Agentic Solution – Phase-wise Delivery Plan

The Agentic solution is being developed to support the following key use cases:

Target Use Cases

1. **Upgrade** – Make the upgrade process simpler and faster.
2. **Product Development** – Empower R&D, CSS, Partners, and the broader Developer Community. Simplify the Product Development.
3. **Support & Documentation** – Enhance issue analysis and resolution. Provide on-demand product and feature insights through a Virtual Product Manager.
4. **QA** – Support QA efforts with automated test case generation, intelligent recommendations, and validation assistance.

Overall Plan

- The delivery is planned in **4 phases**, each with a duration of **2 months**.
- Every phase will include interim deliverables.
- Dependencies are shared between **MetricStream** (product knowledge, metadata, use case definitions) and **Coditas** (execution, development).

Phase 1: Conversion Capabilities (Duration: 2 Months)

1.1 Base Framework Setup

- Initial LLM model and agent orchestration layer
- Integration hooks for downstream conversion modules

2.1 UX Development

- Lightweight UX screens for conversions.
- File-To-File Conversion (Copy File Content & File attachment)
- Editor with Chat option

3.1 Generic Conversion → File-to-File Conversions (Driven by new UX and base framework)

Pre-requisites (to be provided by MetricStream):

- **Not required**: Product metadata, object definitions details.
- **Required**:
  - BR Guide
  - mPower API Guide

    - Report API Guide
    - Relevant conversion samples

Conversion Modules (including Syntax and Compilation Validation):

- 3.1.1 JS to BR
- 3.1.2 JS to React (via mPower APIs)
- 3.1.3 JS to React (via Report APIs)
    - 3.1.3.1 Migrate existing Kotlin-based Diff Tool (OpenAPI + GPT-4o) into this LLM flow
- 3.1.4 PLSQL to Emery
- **Delivery 1:** All the conversion can be delivered without connection to instance.
- **Delivery 2**: All the conversion can be delivered with connection to instance.

3.2 Context-aware Conversion → Editor-Based Conversion (Human Input → BR/React via BR API/mPower/Report APIs)

Pre-requisites:

- Product metadata and object definitions using GET APIs. API details from
- **Dependency:** Use case definitions for Editor from (MetricStream)
- Editor usecase for Developer community- JS - Team Apps - Confluence
- 3.2.1 JS to BR → Need to have context of
    - Module → Objects → FORMs → FORM
    - Module → Objects → FORMs → FORM --> FORM Fields/Sections/Grid/container/form header/etc…
- 3.2.2 JS to React (via mPower APIs) → Add context details
- 3.2.3 JS to React (via Report APIs) → Need to have context of Module, Objects, Reports, Report
- 3.2.4 PLSQL to Emery → Need to have context of
    - Module → Objects → Workflows → Workflow
    - Module → Objects → Workflows → Workflow --> WF Transition/Submit Action/Custom Function/etc…
    - Module → Objects → Workflows → Workflow --> config.json (in instance File System) → List all PLSQL & Emery hook with order

3.3 Conversion Validation

Pre-requisites:

- Currently the BRs & Emerys get validated during runtime. MetricStream **needs to find out way to do the same validation during design time**. This validation will solve a lot of time for all developers' community.
- **Dependency:** Provide the way to do validation in design time to Coditas - (MetricStream)
- 3.3.1 Every Conversion File-to-File or Editor will provide the Converted file or BR/Emery/React are functionally validated.

4.1 Phase 2 Definition & UX Prototype