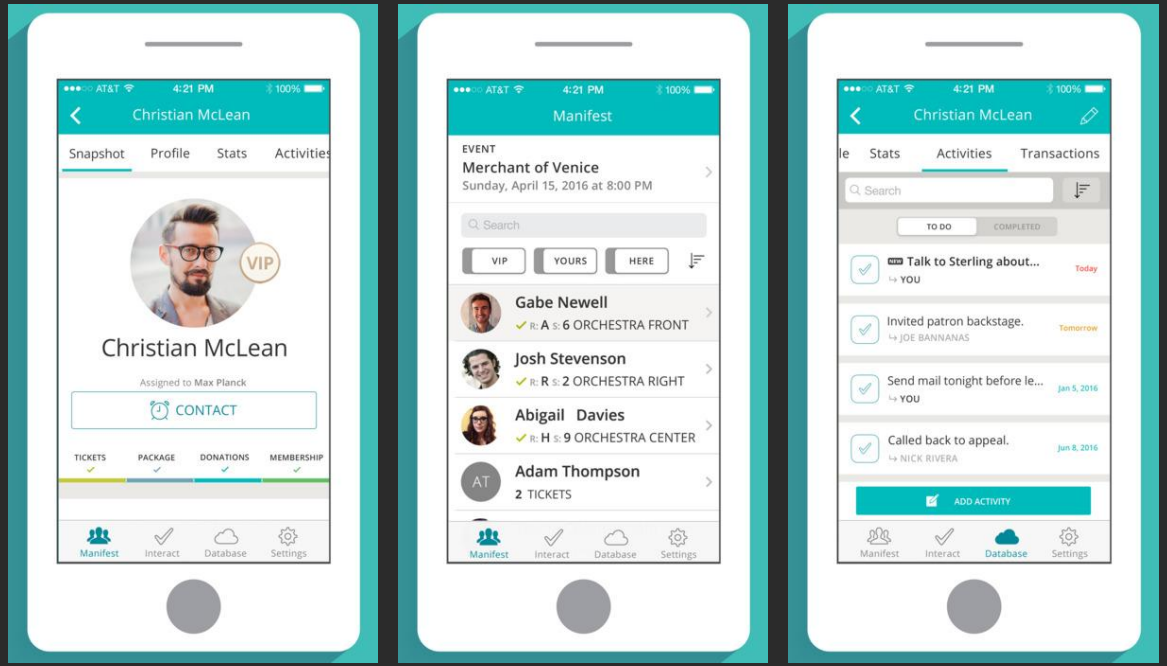
Cenni sulle applicazioni mobile

- Le nostre applicazioni mobile sono basate su ionic

- Framework che si appoggia sopra Angular
- Integra *Apache Cordova* (ex PhoneGap)
- Serve a sviluppare applicazioni mobili *ibride*: hanno caratteristiche native (accesso a fotocamera, sensori, ecc...) ma sono scritte in JavaScript (invece di Java, Objective-C o Swift)
- https://ionicframework.com/

Un esempio di applicazione - CRM Patron Connect

CRM mobile (iOS e Android) per la gestione dei rapporti con gli utenti finali (chi compra biglietti)



VENDINI

Le figure professionali

VENDINI

Le figure professionali tech a Vendini oggi (1 di 5)

"Backend developer"

Sviluppatori PHP principalmente concentrati sullo sviluppo di API.

Le figure professionali tech a Vendini oggi (2 di 5)

"Frontend developer"

Sviluppa applicazioni JavaScript (AngularJS, React) oppure app native (iOS, Android).

Le figure professionali tech a Vendini oggi (3 di 5)

"Frontend UI/UX"

Crea interfacce grafiche e le traduce in HTML e CSS

Le figure professionali tech a Vendini oggi (4 di 5)

"Test Engineer"

Scrive test funzionali, tiene alta la qualità del codice tramite la review del codice degli sviluppatori.

Le figure professionali tech a Vendini oggi (5 di 5)

"Operations Engineer"

Si occupa della gestione, manutenzione e configurazione delle macchine server.

Workflow in un team distribuito

Workflow (panoramica di alto livello)

1. Product Team crea le specifiche
2. Il Product Team crea il backlog dei "casi"
3. Lo scrum team stima i casi
4. Si crea lo sprint
5. Si inizia il lavoro (coding)
6. Al termine dello sprint si crea la release candidate
7. Le funzionalità non implementate slittano nello sprint successivo.

1 - Gli scrum team

Gli scrum team sono composti da:
Product Manager, Ingegneri del Software e Test Engineers, in genere 5 persone per team.

- ★ Sviluppatori: 3 o 4
- ★ Test engineer: 1
- ★ Product Manager: 1

2 - Sprint ed Estimate

★ Uno sprint può durare 5 o 10 giorni lavorativi

- Al termine dello sprint si ha una (o più) unità di codice rilasciabile

★ "L'estimate" consiste nel quantificare l'impegno necessario per implementare una data unità lavorativa (in altre parole "il tempo necessario").

NOTA: Metodologia di assegnazione degli story point tramite la sequenza di "Fibonacci" (1 - 2 - 3 - 5 - 8 - 13 - 21)

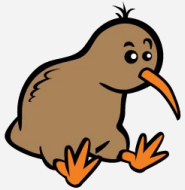
3 - Sistemi di "bug tracking"

Sono dei software che servono per:

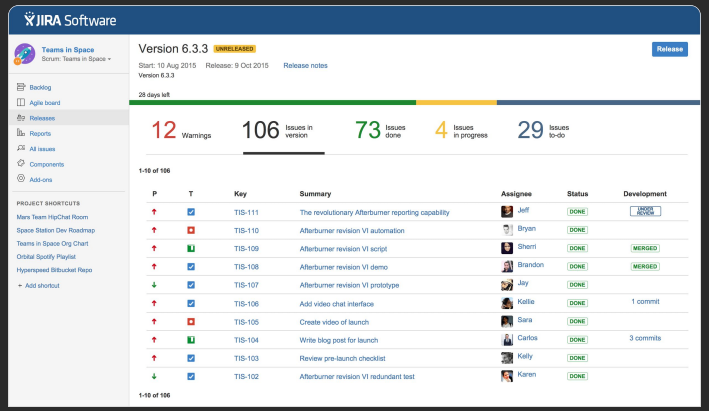
- ★ Tenere traccia delle funzionalità da implementare
- ★ Organizzare i task
- ★ Prioritizzare i bug
- ★ Gestire gli step di sviluppo del codice.

Esempi di sistemi di "bug tracking"

JIRA

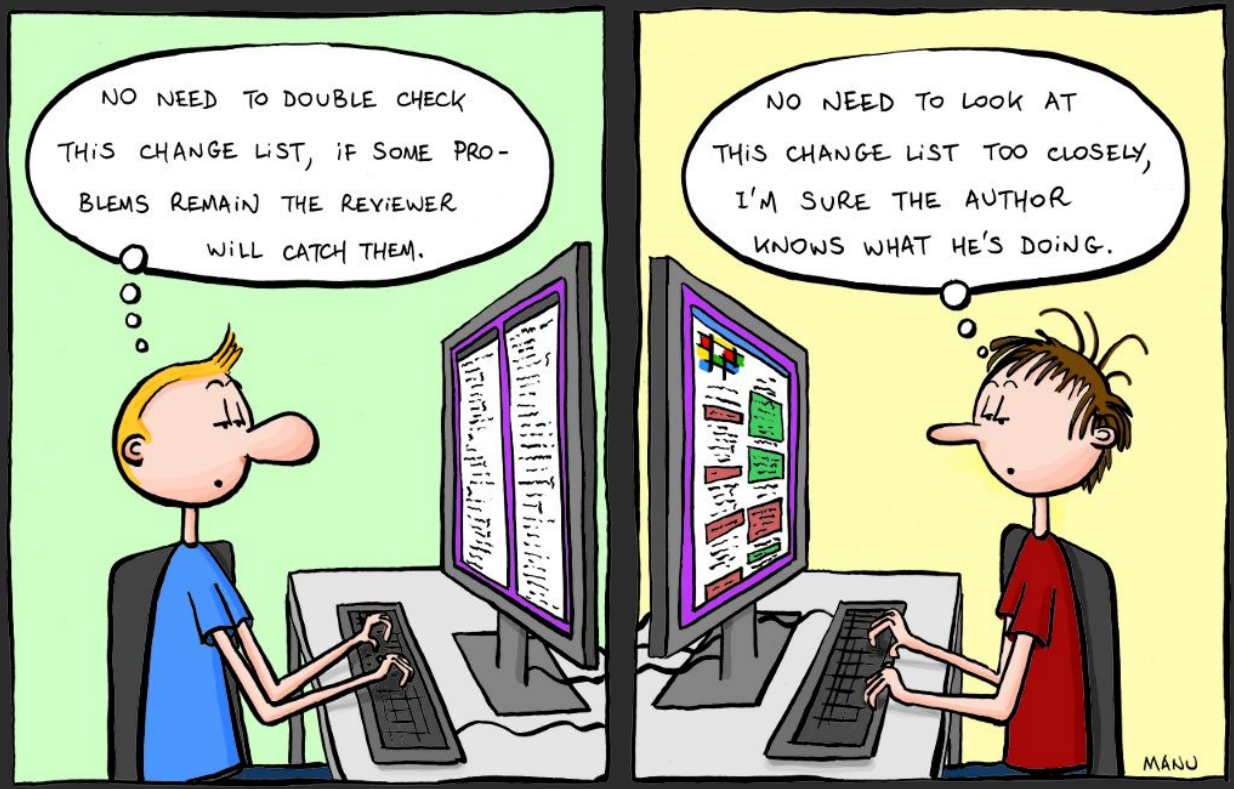


FogBugz



VENDINI

4 - Code review



4 - Code review

Ogni riga di codice che viene scritta deve essere rivista da un secondo sviluppatore

- Riduce la possibilità di introdurre falle di sicurezza
- Migliora la qualità del codice
- Riduce il numero di bug
- Porta lo sviluppatore a scrivere del codice "migliore"
- Migliora la conoscenza del software tra i membri del team

Il controllo di versione

Git è un software di controllo versione distribuito utilizzabile da interfaccia a riga di comando, creato da Linus Torvalds nel 2005.

- Permette di lavorare in team sulla medesima codebase.
- Permette di "versionare" il proprio lavoro anche se si è da soli.

L'importanza della scrittura di test automatici

Gli sviluppatori non solo possono ma DEVONO dedicare del tempo alla scrittura dei test automatici (unit, integration e funzionali).

- ★ Migliorano (di tantissimo) la robustezza dell'applicazione.
- ★ Permettono di rifattorizzare codice.
- ★ Velocizzano il processo di QA.
- ★ Servono da documentazione.
- ★ Sono super fighissimi e puoi farci il figo con il tuo amico che lavora da un'altra parte e dice di non aver tempo per scriverli.

Tool di "continuous integration" - TravisCI

- ★ Permettono di automatizzare l'esecuzione dei test automatici
  - Ad ogni push sulla propria feature branch.
  - Ad ogni merge sulla master branch.

- ★ Ti notificano se una modifica "rompe la build"

- ★ Ti permettono di essere agile perchè il codice che rilasci è meno propenso ad essere affetto da bug (ammesso che i test siano stati scritti bene)

Riassumiamo il ciclo di sviluppo software

1. Definizione delle specifiche e creazione dei casi
2. Sprint planning
3. Inizio dello sviluppo
4. Creazione della Release Candidate
5. Merge della release candidate su master
6. Aggiornamento dell'ambiente di verifica "staging"
7. Messa in produzione del codice (deploy)

Domande?

Bonus slide, due consigli:

- ★ Una buona lettera di presentazione colpisce più di un buon CV.
- ★ Create una lettera di presentazione ad hoc per ogni posizione (non mandate il CV a tutte le aziende "a caso".
- ★ Mettete nel vostro CV i vostri progetti personali e passioni.
- ★ Se ci tenete a lavorare per un'azienda, assicuratevi che il vostro CV sia arrivato alle persone "giuste" dell'azienda prima di lasciar perdere (insistete un pochino).