

# Meeting Planner

*By Nick Müllner*

<https://github.com/NickMuellner/SWEN2-meeting-planner>

## Table of Contents

Architecture.....	2
Layers .....	2
Library decisions .....	2
Design patterns .....	2
Class diagram .....	3
Testing.....	4

# Architecture

## Layers

- **View Layer**  
Built with JavaFX and FXML, this layer handles user interaction. Controllers respond to user input and delegate actions to the ViewModel.
- **ViewModel Layer**  
The ViewModel acts as an intermediary between the View and the Model. It handles input validation, state management, and event propagation.
- **Model Layer**  
Contains core domain classes such as Meeting, MeetingNote, and relevant data structures. These classes represent the internal state of the application.
- **Service Layer**  
Provides application logic and persistence operations (CRUD). Services abstract database access and perform logic related to meetings and meeting notes.
- **Utility Layer**  
Includes reusable helpers like the EventManager (custom event bus), report generation, and JPA Manager.

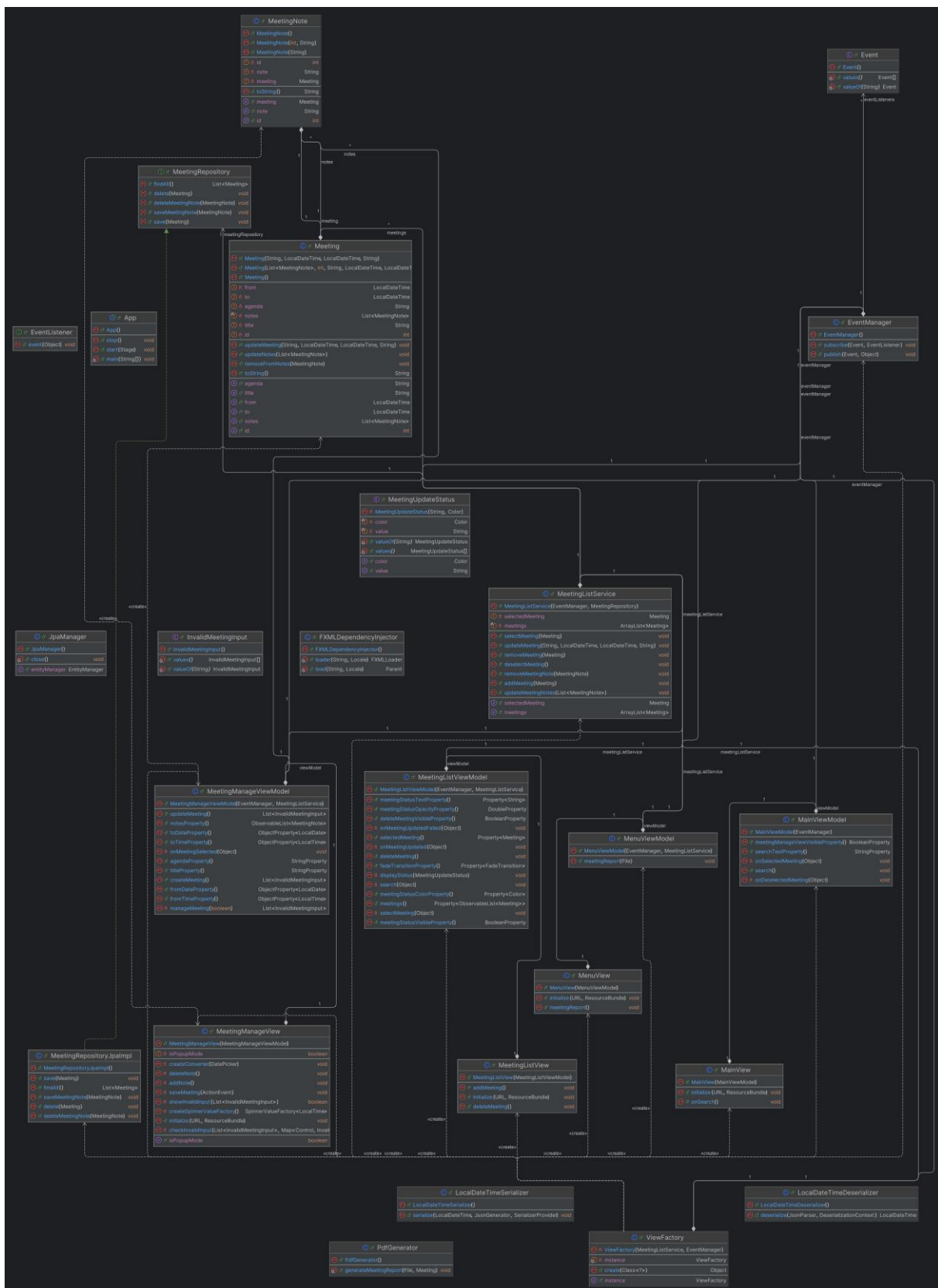
## Library decisions

- **JavaFX** – GUI framework
- **Jackson (FasterXML)** – JSON serialization and deserialization
- **JUnit 5** – Unit testing framework
- **IText** – PDF generation
- **Apache Log4j** – Logging tool
- **PostgreSQL JDBC Driver** – Enables connection to database
- **Jakarta** - Persistence
- **Hibernate** – ORM

## Design patterns

- **MVVM (Model-View-ViewModel)**  
Enables separation of concerns between UI and logic.
- **Observer Pattern**  
Observables are used to propagate state changes from the model to the view.
- **Factory Pattern**  
Factories are used for service instantiation and configuration.

## Class diagram



## Testing

### *MeetingManageViewModelTest*

This test verifies validation and update logic in the view model that handles creating and editing meetings.

### *MeetingListServiceTest*

Tests the service responsible for managing meetings and meeting notes.

### *EventManagerTest* and *EventManagerAdditionalTest*

Tests the EventManager who is responsible for subscribing and publishing events through all Views.

### *MeetingTest*, *MeetingNoteTest* and *MeetingUpdateStatusTest*

Tests the model (constructor, CRUD) in the application.