Logo non trovato
(assets/Logo.jpg)

BugBusters

Come Usare il POC Nexum

Guida all'installazione e utilizzo

Stato	In redazione
Responsabile	
Verificatore	
Redattori	Alberto Autiero
Destinatari	BugBusters, Eggon, Prof. Tullio Vardanega, Prof. Riccardo Cardin

Registro delle Modifiche

Versione	Data	Descrizione	Redatto	Verificato	Approvato
0.0.2	05/01/2026	Aggiunta sezione image generator	Alberto Autiero	-	-
0.0.2	31/12/2025	Migliorie alla guida	Alberto Autiero	-	-
0.0.1	29/12/2025	Prima stesura della guida all'utilizzo del POC del documento	Alberto Autiero	-	-

Indice

1 Introduzione

Questa guida spiega in dettaglio come configurare, avviare e utilizzare il Proof of Concept (POC) Nexum. Seguire attentamente le istruzioni per garantire il corretto funzionamento di tutti i componenti.

2 Prerequisiti

Prima di iniziare, assicurarsi di avere installato:

- **Ruby**
- **PostgreSQL**
- **Node.js** (per il frontend Angular)
- **Credenziali AWS** con accesso a Bedrock
- **PowerShell** (già incluso in Windows)

3 Installazione

3.0.1 Scaricare Ruby

1. Visitare https://rubyinstaller.org/downloads/
2. Scaricare Ruby+Devkit
3. Scegliere la versione x64 (64-bit)

3.0.2 Installare Ruby

1. Eseguire il file di installazione scaricato
2. Durante l'installazione, selezionare tutte le opzioni predefinite
3. **Importante**: Selezionare l'opzione per installare MSYS2 e le componenti di sviluppo (DevKit)
4. Al termine dell'installazione, verificare l'installazione aprendo PowerShell e digitando:

Listing 1: Verifica installazione Ruby

```
ruby --version
```

5. Dovrebbe essere visualizzata la versione installata)

3.0.3 Installare Bundler

Bundler è necessario per gestire le gem Ruby del progetto.

Listing 2: Installazione Bundler

```
gem install bundler
```

3.1 Installare PostgreSQL

3.1.1 Scaricare PostgreSQL

1. Visitare https://www.postgresql.org/download/windows/
2. Scaricare il PostgreSQL Installer per Windows

3.1.2 Installare PostgreSQL

1. Eseguire il file di installazione
2. Durante l'installazione:
   - Scegliere tutte le componenti predefinite
   - **Importante**: Impostare una password per l'utente `postgres` (ricordarla, servirà per la configurazione)
   - Lasciare la porta predefinita (5432)
3. Al termine dell'installazione, verificare che il servizio PostgreSQL sia in esecuzione

3.1.3 Configurare l'avvio automatico di PostgreSQL

Per evitare di dover avviare manualmente PostgreSQL ogni volta, è possibile configurarlo per l'avvio automatico:

Listing 3: Configurazione avvio automatico PostgreSQL

```
Set-Service postgresql-x64-18 -StartupType Automatic
```

Nota: Sostituire `postgresql-x64-18` con il nome corretto del servizio PostgreSQL installato. Per trovare il nome esatto del servizio:

Listing 4: Trovare il nome del servizio PostgreSQL

```
Get-Service | Where-Object {$_.Name -like "*postgresql*"}
```

3.2 Installare Node.js

Node.js è necessario per il frontend Angular.

3.2.1 Installare Node.js con winget

1. Aprire PowerShell e eseguire:

Listing 5: Installazione Node.js con winget

```
winget install OpenJS.NodeJS.LTS
```

2. Verificare l'installazione con:

Listing 6: Verifica installazione Node.js

```
node --version
npm --version
```

3. Dovrebbero essere visualizzate le versioni installate

3.2.2 Installare le dipendenze del progetto

Dopo aver clonato o scaricato il progetto, è necessario installare le dipendenze Node.js per il frontend:

1. Aprire CMD nella cartella del progetto
2. Spostarsi nella cartella frontend:

Listing 7: Spostarsi nella cartella frontend

```
cd frontend
```

3. Installare le dipendenze usando `cmd` (necessario su Windows per evitare problemi con PowerShell):

Listing 8: Installazione dipendenze Node.js

```
cmd /c "npm install"
```

4. Attendere il completamento dell'installazione

Nota: L'uso di `cmd /c` è necessario perché PowerShell su Windows può avere problemi nell'eseguire direttamente i comandi npm.

3.3 Configurare PowerShell

Per eseguire gli script PowerShell del progetto, è necessario modificare la policy di esecuzione.

3.3.1 Cambiare la policy di esecuzione

Esegui il seguente comando:

Listing 9: Cambiare policy di esecuzione PowerShell

```
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope
   CurrentUser
```

Nota: Potrebbe essere richiesto di confermare con `T`.

4 Configurazione Iniziale

4.1 Configurare PostgreSQL

Il POC richiede PostgreSQL in esecuzione con le credenziali corrette.

4.1.1 Modificare la password nel file di configurazione

Aprire il file `backend/config/database.yml` e modificare la password PostgreSQL in tutte le sezioni (development, test, production):

Listing 10: Configurazione database.yml

```
development:
  <<: *default
  database: nexum_poc_development
  username: postgres
  password: "TUA_PASSWORD_QUI"  # <-- Modifica questa riga
  host: 127.0.0.1
  port: 5432
```

Nota: Sostituire `"TUA_PASSWORD_QUI"` con la password dell'utente PostgreSQL.

4.1.2 Verificare che PostgreSQL sia in esecuzione

Avviare il servizio PostgreSQL con il seguente comando:

Listing 11: Avviare servizio PostgreSQL

```
Start-Service postgresql-x64-18
```

Nota: Sostituire `postgresql-x64-18` con il nome corretto del servizio PostgreSQL installato.

4.2 Configurare le Credenziali AWS

Il POC necessita delle credenziali AWS per accedere a Bedrock. Creare un file `.env` nella cartella `backend`.

4.2.1 Creare il file .env

Creare un file chiamato `.env` nella cartella `backend/` con il seguente contenuto:

Listing 12: File .env di esempio

```
AWS_ACCESS_KEY_ID=la_tua_access_key
AWS_SECRET_ACCESS_KEY=la_tua_secret_key
AWS_SESSION_TOKEN=la_tua_session_token
AWS_REGION=us-east-1
```

Dove trovare le credenziali AWS:

1. Accedere al portale AWS Eggon (https://eggon.awsapps.com/start/)
2. Selezionare il ruolo "Bedrock-Bugbuster"
3. Copiare le credenziali temporanee (Access Key ID, Secret Access Key, Session Token)
4. Incollare i valori nel file `.env`

Importante:

- Il file `.env` è già ignorato da git (non verrà committato)
- Le credenziali AWS scadono dopo alcune ore
- Quando le credenziali scadono, crearne di nuove dal portale AWS

5 Avvio del POC

5.1 Script PowerShell avvia-poc.ps1

Il modo più semplice per avviare il POC:

Listing 13: Script combinato

```
.\avvia-poc.ps1
```

Questo script:

- Apre una nuova finestra PowerShell per il Backend
- Apre una nuova finestra PowerShell per il Frontend
- Mostra un messaggio con tutti gli URL disponibili

Per fermare i servizi: Premere CTRL+C

6 URL Disponibili

Una volta avviati i servizi, avrai accesso a:

6.1 Applicazioni Principali

- **Frontend Angular**: http://localhost:4200
- **Backend Rails API**: http://localhost:3000

6.2 Pagine di Test HTML

- **Generazione Testo**: http://localhost:3000/tester.html
- **Analisi Documenti**: http://localhost:3000/documentTester.html
- **Generatore Immagini**: http://localhost:3000/test-generator.html

7 Come Testare il POC

7.1 Test 1: Analisi Documenti

1. Aprire http://localhost:3000/documentTester.html
2. Cliccare su "Scegli file" e selezionare un documento dalla cartella `documenti-test-ocr` (sia pdf che immagini)
3. Cliccare su "Carica e Analizza"
4. Attendere che l'AI analizzi il documento
5. Visualizzare i risultati estratti

Creare documenti di test: Per generare documenti di test più realistici, puoi eseguire:

Listing 14: Generare documenti realistici

```
python genera-immagini-test.py
```

Questo script crea vari tipi di documenti (cedolini, fatture, CUD, ricevute) sia come immagini che PDF.

Alternativa: Se preferisci creare un PDF semplice, puoi ancora usare:

```
ruby crea-pdf-test.rb
```

7.2 Test 2: Generazione Testo con Conversazioni

1. Aprire http://localhost:3000/tester.html
2. I toni verranno caricato in automatico all'apertura
3. Selezionare un tono dal menu a tendina
4. Scrivere un prompt (es: "Scrivi un'email di presentazione")
5. Cliccare su "Genera"
6. Visualizzare la risposta dell'AI

7.3 Test 3: Generazione Immagini

1. Aprire `http://localhost:3000/test-generator.html`
2. Scrivere un prompt ed attendere la risposta
3. Spostarsi ora nella sezione genera immagine e compilare i campi richiesti (prompt, dimensioni).
4. Cliccare su "Genera"
5. Attendere che l'immagine venga creata

8 Script PowerShell Disponibili

Il progetto include diversi script PowerShell utili:

Script	Descrizione
`avvia-poc.ps1`	Avvia sia Backend che Frontend (apre due finestre)
`avvia-backend.ps1`	Avvia solo il Backend
`avvia-frontend.ps1`	Avvia solo il Frontend
`verifica-postgres.ps1`	Verifica la connessione a PostgreSQL (chiede password)
`crea-pdf-test.rb`	Crea un PDF di test per l'API di analisi

Tabella 1: Script disponibili nel progetto

9 Troubleshooting

9.1 Errore: "company_id mancante"

- Assicurati di includere `company_id=1` nella query string o nel form
- Le pagine HTML di test ora includono un campo per il Company ID

9.2 Errore: "Database non inizializzato"

- Esegui manualmente: `cd backend && bundle exec rails db:create db:migrate db:seed`

9.3 Errore: "Password PostgreSQL errata"

- Verifica che la password in `backend/config/database.yml` corrisponda alla password del tuo utente PostgreSQL
- Usa lo script `verifica-postgres.ps1` per testare la connessione

9.4 Errore: "AWS credentials expired"

- Le credenziali AWS scadono dopo alcune ore
- Ottieni nuove credenziali dal portale AWS IAM Identity Center
- Aggiorna il file `backend/.env` con le nuove credenziali

9.5 Errore: "Could not find gem"

- Esegui: `cd backend && bundle install`

9.6 Errore: "Port already in use"

- La porta 3000 (backend) o 4200 (frontend) è già in uso
- Chiudi l'applicazione che usa quella porta o modifica la porta nel file di configurazione

9.7 Il server non si avvia

- Verifica che PostgreSQL sia in esecuzione
- Controlla che le variabili AWS siano configurate nel file `.env`
- Controlla i log in `backend/log/development.log`

9.8 Warning VIPS

- I warning VIPS (riguardanti moduli opzionali per immagini) possono essere ignorati - non bloccano il funzionamento dell'applicazione.

9.9 Errore: "Script execution is disabled on this system"

Se ricevi un errore quando provi a eseguire gli script PowerShell:

Soluzione 1: Cambiare la policy di esecuzione (permanente) Per l'utente corrente:

Listing 15: Cambiare policy di esecuzione PowerShell

```
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope
   CurrentUser
```

Questo comando permette l'esecuzione di script locali e richiede la firma solo per script scaricati da Internet.

Nota: Potrebbe essere richiesto di confermare con `S` o `Y`.

10 Note Importanti

- **Database**: Il seed iniziale crea una Company con ID=1, alcuni Tone di esempio e una conversazione di esempio

- **Company ID**: La maggior parte delle API richiede `company_id=1` (la company di default)
- **Storage**: I file caricati vengono salvati localmente in `backend/storage/`
- **Log**: I log del backend sono disponibili in `backend/log/development.log`
- **Hot Reload**:
  - Rails: Si ricarica automaticamente quando modifichi i file Ruby
  - Angular: Si ricompila automaticamente quando modifichi i file TypeScript/HTML

11 Riavvio Dopo Modifiche

Dopo aver modificato:

- **File Ruby (modelli, controller, ecc.)**: Rails si ricarica automaticamente
- **File di configurazione Rails**: Riavviare il POC (Ctrl+C e rieseguire `.\avvia-poc.ps1`)
- **File Angular**: Angular si ricompila automaticamente
- **File .env**: Riavviare il POC (Ctrl+C e rieseguire `.\avvia-poc.ps1`)

12 Risorse Aggiuntive

- **Gemfile**: Lista delle dipendenze Ruby in `backend/Gemfile`
- **package.json**: Lista delle dipendenze Node.js in `frontend/package.json`