Università degli Studi di Padova

GlitchHub TEAM

# Glossario

Versione **0.6.0**

| | |
|---|---|
| **Stato** | Verificato |
| **Distribuzione** | Pubblica |

# Registro Modifiche

| Ver. | Data | Autore | Verificatore | Descrizione |
|---|---|---|---|---|
| 0.6.0 | 25/03/2026 | Alessandro Dinato | Michele Dioli | Definizione termini: «Apidog», «Endpoint», «Fire-and-forget»,«Dependency Injection», «Uber Fx» |
| 0.6.0 | 18/03/2026 | Alessandro Dinato | Jaume Bernardi | Definizione termini: «Dev Container», «GitHub Secret», «Golangci-lint», «Gofumpt»,«MKDocs»,«Deployment Diagram»,«Code Diagram»,«Component Diagram»,«Container Diagram»,«Context Diagram»,«C4 Model», «Manuale utente», «Specifica tecnica» |
| 0.5.1 | 03/03/2026 | Elia Ernesto Stellin | Jaume Bernardi | Rimozione numerazione per sezioni di «raggruppamento» dei termini |
| 0.5.0 | 18/02/2026 | Alessandro Dinato | Riccardo Graziani | Aggiornamento definizione Milestone e Baseline, rimozione termini ridondanti |
| 0.4.0 | 14/02/2026 | Alessandro Dinato | Riccardo Graziani | Aggiornamento dei termini Consuntivo di Periodo e Consuntivo a finire |
| 0.3.1 | 08/02/2026 | Elia Ernesto Stellin | Alessandro Dinato | Reinserimento dei collegamenti intra-documento e aggiunta di collegamenti vari |
| 0.3.0 | 07/02/2026 | Alessandro Dinato | Riccardo Graziani | Aggiunta termini tecnici incontrati nello sviluppo del PoC e inserimento introduzione al documento |
| 0.2.1 | 23/01/2026 | Elia Ernesto Stellin | Riccardo Graziani | Sistemate alcune definizioni per compatibilità con automazione |
| 0.2.0 | 15/01/2026 | Elia Ernesto Stellin | Riccardo Graziani | Aggiunti termini relativi a GitHub, termini relativi a UC / Attori, termini relativi a gateway e termini relativi a sensori; Impostati link intra-documento |

| Ver. | Data | Autore | Verificatore | Descrizione |
|---|---|---|---|---|
| 0.1.3 | 28/11/2025 | Alessandro Dinato | Riccardo Graziani | Correzione consuntivo di periodo e preventivo a finire |
| 0.1.2 | 28/11/2025 | Siria Salvalaio | Hossam Ezzemouri | Aggiunta definizione |
| 0.1.1 | 4/11/2025 | Michele Dioli | Alessandro Dinato | Sistemate alcune definizioni |
| 0.1.0 | 3/11/2025 | Alessandro Dinato | Michele Dioli | Definizione termini specificati a lezione |
| 0.0.2 | 1/11/2025 | Alessandro Dinato | Michele Dioli | Definizione termini tecnici glossario |
| 0.0.1 | 31/10/2025 | Alessandro Dinato | Michele Dioli | Creazione glossario |

# Indice

# 1. Introduzione

Il presente documento ha lo scopo di fornire una raccolta di termini e definizioni utilizzati nell'ambito del **progetto didattico**.

Il glossario ha lo scopo di raccogliere termini teorici e tecnici nell'ambito SWE associandone una definizione chiara e concisa adottata da tutto il gruppo.

Il documento è destinato ad essere un riferimento per tutti i membri del gruppo, fissando concetti chiave e terminologia comune, al fine di facilitare la comunicazione e la comprensione all'interno del progetto.

# A

### Account - NATS

Entità in **NATS** che rappresenta un'unità di isolamento (tenant), con propri utenti, permessi e stream. Gli Account possono comunicare tra loro tramite permessi specifici.

### ACK

Abbreviazione di *"Acknowledgment"*, è un messaggio di conferma inviato da un destinatario per indicare che ha ricevuto e processato correttamente un messaggio o una richiesta.

### Agile

Insieme di metodologie di sviluppo software basate su iterazioni brevi, collaborazione costante e adattamento continuo ai cambiamenti.

### Analisi dei Requisiti

È l'attività che studia i bisogni dell'utente e del dominio d'uso per definire che cosa il prodotto software deve fare per soddisfarli, senza descrivere come sarà realizzato. Produce una specifica *completa*, *verificabile* e *tracciabile* dei requisiti del sistema (lato soluzione).

### Angular

Framework front-end sviluppato da Google per creare applicazioni web complesse e modulari, è un framework opinionated.

### Apache Kafka

Piattaforma distribuita per lo streaming di dati progettata per elaborare flussi di eventi in tempo reale, basata su un modello publish-subscribe, con tolleranza ai guasti, scalabilità orizzontale e persistenza dei messaggi su disco.

### API

L'*Application Programming Interface* è un'interfaccia che permette a due sistemi software di comunicare tramite richieste e risposte strutturate.

### API Client

Un qualunque client che possa accedere ai dati esposti dall'**API** pubblica del sistema **cloud**.

### Apidog
Strumento di documentazione API che consente di generare documentazione interattiva e facilmente navigabile per le API di un sistema, supportando l'importazione di specifiche OpenAPI e la personalizzazione del layout e dello stile della documentazione.

### Architettura serverless
Modello di progettazione **Cloud** in cui l'infrastruttura sottostante è completamente gestita dal provider. Permette esecuzione di funzioni on-demand, scalabilità automatica e fatturazione *pay-as-you-go*.

### Architettura/Design
Descrive come il sistema è organizzato internamente: componenti, interazioni e tecnologia scelta. Indica come verranno soddisfatti i requisiti software. Ne è responsabile il *progettista*

### Attore
Nel contesto di un **caso d'uso**, è un'entità esterna al **sistema** preso in considerazione che interagisce con lo stato di quest'ultima in lettura e/o scrittura. Può corrispondere a un essere umano o a un agente automatizzato che compie un'azione specifica.

## B

### Back-end
È la parte di un software che gestisce la logica applicativa, l'elaborazione dei dati e la comunicazione con database o servizi esterni. Inoltre gestisce l'autenticazione e l'autorizzazione dell'utente.

### Baseline
La baseline è un insieme di documenti e artefatti, con una versione specifica, volti a fornire un'evidenza concreta del raggiungimento dello stato di avanzamento desiderato e stabilito nella **Milestone**.

### Bluetooth Low Energy (BLE)
Versione a basso consumo del Bluetooth, progettata per sensori e dispositivi **IoT** che richiedono comunicazioni energeticamente efficienti.

## C

### C4 Model
Modello di visualizzazione dell'architettura software che utilizza quattro livelli di astrazione: *Context*, *Container*, *Component* e *Code*. Ogni livello rappresenta una diversa prospettiva del sistema, facilitando la comprensione e la comunicazione dell'architettura.

### Capitolato d'appalto
Documento redatto dalla **proponente** con lo scopo di esporre il prodotto richiesto. Esso contiene vincoli, suggerimenti e aspettative (**requisiti utente**)

## Caso d'uso
La descrizione di una o più funzionalità del prodotto dal lato degli *user needs*, ovvero delle necessità dell'utente, tramite il linguaggio visivo UML e un'aggiuntiva descrizione testuale. I casi d'uso sono descritti nell'**Analisi dei requisiti**.

## CD
Insieme di pratiche di automazione del rilascio software. Vd. **Continuous Delivery** e **Continuous Deployment**

## CI
Acronimo di *Continuous Integration*. È una pratica di sviluppo software in cui le modifiche al codice vengono integrate frequentemente nel repository principale, eseguendo automaticamente build e test per individuare problemi precocemente.

## Ciclo di vita
Insieme delle fasi attraversate da un prodotto software: *concezione*, *sviluppo*, *utilizzo* e *ritiro*.

## ClickUp
Strumento di project management che consente di organizzare task, documenti, obiettivi e comunicazione del team.

## Cloud
Modello che permette di utilizzare risorse informatiche (server, database, servizi) tramite Internet senza gestire l'infrastruttura fisica.

## Code coverage
Percentuale di codice coperta dai test automatici, indica quanto è testata l'applicazione.

## Code Diagram
Quarto livello del **C4 Model** che rappresenta il codice sorgente a livello di classi, moduli o funzioni, evidenziando le relazioni tra di essi.

## Comando (gateway)
Messaggio inviato a un **Gateway** per modificarne lo stato o la configurazione attuale.

## Commissioning (gateway)
Procedura di associazione di un **gateway** a un **tenant** specifico e di impostazione di una **configurazione** precisa.

## Committente
Nel *progetto didattico* lo sono i docenti ed hanno lo scopo di: misurare il progresso con due revisioni, regolare il progetto didattico e valutarlo

## Component Diagram
Terzo livello del **C4 Model** che dettaglia i componenti all'interno di un container specifico, mostrando le loro responsabilità e interazioni.

### Configurazione (gateway)
Insieme di parametri associati a un **gateway** che ne dettano le specifiche di funzionamento.

### Configurazione di fabbrica (gateway)
**Configurazione** di un **gateway** di default.

### Consuntivo di periodo
È una sezione nel **Piano di Progetto**, che viene compilata ogni **sprint**, nel quale vengono analizzate le ore effettivamente consumate nello **sprint** interessato e di conseguenza le risorse utilizzate.

### Container Diagram
Secondo livello del **C4 Model** che mostra i principali contenitori software (applicazioni, database, servizi) all'interno del sistema e le loro interazioni.

### Context Diagram
Primo livello del **C4 Model** che rappresenta il sistema come un unico blocco, mostrando le interazioni con gli attori esterni e i sistemi con cui comunica.

### Continuous Delivery
Pratica di rilascio software in cui il codice è sempre pronto per essere rilasciato in produzione, ma il deploy avviene manualmente.

### Continuous Deployment
Pratica di rilascio software in cui ogni modifica che passa i test automatici viene rilasciata automaticamente in produzione.

### Controllo di versione (VCS)
Sistema che registra la storia delle modifiche ai file, permettendo collaborazione, tracciamento e ripristino di versioni precedenti.

## D

### Daily Scrum
Breve meeting quotidiano (circa 15 minuti) in cui il team sincronizza le attività e identifica eventuali impedimenti.

### Debito tecnico
Conseguenza di scelte di sviluppo o di design che privilegiano la velocità iniziale a scapito della qualità, richiedendo iterazioni future per correggerle.

### Decommissioning (gateway)
Procedura di disassociazione di un **gateway** al **tenant** a cui è correntemente associato e di impostazione del gateway alla **configurazione di fabbrica**.

### Dependency Injection
Pattern di progettazione in cui le dipendenze di un componente vengono fornite dall'esterno, invece di essere create internamente, facilitando la modularità, la testabilità e la manutenibilità del codice.

### Deployment Diagram
Diagramma, considerato il quinto livello del **C4 Model**, che mostra la distribuzione fisica dei componenti software su nodi hardware o ambienti di esecuzione, evidenziando le comunicazioni tra di essi.

### Dev Container
Strumento che consente di definire un ambiente di sviluppo completo all'interno di un container Docker, facilitando la configurazione e la condivisione dell'ambiente di sviluppo tra i membri del team.

### Diagramma dei casi d'uso
Diagramma UML che descrive le interazioni tra *attori* e *sistema* evidenziando cosa può essere richiesto al software.

### Diagramma delle classi
Diagramma UML che mostra le classi del sistema, attributi, metodi e relazioni strutturali.

### Dichiarazione degli impegni
Documento redatto in fase di candidatura del gruppo presso un **capitolato d'appalto**. Contiene l'analisi dei ruoli e dei rischi, la divisione delle risorse e il preventivo costi.

### Discord
Piattaforma di comunicazione che permette chat testuali, vocali e video, utilizzata spesso per la collaborazione tra gruppi di lavoro.

### Docker
Piattaforma che permette di creare, distribuire e eseguire applicazioni all'interno di container isolati, garantendo portabilità tra diversi ambienti.

### Docker Compose
Strumento che permette di definire e gestire applicazioni multi-container **Docker** tramite un file YAML, semplificando l'orchestrazione e la configurazione dei servizi.

### Documento incrementale
Documento che viene redatto e aggiornato progressivamente insieme all'avanzamento del progetto. Può contenere inizialmente sezioni vuote e incomplete e ogni versione stabile include solo le parti effettivamente compilate e verificate. Viene pubblicato man mano tramite versioni aggiornate.

## E

### ECG Custom Profile (Profilo GATT)
**Profilo GATT** usato per l'invio via **BLE** di dati relativi a misurazioni di elettrocardiogrammi.

### Economicità
È la combinazione tra **efficacia** e **efficienza**, ovvero l'equilibrio tra costi sostenuti, risorse impiegate e risultati ottenuti, con l'obiettivo di minimizzare sprechi o spese inutili.

### Efficacia
Misura della capacità di un prodotto o processo di raggiungere gli obiettivi prefissati.

### Efficienza
Misura di capacità di ottenere un risultato usando il minor numero possibile di risorse.

### Endpoint
Punto di accesso a una risorsa o funzionalità specifica esposta da un'API, che consente ai client di interagire con il sistema tramite richieste HTTP.

### Environmental Sensing Service (Profilo GATT)
**Profilo GATT** usato per l'invio via **BLE** di dati relativi a misurazioni di temperatura e umidità ambientali.

## F

### Fiber
Framework web basato sul linguaggio di programmazione Go, ispirato a Express.js. Rispetto a Gin, è più leggero e performante, ma offre meno funzionalità built-in, richiedendo l'integrazione di librerie esterne per alcune funzionalità avanzate.

### Fire-and-forget
Modalità di comunicazione in cui un client invia una richiesta a un server senza attendere una risposta ed in cui il server deve essere in ascolto in quell'esatto momento per ricevere la richiesta, altrimenti questa viene persa. Un esempio di comunicazione fire-and-forget è **NATS**

### Fornitore
È il singolo gruppo che si aggiudica il capitolato. Il suo obiettivo è rispettare vincoli e aspettative del progetto producendo la documentazione necessaria e il prodotto software richiesto dalla **proponente**.

### Framework
Insieme di strumenti, librerie e regole che facilitano lo sviluppo software fornendo una struttura predefinita.

## G

### Gateway
Sono hub a cui si collegano i dispositivi **IoT** per centralizzare le comunicazioni. Hanno il ruolo di comunicare con il **Cloud**, pre-elaborare i dati, garantire sicurezza e integrità.

### Gin
Framework web basato sul linguaggio di programmazione Go. Offre funzionalità built-in per routing, middleware, gestione delle richieste e risposte, e supporta la creazione di API RESTful in modo semplice.

### Git
Sistema di *controllo di versione distribuito* che permette di tracciare modifiche al codice e collaborare tra sviluppatori.

### GitHub
Piattaforma online basata su **Git** per archiviare **repository**, collaborare sul codice e gestire progetti software.

### GitHub Action
Sistema di automazione integrato in **GitHub** che esegue pipeline **CI/CD**, test, build e deploy al verificarsi di eventi.

### GitHub Issue
Strumento di **GitHub** per tracciare bug, funzionalità e attività di progetto tramite ticket assegnabili e commentabili.

### GitHub Pages
Servizio di GitHub che permette di pubblicare siti web statici direttamente da una repository.

### GitHub Secret
Strumento di GitHub che permette di memorizzare in modo sicuro variabili d'ambiente e credenziali, che possono essere utilizzate nei workflow di GitHub Actions senza esporre informazioni sensibili nel codice sorgente.

### Go
Linguaggio di programmazione ad alto livello, compilato e tipizzato staticamente. È utilizzato per applicazioni **back-end**, sistemi distribuiti e applicazioni ad alta concorrenza.

### Gofumpt
Strumento di formattazione del codice Go che applica regole più rigorose rispetto a `gofmt`, garantendo uno stile di codice più uniforme e leggibile.

### Golangci-lint
Strumento di linting per il linguaggio Go che integra più linter in un'unica interfaccia, permettendo di identificare e correggere problemi di stile, errori comuni e potenziali bug nel codice.

### Google Cloud Platform
Piattaforma **Cloud** di Google che fornisce servizi di computing, database, storage, machine learning, networking e molto altro.

### Grafana
Piattaforma per la visualizzazione e l'analisi di dati, spesso utilizzata per monitorare metriche e log di sistemi IT, inclusi quelli generati da dispositivi IoT.

### GraphQL
Linguaggio di query per **API** che permette ai client di richiedere esattamente i dati necessari e ridurre l'*over-fetching*, ovvero la ricezione di dati non necessari

# H

### Health Thermometer Service (Profilo GATT)
**Profilo GATT** usato per l'invio via **BLE** di dati relativi a misurazioni di temperatura utilizzate in ambito medico.

### Heart Rate Service (Profilo GATT)
**Profilo GATT** usato per l'invio via **BLE** di dati relativi a misurazioni di battito cardiaco.

### HTML
L'*HyperText Markup Language* è il linguaggio usato per strutturare contenuti nelle pagine web.

### HTTPS
Protocollo di comunicazione sicura su Internet che estende HTTP tramite cifratura **TLS/SSL**, garantendo autenticità del server, integrità dei dati e riservatezza del traffico tra client e server.

### HyperTable
Struttura di tabella in TimescaleDB che consente di gestire grandi volumi di dati temporali suddividendoli in partizioni basate su intervalli di tempo, migliorando le prestazioni e la scalabilità.

# I

### Impersonificazione
Atto con cui un **Super admin** si fa temporaneamente riconoscere dal sistema **cloud** come utente appartenente a uno specifico **tenant**, attribuendo a quest'ultimo la facoltà di compiere le stesse azioni di un **Tenant Admin**. Questo può avvenire soltanto se il **tenant** impersonato ha precedentemente accettato un'apposita clausola contrattuale.

### Incremento
Aggiunta rilasciabile che estende il prodotto rispetto alla versione precedente(scopo *costruttivo*).

### IoT
L'*Internet of Things* è l'insieme di dispositivi fisici connessi a Internet (sensori, elettrodomestici, veicoli, wearable, macchinari industriali...) in grado di raccogliere, trasmettere e scambiare dati autonomamente.

### ISO/IEC 12207:1995
Standard internazionale che definisce i processi primari del ciclo di vita del software: *analisi*, *progettazione*, *realizzazione* e *manutenzione*.

### Issue Form
Strumento messo a disposizione da **GitHub** per semplificare la creazione di **GitHub Issues** che presentano campi simili. Permettono a chi crea l'*issue* di selezionare da una lista di template precompilati.

### Iterazione
Passaggio che comprende raffinamenti o rivisitazioni a scopo *distruttivo*, tornando indietro nell'avanzamento del progetto.

# J

### JavaScript
Linguaggio di programmazione interpretato, usato principalmente per lo sviluppo web lato client e lato server, per rendere interattive le pagine e le applicazioni.

### JWT
JSON Web Token è uno standard aperto per la trasmissione sicura di informazioni tra parti come oggetti JSON. Il token solitamente contiene un header, un payload e una firma, esso contiene informazioni che possono essere verificate e affidabili. Permette l'autenticazione e l'autorizzazione in sistemi distribuiti.

# K

### Kubernetes
Orchestrator di container che automatizza deploy, scalabilità e gestione di applicazioni distribuite.

# L

### Lettera di candidatura
Lettera formale rivolta ai committenti, in cui il gruppo dichiara ufficialmente la candidatura verso un **capitolato d'appalto** specifico.

### LLM
Un *Large Language Model* è un modello di IA capace di comprendere e generare testo in linguaggio naturale, addestrato su grandi quantità di dati.

# M

### Manuale utente
Documento che ha lo scopo di spiegare ai diversi tipi di utenti finali come utilizzare il prodotto software, illustrando le funzionalità disponibili e fornendo istruzioni passo-passo per eseguire operazioni specifiche.

### Micro-servizio
Architettura in cui un'applicazione è suddivisa in servizi indipendenti che comunicano tramite **API** o messaggi.

### Milestone
Rappresenta un punto nel calendario di progetto, inteso come data, il quale definisce un determinato stato di avanzamento. Questo strumento è utile per fissare obiettivi intermedi abilitando alla pianificazione a medio e breve termine.

### MKDocs
Generatore di siti statici progettato per creare documentazione tecnica, basato su Markdown e configurabile tramite un file YAML.

### Modello di ciclo di vita
Descrivono gli stati e le transizioni che caratterizzano lo sviluppo di un prodotto software, indicando quali processi devono essere attivati. Permettono di pianificare, organizzare ed eseguire il lavoro in modo strutturato, aiutando a studiare, comprendere, misurare e trasformare il sistema in sviluppo.

### MongoDB
Database **NoSQL** orientato ai documenti, flessibile e adatto a grandi quantità di dati non strutturati.

### MQTT
Protocollo di messaggistica leggero basato su publish-subscribe, ottimizzato per dispositivi **IoT** e reti con larghezza di banda limitata o instabili, con meccanismi di qualità del servizio (*QoS*) e mantenimento della connessione persistente.

### MVP
Il *Minimum Viable Product* è un'approssimazione del prodotto atteso dalla **proponente**, dotata di funzionalità minime ma sufficienti a essere testata e valutare la bontà della visione iniziale.

## N

### NATS
Sistema di messaggistica publish-subscribe ad alte prestazioni, leggero e distribuito, progettato per la comunicazione asincrona tra micro-servizi, applicazioni **Cloud**-native e dispositivi **IoT**.

### NATS Jetstream
Estensione di **NATS** che fornisce funzionalità di messaggistica persistente, supporto a code di messaggi e funzionalità avanzate per la gestione dei messaggi, rendendolo adatto a scenari che non accettano nessuna perdita di dati.

### NATS Server
Componente centrale del sistema di messaggistica **NATS** che gestisce la comunicazione tra i client, instrada i messaggi e garantisce la consegna. Il server **NATS** è progettato per essere leggero, ad alte prestazioni e scalabile.

### Nest.js
Framework modulare per **Node.js** che facilita la costruzione di applicazioni server robuste e scalabili con architettura strutturata.

### NKEY - NATS
Sistema di gestione delle chiavi di **NATS** basato su ed25519, che consente di generare chiavi pubbliche e private per l'autenticazione e la crittografia dei messaggi. Le NKEY sono utilizzate per firmare i **JWT** e garantire perciò l'autenticità dei token.

### Node.js
Ambiente runtime **JavaScript** lato server, ottimizzato per applicazioni scalabili, event-driven e in tempo reale.

### Normalizzazione
Procedura che trasforma dei dati eterogenei in un insieme di dati che condividono lo stesso formato e le stesse caratteristiche numeriche.

### Norme di Progetto
Documento esecutivo interno al gruppo che definisce procedure, regole e strumenti per organizzare il lavoro in modo professionale e ripetibile.

### NoSQL
Categoria di database non relazionali progettati per archiviare e gestire grandi quantità di dati non strutturati o semi-strutturati.

### NSC - NATS
**NATS Server Configuration** è uno strumento di gestione della configurazione del server **NATS**, che consente di creare e gestire utenti, permessi, soggetti e stream in modo semplice e organizzato.

## O

### On-demand
Servizio o funzionalità disponibile su richiesta dell'utente e attivabile quando necessario.

### Operator - NATS
Entità in **NATS** che rappresenta un'organizzazione, responsabile della creazione e gestione di Account, utenti e permessi all'interno del sistema. Esso firma il proprio JWT e quelli degli Account che gestisce.

## P

### PB
La *Product Baseline* valuta la maturità della baseline architetturale del software e la sua realizzazione, includendo il design definitivo nel documento di **Specifica tecnica**. È presente un avanzamento sostanziale del prodotto software che viene sottoposto alla **proponente** (come **MVP**) per valutarne qualità e adeguatezza.

### PDSA Cycle
Ciclo *Plan-Do-Study-Act*: modello iterativo di miglioramento continuo basato su pianificazione, esecuzione, studio dei risultati ed eventuale correzione iterando nuovamente. A seguito dello studio dei risultati può esserci l'adozione o l'abbandono della misura messa in atto.

### Piano di Progetto
Documento gestionale che pianifica tempi, costi, risorse e rischi del progetto. Serve a monitorare l'avanzamento confrontando, per ogni sprint: **preventivo**, **consuntivo di periodo**, **preventivo a finire** e analisi dei compiti svolti e dei rischi. In modo tale da ricalibrare le attività in corso d'opera.

### Piano di Qualifica
Documento gestionale che descrive come verranno svolte **Verifica** e **Validazione**, fissando gli obiettivi di qualità, le metriche e gli strumenti di controllo.

## PostgreSQL
Database relazionale open-source con supporto completo a transazioni *ACID* e capacità di gestire grandi volumi di dati con elevata affidabilità e performance.

## Preventivo
È una sezione del **Piano di Progetto**, che viene compilata ogni **sprint**, nel quale si stimano le ore e le risorse necessarie per completare le attività pianificate.

## Preventivo a finire
È una sezione nel **Piano di Progetto**, che viene compilata ogni **sprint**, nel quale viene aggiornata la pianificazione futura e le risorse rimanenti in funzione delle risorse utilizzate nello **sprint** corrente.

## Processo di ciclo di vita
Sono l'insieme di attività che guidano, secondo best practice, un cambio di fase all'interno di un prodotto software (es: concezione -> sviluppo). Un processo è composto da attività *correlate* e *coese* che trasformano *bisogni* in *prodotti*, secondo *regole definite*, consumando *risorse* nel farlo

## Product Backlog
Lista ordinata delle funzionalità, **requisiti** e migliorie da implementare nel prodotto.

## Profilo BLE
Specifica standardizzata che descrive i dettagli della comunicazione tra dispositivi **BLE**.

## Profilo GATT
**Profilo BLE** che definisce nello specifico come avviene lo scambio di dati tra due dispositivi **BLE**, tramite un modello Server–Client. Un dispositivo BLE può essere associato a uno dei molteplici profili GATT standard.

## Progetto didattico
Progetto svolto dal gruppo *GlitchHub Team* per l'esame di Ingegneria del Software del Corso di Laurea Triennale in Informatica dell'Università del Padova, svoltosi nell'anno accademico 2025—2026.

## Prometheus
Sistema di monitoraggio e allerta progettato per raccogliere e memorizzare metriche in tempo reale, spesso utilizzato insieme a Grafana per visualizzare i dati raccolti.

## Proof of Concept
Artefatto realizzato ad inizio progetto con lo scopo di valutare la *fattibilità tecnologica* del prodotto atteso. Deve rappresentare le richieste principali del capitolato.

## Proponente
È l'azienda che presenta un capitolato e richiede lo sviluppo di un prodotto software. Definisce aspettative, **requisiti**, e funzionalità. Segue e aiuta nel processo di sviluppo (offre chiarimenti, supporto tecnico e feedback).

### Prototipo
Versione semplificata del sistema o di una funzionalità con lo scopo di aiutare la scelta delle soluzioni da adottare nel prodotto finale.

### Provisioning (gateway)
Procedura di autenticazione e di associazione di un **gateway** alla piattaforma **Cloud**, in cui le due componenti si scambiano le informazioni di autenticazione e le chiavi necessarie per scambiarsi dati in maniera crittografata.

### Pull Request
Nella piattaforma **GitHub**, è una richiesta formale di unire codice da un branch a un altro (operazione di *«merge»*) in una **repository**, a seguito di una revisione.

### Pulse Oximeter Service (Profilo GATT)
**Profilo GATT** usato per l'invio via **BLE** di dati relativi a misurazioni di saturazione di ossigeno (*pulsossimetria*).

## R

### RabbitMQ
Message broker basato sul protocollo *AMQP* (Advanced Message Queuing Protocol) che permette la comunicazione asincrona tra applicazioni o micro-servizi tramite code di messaggi.

### Redis
Database in-memory ad alte prestazioni utilizzato per caching, gestione delle sessioni e code di messaggi.

### Repository
Archivio di progetto che contiene codice, file e cronologia delle versioni, può essere gestito tramite un sistema di controllo versione come **Git**.

### Requisiti di sicurezza
Vincoli e misure necessarie per garantire protezione dei dati, autenticazione, autorizzazione e continuità del servizio.

### Requisiti funzionali
Specificano cosa il sistema deve fare: funzionalità, comportamenti e risposte a determinati input.

### Requisiti non funzionali
Descrivono caratteristiche qualitative del sistema, come prestazioni, usabilità, sicurezza e affidabilità.

### Requisito
Una capacità di cui un *utente* ha bisogno per raggiungere un *obiettivo* (lato bisogno), oppure una capacità che un *sistema* deve possedere per rispondere ad un'*aspettativa* (lato soluzione).

### Requisito software
Una capacità che un *sistema* deve possedere per rispondere ad un'*aspettativa* (lato soluzione). L'**analisi dei requisiti** esplora il punto di vista lato soluzione, ovvero ciò che il prodotto deve fare per soddisfare i bisogni.

### Requisito utente
Una capacità di cui un *utente* ha bisogno per raggiungere un *obiettivo* (lato bisogno). Il capitolato specifica le aspettative della **proponente** fissate nella fase di studio del problema.

### Reset (gateway)
**Comando** inviato a un **gateway** per reimpostarlo alla **configurazione di fabbrica**.

### Retrospettiva
Riunione interna al team con valutazione *qualitativa*. Durante l'incontro si analizza cosa è andato bene o male e si discutono possibili correzioni.

### Riattivazione (gateway)
**Comando** inviato a un **gateway** per riprendere l'invio di tutti i suoi dati alla piattaforma **Cloud**, precedentemente interrotto.

### Riattivazione (sensore)
**Comando** inviato al **gateway** associato a un sensore specifico, per riprendere l'invio dei dati da esso ricevuti alla piattaforma **Cloud**.

### Riavvio (gateway)
**Comando** inviato a un **gateway** per spegnerlo e riaccenderlo da remoto, con lo scopo di risolvere eventuali errori di corruzione di memoria volatile. Il riavvio mantiene la **configurazione** del gateway.

### Riuso
Pratica che consiste nell'impiegare componenti, codice o documenti già esistenti per ridurre tempi di sviluppo e rischi. Quanto più basso è il *costo di adozione* e l'impatto sul workflow maggiore è la frequenza di riuso.

### RTB
La *Requirements and Technology Baseline* è una fase del **Progetto didattico** che fissa in modo stabile i **requisiti** da soddisfare, concordati con la **proponente**, e motiva le tecnologie, i framework e le librerie scelti. È supportata dal **Proof of Concept**.

## S

### Schema
Standard che descrive formalmente come strutturare un insieme di dati in uno specifico linguaggio di markup.

### Scrum
Framework **Agile** che organizza il lavoro in **Sprint** e definisce ruoli ed eventi per migliorare la produttività del team.

### Semantic Versioning
Schema di versionamento software basato su tre numeri *MAJOR.MINOR.PATCH* che indicano rispettivamente cambiamenti incompatibili, nuove funzionalità e correzioni di bug.

### Sensore simulato
Entità logica che corrisponde a un sensore collegato al **simulatore di gateway** simulato interamente in software che non utilizza le stesse tecnologie di un vero **gateway** o di veri sensori **BLE**.

### Simulatore di gateway
Programma che simula in software il funzionamento di un **gateway**, senza richiedere hardware specializzato a tale scopo.

### Sistema
La parte del prodotto che si prende in considerazione nel contesto di un **caso d'uso** specifico. Si noti che il sistema di un caso d'uso può corrispondere all'**attore** di un altro caso, a seconda della funzionalità che si intende descrivere.

### Slack time
Margine di ritardo tollerabile senza compromettere scadenze successive.

### Sospensione (gateway)
**Comando** inviato a un **gateway** per interrompere l'invio di tutti i suoi dati alla piattaforma **Cloud**, ma non la ricezione di comandi da questa. Si noti che la sospensione non spegne completamente il gateway.

### Sospensione (sensore)
**Comando** inviato al **gateway** associato a un sensore specifico, per interrompere l'invio dei dati da esso ricevuti alla piattaforma **Cloud**. Si noti che la sospensione non spegne completamente il sensore interessato

### SPA
Una *Single Page Application* è un'applicazione web che carica una sola pagina e aggiorna dinamicamente i contenuti senza ricarichi completi.

### Specifica tecnica
Documento che descrive dettagliatamente le scelte fatte in ambito di progettazione e implementazione, spiegando il motivo di tali scelte e fornendo informazioni tecniche necessarie per comprendere il sistema.

### Sprint
Periodo breve e definito a monte (solitamente 1-2 settimane) in cui il team sviluppa un incremento di prodotto, cercando di svuotare lo **Sprint Backlog**.

### Sprint Backlog
Insieme di attività selezionate dal **Product Backlog** e pianificate per uno specifico **Sprint**.

### Sprint Planning
Riunione di inizio **Sprint** nella quale il team pianifica il lavoro da svolgere e gli obiettivi da raggiungere.

### Sprint Retrospective
Riunione interna del team volta a migliorare il **way of working**. Durante l'incontro si analizza cosa non ha funzionato nello **Sprint** appena concluso e si discutono possibili azioni per migliorare il processo nei successivi.

### Sprint Review
Riunione di fine **Sprint** in cui il team mostra il lavoro svolto agli **stakeholder** e raccoglie feedback.

### SSL
Protocollo crittografico progettato per proteggere le comunicazioni online, oggi rimpiazzato dal **TLS**.

### Stakeholder
Chiunque abbia interesse nel progetto: **committente**, **proponente**, utenti finali, **fornitore** e figure che influenzano **requisiti** e valutazioni.

### Stream - NATS Jetstream
Concetto in **NATS** Jetstream che rappresenta un flusso di messaggi persistenti, organizzati in code, che possono essere consumati da più sottoscrittori. I stream permettono di gestire grandi volumi di messaggi con garanzia di consegna e persistenza.

### Subject - NATS
Concetto fondamentale in **NATS** che rappresenta un canale di comunicazione su cui i messaggi vengono pubblicati e sottoscritti. I soggetti permettono di organizzare e filtrare i messaggi all'interno del sistema di messaggistica.

### Super admin
Utente autenticato con poteri di amministrazione globali a tutti i **tenant** associati alla piattaforma **cloud**

## T

### Tenant
In un sistema multi-tenant, è un cliente o gruppo di utenti che condivide la stessa piattaforma, ma con dati e accessi isolati dagli altri.

### Tenant admin
Utente autenticato appartenente a un **tenant** specifico con privilegi di amministrazione all'interno del proprio tenant e sui **gateway** a esso associati.

### Tenant user
Utente autenticato senza privilegi appartenente a uno specifico **tenant**. Ha solo il potere di visualizzare i dati dei sensori ricevuti dai **gateway**.

### Test di integrazione
Verificano che più componenti o servizi del sistema funzionino correttamente insieme.

### Test di scalabilità e carico
Misurano come il sistema si comporta sotto traffico elevato e quanto riesce a scalare.

### Test di sicurezza
Verificano che il sistema protegga dati e accessi da vulnerabilità o minacce.

### Test di sincronizzazione cloud
Controllano che i dati rimangano coerenti tra dispositivi, server e servizi **Cloud**.

### Test Driven Development
Tecnica di sviluppo in cui prima si scrivono i test, poi il codice necessario a soddisfarli ed infine si ristruttura il codice. Garantisce verificabilità e qualità

### Test multi-tenant
Test che verificano isolamento dei dati e corretto funzionamento di un sistema con più **tenant**.

### TimescaleDB
Estensione di PostgreSQL progettata per gestire dati temporali, come quelli generati da sensori IoT. Offre funzionalità avanzate per l'analisi e la gestione di serie temporali, come per esempio le HyperTables.

### TLS
Protocollo crittografico che garantisce la sicurezza delle comunicazioni tra client e server su reti non sicure. Fornisce cifratura dei dati, integrità tramite checksum e autenticazione delle parti coinvolte.

### Typescript
Linguaggio basato su **JavaScript** con tipizzazione statica, utile per ridurre errori e migliorare manutenzione del codice.

### Typst
Linguaggio di markup moderno utilizzato per creare documenti in modo semplice e programmabile, simile a *LaTeX* ma con sintassi più intuitiva.

## U

### Uber Fx
Framework di Dependency Injection per il linguaggio Go, che semplifica la gestione delle dipendenze e l'organizzazione del codice, promuovendo un'architettura modulare e testabile.

### UI
La *User Interface* è l'insieme degli elementi grafici e interattivi tramite cui l'utente interagisce con un'applicazione o un sito web.

### Unit test
Test che verificano singole funzioni o componenti in modo isolato.

### User - NATS
Entità in **NATS** che rappresenta un utente specifico con credenziali di autenticazione e permessi associati. Gli utenti possono essere creati all'interno di un Account e utilizzati per accedere al server **NATS** a cui dovranno fornire un **JWT** firmato dall'Account di appartenenza e firmare una challenge con la propria **NKEY** privata.

### User story
Breve descrizione di una funzionalità significativa dal punto di vista dell'utente finale, che specifica *cosa* l'utente vuole ottenere con la funzionalità e *perché*.

### UX
La *User Experience* rappresenta l'esperienza complessiva dell'utente durante l'uso di un prodotto, considerando facilità d'uso, soddisfazione e percezione generale.

## V

### Validazione
Accertamento che il prodotto finale soddisfi le attese degli **stakeholder**. Il fornitore dimostra che tutti i **requisiti utente** siano soddisfatti con il collaudo del prodotto.

### Valutazione capitolati
Documento in cui il gruppo, secondo i pareri dei diversi componenti, elabora un pensiero comune riguardo ogni **capitolato d'appalto**

### Verifica
Accertamento che lo sviluppo non introduca errori e rispetti i **requisiti software**. Pratica che si svolge lungo tutto il periodo di progetto.

## W

### Way of working
Rappresenta l'insieme di pratiche volte a rendere l'organizzazione delle attività di progetto *il più professionale possibile*. Deve essere incrementale nel tempo e includere sempre le nuove attività nelle norme prima della loro attuazione.

### Wildcard - NATS
Carattere speciale utilizzato nei soggetti di **NATS** per rappresentare uno o più livelli di soggetti. Il wildcard "*" rappresenta un singolo livello, mentre ">" rappresenta tutti i livelli successivi. Esempio: "sensors.*"

corrisponde a “sensors.temperature” e “sensors.humidity”, mentre “sensors.>” corrisponde a tutti i soggetti che iniziano con “sensors.”

**Jaume Bernardi**

Firma del revisore interno