06 · Guida tecnica per sviluppatori

□□ Architettura, dati, permessi, configurazione, controller e verifiche per gli sviluppatori.

- □□ Architettura e mappa del progetto
- □□ Modello dati e ciclo della manutenzione
- □□ Autenticazione, autorizzazioni e menu
- □□ Avvio locale, configurazione e pubblicazione
- □□ Controller, richieste asincrone e documenti
- □□ Verifiche, problemi noti e manutenzione del manuale

?? Architettura e mappa del progetto

Destinatari: sviluppatori e manutentori. **Riferimento:** sorgenti del progetto collegato, verificati il 13 ottobre 2026.

? Componenti

Progetto	Responsabilità
BrixiaPAP	Applicazione ASP.NET Core MVC: controller, ViewModel, Razor Views e risorse web.
BrixiaPAP.DAL	Modelli EF Core, contesto SQL Server, repository e servizi applicativi.
Brixia.Shared	Modelli Custom, enumerazioni e librerie comuni, incluso InterventoPdf.
BrixiaPAP.TEST	Test MSTest con accesso ai servizi e al database configurato.

Tutti i progetti indicati hanno target **net9.0**. La soluzione è BrixiaPAP.sln.

? Flusso di una richiesta

Browser → Controller → IManagerServices → Service → IUnitOfWork / Repository → BrixiaPapDBContext → SQL Server.

I servizi restituiscono ResponseService con esito, messaggio e risultato. AutoMapper collega modelli EF e oggetti Custom. Le pagine complete usano Razor Views; le finestre e gli aggiornamenti parziali usano anche PartialView e risposte JSON.

?? Dove cercare

- **Controllers:** flussi e autorizzazioni delle azioni.
- **Views e ViewModel:** campi, condizioni di visibilità e dati della pagina.
- **wwwroot/js/app:** interazioni dei moduli e richieste asincrone.
- **wwwroot/json:** menu per ruolo.
- **DAL/BL/Services:** regole dei dati e pianificazione.
- **Shared/Custom:** enum di ruoli, azioni, esiti e periodicità.
- **Shared/Library/InterventoPdf.cs:** generazione del rapporto PDF.

? Dipendenze verificate nel progetto

Entity Framework Core SQL Server, AutoMapper, Serilog, QuestPDF, Bootstrap/Tabler, Select2, Dropzone e Signature Pad. Le versioni effettive sono quelle dei file di progetto e delle risorse installate, non quelle più recenti disponibili online.

? Fonti da mantenere allineate

Program.cs, GlobalWeb.cs, Controllers/BaseController.cs, DAL/BL/Services/ManagerServices.cs e i file .csproj.

?? Modello dati e ciclo della manutenzione

? Relazioni principali

Cliente → Sede → Presidio. Contratto → DettContratto → Scadenza → DettGiroVisita → Intervento → DettIntervento → DettInterventoAnomalia.

Una sede contiene anche Referenti ed eventuale Amministratore. Utente può essere collegato a Cliente e viene assegnato ai giri. Allegato usa IdRecord e TipoRecord per il collegamento applicativo.

? Generazione delle scadenze

ContrattoController.Scheda salva il contratto e richiama ScadenzaService.CreaScadenzeContratto, passando le righe e appConfig.AnniCreazioneScadenze. Il generatore parte da DataContratto più l'intervallo e termina a DataContratto.AddYears(AnniCreazioneScadenze).

Il campo Durata della testata è espresso in mesi nell'interfaccia, ma il ciclo esaminato usa l'orizzonte configurato in anni. Non confondere i due parametri.

? Raggruppamento e giri

RaggruppaScadenzaPerTestata raggruppa per IdSede e DataScadenza e ignora le righe senza sede. Il salvataggio del giro ricava le scadenze della testata e crea i dettagli collegati, con IsMaster=true.

La condivisione crea copie dei dettagli con IsMaster=false nei giri dei destinatari. Verificare il comportamento quando una riga viene condivisa più volte.

? Rapportino

InterventoController usa E_STATO_APERTURA: NUOVO=0, DA_COMPILARE=1, DA_REVISIONE_ANOMALIE=2, DA_FIRMARE=3, FIRMATO=4. Il commento sui parametri di Scheda non corrisponde completamente all'enumerazione: per l'integrazione usare i valori dell'enum.

SalvaAzione ammette un dettaglio per la coppia intervento/presidio; registra tipo azione, istante e utente. La prima azione valorizza l'inizio dell'intervento. Firmare determina l'esito dalle anomalie; Finalizza registra la fine e richiama AggiornaDatePresidi.

?? Eliminazione e audit

I modelli hanno IsEliminato e campi di creazione/modifica. Non assumere che tutte le eliminazioni siano solo logiche: ScadenzaService contiene anche comandi SQL di eliminazione fisica dei dettagli e delle scadenze marcate per rigenerazione.

? Fonti

DAL/Models, Shared/Custom, Controllers/ContrattoController.cs, Controllers/GiroVisitaController.cs, Controllers/InterventoController.cs e DAL/BL/Services/ScadenzaService.cs.

? Autenticazione, autorizzazioni e menu

? Login

LoginController verifica le credenziali tramite UtenteService, crea i claim NomeUtente, Ruolo, RuoloAuth, IdUtente, IdCliente e Img, poi effettua SignInAsync con CookieAuthenticationDefaults.AuthenticationScheme. Il ruolo indirizza alla Home generale, operatori o clienti.

? Schemi cookie

Program.cs registra due schemi: uno con nome NomeUtente e durata configurata da GlobalWeb.COOKIE_EXPIRE (24 ore), e lo schema standard Cookies con durata di un’ora. Il login usa lo schema standard e imposta ExpiresUtc a un’ora. Per modificare la sessione verificare lo schema effettivamente usato.

? Policy

Le policy sono definite in Program.cs e GlobalWeb.cs. RuoloAuth contiene il nome dell’enumerazione, non il valore numerico. I gruppi ammessi sono esplicitamente elencati per ciascuna policy.

Policy / area	Ambito verificato
ADMIN	Programmer e Super User; controller Utente e Settings.
AMMINISTRAZIONE	Programmer, Super User, Resp Amministrazione e Amministrazione; Contratto, Manutenzione, GiroVisita e Allegato.
OPERATORI	Include i ruoli interni fino a Operai; Home/IndexOperai.
CONSULTAZIONE	Include anche Consultazione; Home/Index.
CLIENTE	Include anche Cliente; profilo, logout e Home/IndexClienti.

? Menu per ruolo

BaseController.OnActionExecuting legge il ruolo e usa MenuService.LeggiMenuDaRuolo. Il servizio carica menuProgrammer.json, menuSuperUser.json, menuRespAmmi.json o menuOperai.json da wwwroot/json; il default restituisce una lista vuota.

La visibilità del menu non sostituisce l’autorizzazione lato server. Non tutti i controller e le azioni hanno un attributo Authorize nel codice esaminato: verificare anche accesso al record, sede, cliente e rapportino, oltre al ruolo.

? Recupero credenziali

Login/Dimenticata sostituisce la password prima dell’invio email. Se l’invio fallisce, il reset è già avvenuto. Qualunque modifica a questa procedura deve gestire esplicitamente tale esito.

? Fonti

Program.cs, GlobalWeb.cs, Controllers/LoginController.cs, Controllers/BaseController.cs, DAL/BL/Services/MenuService.cs e Shared/Custom/CustomUtente.cs.

? Avvio locale, configurazione e pubblicazione

? Preparare l’ambiente

Il progetto richiede un SDK compatibile con net9.0 e SQL Server raggiungibile. Ripristinare le dipendenze della soluzione. Le librerie front-end indicate in package.json comprendono Dropzone, Select2 e Signature Pad; molte risorse sono già sotto wwwroot.

```
dotnet restore BrixiaPAP.sln
dotnet build BrixiaPAP.sln
dotnet run --project BrixiaPAP --launch-profile https
```

?? Configurazione applicativa

Chiave	Uso
ConnectionStrings:CnnDB	Connessione SQL Server usata da BrixiaPapDBContext.
appConfig:FolderUpload	Cartella fisica degli allegati, esposta tramite /STORAGEWEB.
appConfig:AnniCreazioneScadenze	Orizzonte in anni della generazione scadenze.
appConfig:logoPath	Logo usato dal generatore PDF.
appConfig:Versione	Versione configurata dell’applicazione.
Serilog	Configurazione dei log letta da appsettings.

Configurare i segreti fuori dal manuale, usando il sistema di configurazione dell’ambiente. Non copiare connessioni, password SMTP o credenziali utente nelle pagine.

? Indirizzi di sviluppo

Properties/launchSettings.json prevede http://localhost:5234 e https://localhost:7167. La route predefinita apre Login/Index. Gli screenshot del manuale sono stati acquisiti dall’istanza locale su porta 7167.

? Cartella documenti

All’avvio SetupUseStaticFiles crea FolderUpload se non esiste. L’identità del processo deve poter accedere alla cartella e scrivere i file. Un percorso non accessibile può impedire l’avvio; per

sviluppo usare una cartella dedicata all’ambiente.

? Pubblicazione

```
dotnet publish BrixiaPAP/BrixiaPAP.csproj -c Release -o artifacts/publish
```

Il repository contiene un profilo di pubblicazione in Properties/PublishProfiles. Prima della distribuzione verificare ambiente, runtime, connessione, cartella documenti, logo, menu JSON e risorse wwwroot. Il semplice publish non configura da solo host, database o accessi.

? Verifica dopo il rilascio

Controllare login, menu per ruolo, ricerca di clienti, apertura di sede e contratto, filtri dei giri, PDF e download allegati. Per prove di salvataggio usare un ambiente di test con dati dimostrativi.

? Controller, richieste asincrone e documenti

Destinatari: sviluppatori. Questi percorsi descrivono il funzionamento interno dell’applicazione MVC; non costituiscono un contratto API pubblico versionato.

? Aree applicative

Controller	Azioni principali
Cliente / Sede / Referente	Index, Scheda, CreaSedeCliente, SalvaReferente.
Presidio	SalvaPresidio, GetSottoTipiByTipo, GetNumero, GetDataProssimoEvento, PopolaCombo, Duplica, Sostituisci.
Contratto / DettContratto	ElencoContratti, Scheda, ElencoProforma, SalvaDettaglio.
Manutenzione	ElencoManutenzioni, AbilitaDisabilita, LeggiNote.
GiroVisita	GeneraGiroVisita, SalvaGiroVisita, AccodaAGiroVisita, AggiungiDettaglioGiroVisita, CondividiDettGiroVisita.
Intervento	Scheda, SalvaAzione, Revisione, Firmare, Finalizza, PDF.
Allegato	Scheda, UploadFileGiroVisita, Modifica, Download, ControllaEsistenzaFile.

? Formati e risposte

I metodi dei controller possono ricevere ViewModel da form oppure JSON marcato FromBody. Per esempio GeneraGiroVisita riceve una lista di CustomScadenzaSelezionata; CondividiDettGiroVisita riceve CondividiDettGiro. Controllare le firme aggiornate dei metodi prima di chiamarli.

Le risposte non sono uniformi: sono presenti PartialView, View, oggetti con ret, success/message e InterventoViewModel con IsValid. I client devono leggere l’esito applicativo, non solo il codice HTTP.

Program.cs conserva i nomi delle proprietà JSON, imposta PropertyNameCaseInsensitive=false e WriteIndented=true. Non convertire automaticamente i nomi dei payload in camelCase.

? Upload e archivio

UploadFileGiroVisita riceve IdDettGiroVisita e Request.Form.Files; salva il file in FolderUpload, poi registra CustomAllegato con IdRecord, TipoRecord, NomeFile e Percorso. Verificare la coerenza tra file e record se un passaggio fallisce.

? PDF

Intervento/PDF carica rapportino, dettagli, presidi e anomalie; InterventoPdf genera in memoria il documento QuestPDF, restituito come application/pdf con nome Intervento.pdf. Il percorso attuale accede anche al ramo Scadenza/DettContratto: gestire esplicitamente i rapportini di appuntamenti extra.

? Fonti

Controllers, ViewModel, wwwroot/js/app, Shared/Library/InterventoPdf.cs e Program.cs.

? Verifiche, problemi noti e manutenzione del manuale

Stato: osservazioni sui sorgenti verificati il 13 ottobre 2026. Queste note descrivono punti da verificare o correggere; il manuale non ha modificato il codice dell’app.

? Strategia di verifica

BrixiaPAP.TEST usa MSTest e servizi collegati al database. UnitTestBL legge cliente e sede nel costruttore; non presumere che la suite sia isolata o priva di effetti sui dati. Eseguire i test solo su un database di prova predisposto.

Per verificare un rilascio coprire almeno: accesso e reset, menu per ruolo, cliente/sede/presidio, contratto con periodicità, creazione e condivisione del giro, azione e anomalia, firma, finalizzazione, PDF e allegati. Includere gli esiti di errore e i ruoli senza accesso.

? Problemi osservati nel progetto

Area	Osservazione e verifica richiesta
Periodicità settimanale	ScadenzaService.GeneraScadenze riassegna la data a DataContratto.AddDays(7) a ogni iterazione: può non terminare il ciclo. Correggere l’avanzamento e aggiungere una prova mirata.
Numero scadenza	Il contatore numero è inizializzato dentro il ciclo: verificare il progressivo atteso.
Excel	I rami Excel di Contratto, Manutenzione, GiroVisita e Intervento eseguono solo break.
Home e area cliente	Home generale con segnaposto; dashboard cliente con pannelli vuoti; notifiche di esempio.
Menu	Il servizio gestisce solo quattro ruoli e restituisce lista vuota per gli altri.
Permessi	GiroVisita e Allegato richiedono AMMINISTRAZIONE anche se alcune azioni sono visibili all’operatore. Verificare il percorso end-to-end e la copertura delle altre azioni.
SalvaPresidio	Risponde success=true anche dopo gestione di errori; l’avviso di numero duplicato non blocca il salvataggio.
PDF extra	Accesso al ramo Scadenza senza gestione esplicita dell’assenza del collegamento.

Finalizza	L’esito del salvataggio viene sostituito da AggiornaDatePresidi; verificare propagazione degli errori e coerenza dei passaggi.
Totale contratto	Il totale non forzato somma Prezzo delle righe; verificare il significato di prezzo rispetto a Quantità.
Rigenerazione	Contiene cancellazioni SQL fisiche; verificare collegamenti e risultato prima di applicarla a dati operativi.

?? Diagnostica

Usare i log configurati da Serilog e i messaggi applicativi. Per un problema di avvio controllare FolderUpload; per dati mancanti verificare filtri e collegamenti; per PDF controllare logo, dati e riferimenti; per email verificare configurazione senza esporre credenziali.

? Aggiornare il manuale

1. Confrontare controller, viste, ViewModel, menu e servizi dopo ogni variazione funzionale.
2. Aggiornare prima le istruzioni dell’utente, mantenendo nomi dei pulsanti e percorsi reali.
3. Sostituire gli screenshot quando cambia la schermata, usando moduli vuoti o dati dimostrativi.
4. Rimuovere dalle guide i limiti risolti e aggiornare questa pagina tecnica.
5. Verificare collegamenti, immagini, navigazione e data di revisione.

? Verifica di questa edizione

Le istruzioni sono state confrontate con i sorgenti. Gli screenshot provengono dall’istanza locale su porta 7167 e da moduli vuoti o ritagli senza dati cliente. Non sono stati eseguiti salvataggi di dati operativi per preparare le immagini.