Fase 1: Configuración del Entorno y Modularidad

Este paso asegura que todas las librerías y permisos estén definidos.

Paso	Archivo	Concepto Clave y Propósito
1. Dependencias Externas	`pom.xml`	**Ajustar Dependencias:** Añadir las librerías **`javafx-controls`**, **`javafx-fxml`** y **`lombok`** (con *scope* `provided`) al `pom.xml`. Esto proporciona las herramientas necesarias para construir la GUI.
2. Modularidad Jigsaw	`module-info.java`	**Permitir Inyección:** Usar **`requires javafx.fxml;`** y, crucialmente, **`opens org.example.practicajavafx.controllers to javafx.fxml;`**. Esto permite que el `FXMLLoader` acceda a los miembros `@FXML` privados de tus Controladores. También se añade **`requires static lombok;`** para resolver errores de compilación.

Fase 2: Estructura de Lanzamiento y Utilidades

Define el punto de entrada y la lógica para cambiar de escena.

Paso	Archivo	Concepto Clave y Propósito
3. Punto de Entrada	`HelloApplication.java`	**Carga Inicial del FXML:** Implementar el método **`start(Stage stage)`** que usa **`FXMLLoader`**. La llamada **`getResource("/login-view.fxml")`** es clave para cargar desde la raíz de `resources`.
4. Utilidad de Navegación	`JavaFXUtil.java`	**Aislar el Cambio de Escena (MVC):** Crear el método estático **`loadScene`** para reemplazar la escena actual (`Scene`) en el `Stage` dado, centralizando la lógica de carga FXML.

Fase 3: El Contrato View-Controller (Login)

Define la interfaz de usuario en Scene Builder y la lógica básica en el Controller.

Paso	Archivo	Concepto Clave y Propósito
5. Diseño del Login (FXML)	`login-view.fxml`	**Definir el Contrato (Scene Builder):** 1. Asignar **`fx:controller`** al elemento raíz (`AnchorPane`). 2. Asignar **`fx:id`** a los campos de entrada (`usernameField`, `passwordField`, `messageLabel`). 3. Asignar **`onAction="#handleLoginAction"`** al botón.
6. Lógica del Login	`LoginController.java`	**Inyección y Navegación:** 1. Usar **`@FXML`** para inyectar componentes. 2. Implementar `handleLoginAction` para la validación simulada (`if ("pedrito".equals(username) && "tronco23".equals(password))`. 3. Obtener el `Stage` con `(Stage) ((Node) event.getSource()).getScene().getWindow()` y llamar a `JavaFXUtil.loadScene()` para la navegación.

Fase 4: Modelo de Datos y Listado (Main View)

Implementa el corazón de la aplicación, la tabla de datos.

Paso	Archivo	Concepto Clave y Propósito
7. Clase Modelo	`Tarea.java`	**Propiedades Observables:** Definir los atributos usando **`StringProperty`** (`titulo`, `prioridad`) y crear los métodos **`tituloProperty()`** y **`prioridadProperty()`**. Esto es **crucial** para el enlace de datos con la tabla.
8. Diseño del Main View (FXML)	`main-view.fxml`	**Estructura de Listado:** Usar **`BorderPane`** como raíz (para la división de regiones). Colocar un **`HBox`** (para botones CRUD) en la región **`TOP`** y el **`TableView`** en la región **`CENTER`**. Asignar `fx:id` a la tabla y a sus columnas.

9. Enlace de Datos de la Tabla	`MainController.java`	**Configuración con `initialize()`:** Dentro del método `@FXML public void initialize()`, usar **`setCellValueFactory`** con **`PropertyValueFactory<>("propiedad")`**. El *String* (ej: `"titulo"`) debe coincidir con el nombre del método `Property` del Modelo. Usar **`FXCollections.observableArrayList`** para los datos simulados.