📌 FM Receiver + Real-time Spectrum Visualizer

(Model-View-Controller) Software Development Plan

Layer	Breakdown	Modules	Files

Model (DSP & Data)	- IQ Data Handling
	- DSP Processing
	- Demodulation
	- Spectrum Analysis
		1. **File Reader** (binary IQ → numpy arrays)
		2. **IQ Streamer** (async live capture)
		3. **DSP Core** (filters, FFT, FM demod)
		4. **Audio Output** (to speaker/file)
		5. **Signal Utils** (windowing, normalization)
			`src/model/file_reader.py`
			`src/model/iq_streamer.py`
			`src/model/dsp_core.py`
			`src/model/audio_output.py`
			`src/model/utils.py`

View (GUI / Visualization)	- Spectrum Display
	- Waterfall Display
	- Audio Controls
	- Status Logs
		1. **Main GUI** (Qt/Tkinter/ PySide)
		2. **Spectrum Plot** (Matplotlib/ PyQtGraph)
		3. **Waterfall**
			`src/view/main_window.py`
			`src/view/spectrum_plot.py`
			`src/view/waterfall_plot.py`
			`src/view/audio_panel.py`
			`src/view/status_window.py`

Layer	Breakdown	Modules	Files
		Plot 4. **Audio Control Panel** 5. **Status Window**	

Controller (Orchestration)	- Connects Model ↔ View
	- Handles User Actions
	- Coordinates Threads
		1. **Main Controller** (app entrypoint)
		2. **Pipeline Manager** (routes IQ → DSP → Audio)
		3. **Event Handler** (UI/ config)
		4. **Config Manager** (sampling rate, freq, file paths)
			`src/controller/main_controller.py`
			`src/controller/pipeline_manager.py`
			`src/controller/event_handler.py`
			`src/controller/config_manager.py`

Data / Config	- Captured files
	- Configurations
	- Logs
		IQ data, Config JSON, Logs	`data/raw/*.bin`
			`data/config/*.json`
			`logs/*.log`

Tests	- Unit tests per module
	- Integration tests
		Test cases per module	`tests/test_dsp_core.py`
			`tests/test_file_reader.py`
			`tests/test_pipeline.py`