



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
0.0.3	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 Nexum	Alberto Autiero	-	-

Indice

1	Introduzione	4
2	Installazione - Windows	4
2.1	Ruby	4
2.2	PostgreSQL	4
2.3	Node.js	5
2.4	Configurare PowerShell	5
3	Installazione - Linux e macOS	5
3.1	Ruby	5
3.2	PostgreSQL	6
3.3	Node.js	6
4	Configurazione Iniziale	6
4.1	Configurare le Variabili d'Ambiente	6
4.1.1	Creare il file .env	6
4.1.2	Completare le Credenziali AWS	7
4.1.3	Configurare il Database	7
4.2	Inizializzare il Database	7
5	Avvio del POC	8
5.1	Windows - Con Script PowerShell	8
5.2	Linux/macOS - Manuale	8
5.2.1	Terminale 1: Backend	8
5.2.2	Terminale 2: Frontend	8
6	URL Disponibili	8
6.1	Applicazioni Principali	8
7	Script PowerShell Disponibili	9
8	Troubleshooting	9
8.1	Errore: "Problemi di permessi in fase di configurazione"	9
8.2	Errore: "Database non inizializzato"	9
8.3	Errore: "Password PostgreSQL errata"	9
8.4	Errore: "AWS credentials expired"	9
8.5	Errore: "Could not find gem"	9
8.6	Errore: "Port already in use"	9
8.7	Il server non si avvia	10
8.8	Warning VIPS	10
8.9	Errore: "Script execution is disabled on this system"	10
9	Note Importanti	10
10	Riavvio Dopo Modifiche	10
11	Risorse Aggiuntive	11

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 Installazione - Windows

2.1 Ruby

1. Visitare <https://rubyinstaller.org/downloads/>
2. Scaricare **Ruby+Devkit** versione x64 (64-bit)
3. Eseguire il file di installazione
4. Durante l'installazione, selezionare tutte le opzioni predefinite
5. **Importante:** Installare MSYS2 e le componenti di sviluppo (DevKit)
6. Verificare l'installazione:

Listing 1: Verifica Ruby

```
ruby --version
```

7. Installare Bundler:

Listing 2: Installazione Bundler

```
gem install bundler
```

2.2 PostgreSQL

1. Visitare <https://www.postgresql.org/download/windows/>
2. Scaricare PostgreSQL Installer per Windows
3. Eseguire il file di installazione
4. Durante l'installazione:
 - Scegliere le componenti predefinite
 - **Importante:** Impostare una password per l'utente **postgres** (non dimenticarla!)
 - Lasciare la porta predefinita (5432)
5. Configurare l'avvio automatico (opzionale):

Listing 3: Avvio automatico PostgreSQL su Windows

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

Nota: Sostituire il numero di versione se diverso.

2.3 Node.js

1. Aprire PowerShell e eseguire:

Listing 4: Installazione Node.js con winget

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

2. Verificare l'installazione:

Listing 5: Verifica Node.js

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

3. Installare le dipendenze del progetto (nella cartella frontend):

Listing 6: Dipendenze frontend su Windows

```
cd frontend  
cmd /c "npm install"
```

2.4 Configurare PowerShell

Listing 7: Policy di esecuzione PowerShell

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

Nota: Confermare con Y o S.

3 Installazione - Linux e macOS

3.1 Ruby

- **macOS:** `brew install ruby`
- **Linux (Ubuntu/Debian):** `sudo apt-get install ruby-full build-essential`
- **Linux (Fedora/RHEL):** `sudo dnf install ruby ruby-devel`

Dopo l'installazione:

Listing 8: Verificare e completare Ruby

```
ruby --version  
gem install bundler
```

3.2 PostgreSQL

macOS:

Listing 9: PostgreSQL su macOS

```
brew install postgresql@18
brew services start postgresql@18
```

Linux (Ubuntu/Debian):

Listing 10: PostgreSQL su Ubuntu/Debian

```
sudo apt-get install postgresql postgresql-contrib
sudo systemctl start postgresql
sudo systemctl enable postgresql
```

Linux (Fedora/RHEL):

Listing 11: PostgreSQL su Fedora/RHEL

```
sudo dnf install postgresql postgresql-server
sudo systemctl start postgresql
sudo systemctl enable postgresql
```

3.3 Node.js

macOS:

Listing 12: Node.js su macOS

```
brew install node
```

Linux: Seguire <https://nodejs.org/en/download/package-manager>

Installare le dipendenze del progetto:

Listing 13: Dipendenze frontend su Linux/macOS

```
cd frontend
npm install
```

4 Configurazione Iniziale

4.1 Configurare le Variabili d'Ambiente

La configurazione del progetto avviene tramite il file `backend/.env`. Una copia di esempio è fornita in `backend/.env.example`.

4.1.1 Creare il file `.env`

Copiare `.env.example` e rinominarlo in `.env`:

Windows (PowerShell):

Listing 14: Creare .env su Windows

```
cd backend  
Copy-Item .env.example .env
```

Linux/macOS:

Listing 15: Creare .env su Linux/macOS

```
cd backend  
cp .env.example .env
```

4.1.2 Completare le Credenziali AWS

Aprire il file `backend/.env` e compilare le credenziali AWS:

Listing 16: Sezione AWS in .env

```
AWS_ACCESS_KEY_ID=la_tua_access_key  
AWS_SECRET_ACCESS_KEY=la_tua_secret_key  
AWS_SESSION_TOKEN=la_tua_session_token
```

Dove trovarle: Accedere a <https://eggon.awsapps.com/start/>, selezionare il ruolo “Bedrock-Bugbuster” e copiare le credenziali temporanee.

4.1.3 Configurare il Database

Nel file `backend/.env`, impostare la password PostgreSQL:

Listing 17: Database in .env

```
DB_HOST=localhost  
DB_PORT=5432  
DB_USER=postgres  
DB_PASSWORD=la_tua_password_postgresql  
DB_NAME_DEVELOPMENT=nexum_poc_development  
DB_NAME_TEST=nexum_poc_test
```

Nota: `DB_PASSWORD` deve essere la password dell’utente `postgres` impostata durante l’installazione di PostgreSQL.

4.2 Inizializzare il Database

Windows:

Listing 18: Inizializzare DB su Windows

```
# Avviare PostgreSQL, alternativamente ad aprire pgAdmin  
Start-Service postgresql-x64-18  
  
# Dalla root del progetto, eseguire  
cd backend  
bundle exec rails db:create db:migrate db:seed
```

Linux/macOS:

Listing 19: Inizializzare DB su Linux/macOS

```
cd backend
bundle exec rails db:create db:migrate db:seed
```

5 Avvio del POC

5.1 Windows - Con Script PowerShell

Eseguire il comando nella root del progetto:

Listing 20: Avviare il POC su Windows

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

Questo script apre due finestre PowerShell: una per il Backend e una per il Frontend.

5.2 Linux/macOS - Manuale

5.2.1 Terminale 1: Backend

Listing 21: Backend su Linux/macOS

```
cd backend
bundle exec rails server
```

5.2.2 Terminale 2: Frontend

Listing 22: Frontend su Linux/macOS

```
cd frontend
npm start
```

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>

7 Script PowerShell Disponibili

Il progetto include diversi script PowerShell utili:

Script	Descrizione
<code>avvia-poc.ps1</code>	Avvia sia Backend che Frontend (apre due finestre)
<code>avvia-backend.ps1</code>	Avvia solo il Backend
<code>avvia-frontend.ps1</code>	Avvia solo il Frontend
<code>verifica-postgres.ps1</code>	Verifica la connessione a PostgreSQL (chiede password)

Tabella 1: Script disponibili nel progetto

8 Troubleshooting

8.1 Errore: "Problemi di permessi in fase di configurazione"

- Esegui PowerShell/cmd come amministratore o usa `sudo` su Linux/macOS.

8.2 Errore: "Database non inizializzato"

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

8.3 Errore: "Password PostgreSQL errata"

- Verifica che la password in `backend/.env` corrisponda alla password dell'utente `postgres` impostata durante l'installazione
- Usa lo script `verifica-postgres.ps1` (su Windows) per testare la connessione

8.4 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

8.5 Errore: "Could not find gem"

- Esegui: `cd backend && bundle install`

8.6 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

8.7 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`

8.8 Warning VIPS

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

8.9 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 23: 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**.

9 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

10 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 lo script di avvio)
- **File Angular:** Angular si ricompila automaticamente
- **File .env:** Riavviare il POC (Ctrl+C e rieseguire lo script di avvio)

11 Risorse Aggiuntive

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