**Project Title:**  
NeuroVirtual Web Platform for Patient Data Management and EEG/PSG Study Viewing

**1. Objective**

Design and deliver a secure, single-tenant web application—deployed per customer—that provides:

* A patient database and management portal (analogous to BW Center)
* A study viewer with advanced controls (analogous to BW Analysis, plus live-stream capability)
* Seamless integration for sites with either an existing BW Center database or only BW Analysis file folders

**2. Scope**

**In Scope**

* Patient demographics CRUD (create/edit/delete)
* Study metadata ingestion via one of two connectors (DB or file-watcher)
* Web viewer with:
  + Channel navigation, pan/zoom, speed control
  + Sensitivity (µV/div) and frequency (HP/LP/notch) filters
  + Sleep event overlays (spindles, K-complexes, REM)
  + Live-Study Mode (stream current recording via local agent)
* Role-based access control and audit logging
* Dockerized, single-tenant deployments (one instance per customer)

**Out of Scope (Phase 1)**

* AI-driven scoring or reporting
* Full BW Analysis feature parity (rescore, advanced montage)
* Offline-first / PWA functionality

**3. Macro Plan & Timeline**

| **Phase** | **Duration** | **Deliverables** |
| --- | --- | --- |
| Requirements & Planning | 2 weeks | Functional spec, use-case matrix, deployment guide |
| UI/UX Design | 3 weeks | Wireframes, component library, customer review |
| Backend Development | 6 weeks | APIs, per-customer config, connector framework |
| Frontend Development | 6 weeks | Patient portal, viewer UI, responsive templates |
| Integration & Testing | 4 weeks | End-to-end tests, connector validation, security audit |
| Pilot Deployment | 2 weeks | First-customer rollout, feedback loop |
| Release & Training | 1 week | Admin guide, user manuals, training sessions |

**4. System Architecture & Deployment**

* **Single-Tenant Deployments**
  + One Docker Compose (or Helm) stack per customer
  + Tagged container images + database migrations ensure version control
* **Core Components**
  + **Frontend**: React + TypeScript (Next.js if SSR desired)
  + **Backend/API**: Node.js + TypeScript (NestJS or Express)
  + **Local Agent**:
    - DB Connector omitted for BW Center sites
    - File Connector agent (Python service) for BW Analysis-only sites
    - Live-Study Agent (C#/.NET 6) on local workstations
  + **Database & Storage**:
    - PostgreSQL (one DB per tenant)
    - S3-compatible or on-prem file share for raw EEG/PSG files
  + **Auth & Security**: OAuth2/JWT, TLS everywhere, field-level encryption

**5. Data Ingestion Layer**

| **Connector Type** | **Target Site** | **Method** |
| --- | --- | --- |
| **DB Connector** | BW Center database sites | - Direct read of patient/study tables - ETL or CDC sync into web DB |
| **File Connector** | BW Analysis-only sites | - Python agent watches “data folder” - Parses EDF/BDF headers (e.g. via pyedflib) - Pushes metadata via secured API or direct DB writes - Quarantines corrupt files with alerts |
| **Live-Study Agent** | Any site running desktop | - .NET 6 agent streams ongoing recording over WebSocket to viewer |

*All metadata (from either connector) converges into the same PostgreSQL schema, so the UI and viewer logic remain source-agnostic.*

**6. Key Features**

**6.1 Patient Data Entry**

* Full CRUD on patient records
* Custom demographic fields
* Link/unlink EEG/PSG studies

**6.2 Web Viewer**

* **File Support:** EDF, BDF, proprietary
* **Navigation:** Channel select, pan/zoom, speed control
* **Sensitivity Control:** Adjust µV/div scaling in real time
* **Frequency Filters:** High-pass, low-pass, notch filters on-the-fly
* **Sleep Events:** Overlay automatically detected spindles, K-complexes, REM
* **Live-Study Mode:** Stream current recording via local agent WebSocket

**6.3 Security & Compliance**

* Role-based access (Admin, Tech, Physician)
* Audit trails for data changes and study access
* Encryption at rest and in transit

**7. Programming Language & Tooling Strategy**

| **Criterion** | **Python (FastAPI/Django)** | **Node.js/TypeScript (NestJS/Express)** | **.NET 6 (C#)** |
| --- | --- | --- | --- |
| **Typing** | Dynamic (+ MyPy optional) | Static (built-in) | Static (strong) |
| **Performance** | Good I/O, slower CPU-bound | Excellent I/O, async streaming | Top binary streaming, low latency |
| **Ecosystem** | Mature data/ML, file parsing | Vast npm registry, unified stack | Enterprise Windows integration |
| **Deployment** | Medium footprint, venv/wheels | Lightweight (node:alpine) | Larger runtime, but trimmable |
| **Parsing Libraries** | EDF/BDF via pyedflib, mne | JS parsers less mature | .NET libs available but fewer |
| **Recommendation** | **File Connector agent** in Python | **Backend/API** in Node.js + TypeScript **Frontend** in React + TypeScript | **Local Agent & Live-Stream** in .NET 6 |

**Rationale:**

* **Node.js/TS** for per-tenant API services offers small container size, unified full-stack development, and excellent I/O performance.
* **Python** excels at robust EDF/BDF parsing for the file-watcher agent.
* **.NET 6** provides seamless Windows desktop integration for live-study streaming.

**8. Risks & Mitigation**

| **Risk** | **Impact** | **Mitigation** |
| --- | --- | --- |
| Data-isolation breaches | High | Enforce tenant-scoped DB credentials; automated provisioning scripts |
| Connector failures | Medium | Heartbeat monitoring; retry/backoff; quarantine & alerting |
| Live-stream network issues | Medium | Buffering, reconnection logic, fallback to file-based viewer |
| Version drift across tenants | Medium | Immutable images; nightly rebuilds + vulnerability scans |
| Browser compatibility bugs | Low | CI-driven cross-browser tests (Playwright) |

**9. Assumptions**

* Customer provides server/VM with Docker runtime (or Kubernetes)
* BW Center sites expose DB access; BW Analysis sites allow agent install
* Network firewall permits agent ↔ web-app communication

**10. Dependencies**

* BW Center/BW Analysis export/schema documentation
* Docker registry & CI/CD infrastructure
* DevOps support for per-tenant automation (Helm/Terraform)
* Security/compliance review (HIPAA, FDA 21 CFR 11)

**11. KPIs for Success**

1. **Deployment Time:** ≤ 2 hours per new customer instance
2. **Study Load Time:** ≤ 10 s for a 30-minute recording
3. **Uptime:** ≥ 99.9% per tenant
4. **Sync Reliability:** ≥ 99% successful ingestion
5. **Pilot Feedback:** ≥ 80% positive satisfaction

**12. Stakeholders**

* **Project Lead:** Rodrigo Abreu
* **Frontend Team:** React/TS developers
* **Backend Team:** Node.js/TS engineers
* **Agent Dev:** Python engineer (file connector) & C#/.NET engineer (live agent)
* **QA & Security:** QA Specialist, DevOps/Security Engineer
* **Clinical Advisors:** EEG technicians, Neurologists