University of Antwerp
Faculty of Applied Engineering

5-Software Design

Lab Session 6

20/11/2025
Jens Duym

Course Outline

Outline labs

- **Part A: UML diagrams**
  Sessions 1 – 2
- **Part B: Design Patterns**
  Session 3 – 5
- **Part C: Projects in groups of 2**
  Session 6 – 9

- **Evaluation:**
  - Entire portfolio: zip containing code, UML diagrams, AI usage
    - **Changed:** Submit **before 02-12-'25 (23:59)** on BB
    - Oral defence **4-12-'25**
  - Defence of projects

Part C
Project

Money Tracker

Project

- Choose 1:
  - Smart Meal Planner & Grocery List Generator
  - Board Game Tournament Manager
  - Travel Planner & Trip Overview Manager
- Groups per 2 (fill in form on BB **before sun 23-11-'25**, 1 per team)
  - When you have problems working in team, contact me asap.
- Use all what you have learned during the labs

Non-Functional Requirements (same for all projects)

- The Following Design Patterns **are all mandatory** to implement:
  - MVC
  - Singleton (thread safe)
  - Observer (MVC's Observer does **NOT** count)
  - Factory Method || Abstract Factory || Builder (you can choose)
- Implement **at least 1** of the following:
  - Strategy
  - Decorator
  - Command
  - Adapter
  - Façade
  - Proxy
  - Composite
  - State
- I.e. at least 5 patterns in total
- Every **design pattern** should be implemented where it **fits and is logical**

Non-Functional Requirements (same for all projects)

- UML Diagrams
  - Class diagrams
    - 1 class diagram of whole application (GUI can be abstracted to just a GUI class)
    - 1 Mini class diagram ***for each*** design pattern.
  - Use case diagram of the entire application
    - Show important actors
    - All main use cases corresponding to the functional requirements.
  - 1 Sequence Diagram of a use case
- Tests
  - All Unit tests for ***at least one*** *class*
  - At least one integration test
- Tip: Create UML first and keep it as live updated blueprint
- Tip: Create tests immediately as you go, not only at the end.

Functional Requirements

Option 1: Smart Meal Planner & Grocery List Generator

- **Manage recipes**
  - Title
  - Description
  - Ingredient list
  - optional tags (e.g. *vegetarian*, *quick*, *budget*)
  - …
- **Add/view/edit/remove recipes**

Functional Requirements

Option 1: Smart Meal Planner & Grocery List Generator

- **Manage weekly meal plan**
  - 7 days (mon-sun)
  - Per day:
    - Breakfast
    - Lunch
    - Dinner
    - Snacks
    - *Optional: Let the user configure this?*

- ***User can:***
  - *Choose recipes for each day*
  - *View the plan*
  - *Change or remove the planned recipe for a day*

Functional Requirements

Option 1: Smart Meal Planner & Grocery List Generator

- **Generate grocery list**
  - Based on the current weekly plan
  - All ingredients are collected
    - Quantities per ingredient are summed (e.g. 2x 100g pasta = 200g pasta)
  - Automatically generated and updated
  - In the grocery list:
    - User can check of items as bought
    - Add extra items manually

- **Optional:**
  - Actual data persistence across sessions (use file or database)
  - You are allowed to add custom features

Functional Requirements

Option 2: Board Game Tournament Manager

- **Manage Players:**
  - Name
  - Skill Level/rating
  - Descriptions
  - Age
  - …
- **Add / view / edit / remove players**

Functional Requirements

Option 2: Board Game Tournament Manager

- **Manage Game Types:**
  - Name
  - Min/Max Number of players per match
  - Scoring rule (points)
  - Descriptions
  - Variant
  - …
- **At least a few predefined game types**
  - E.g. Java, Catan, Carcassonne, Chess, …
- **User can:**
  - Select, edit, create, view, delete game types

Functional Requirements

Option 2: Board Game Tournament Manager

- **Create and manage tournaments**
  - Chosen game type
  - Number of rounds
  - List of registered players
  - Day of tournament
  - Descriptions
  - …
- **User can**
  - Create, edit, delete, view tournament
  - Add players
  - View basic information

Functional Requirements

Option 2: Board Game Tournament Manager

- **User can**
  - Generate pairings for each round
    - (paring strategy is can be chosen by you)
  - List of matches for that round can be displayed
  - A result for each match can be entered (e.g. winner, draw, …)
  - View results per round
- **Automatically calculate total points per player**
- **End of tournament:**
  - overall ranking + intermediate standings for each round.
- **Optional:**
  - Actual data persistence across sessions (use file or database)
  - You are allowed to add custom features

Functional Requirements

Option 3: Travel Planner & Trip Overview Manager

- **Manage trips**
  - Title
  - Destination
  - Start/End date
  - Description
  - …
- **Create/view/edit/delete trips**

Functional Requirements

Option 3: Travel Planner & Trip Overview Manager

- **Manage travellers / participants**
  - Name
  - Contact info
  - Age
  - Nationality/passport info
  - …
- **Create/view/edit/delete travellers**
- **User can:**
  - Assign travellers to one or more trips
  - Edit travellers from trip
  - Remove travellers from trip
  - View travellers participating in a certain trip

Functional Requirements

Option 3: Travel Planner & Trip Overview Manager

- **Manage itinerary (schedule) per trip**
- **Itinerary items**
  - Title
  - Date/time
  - Type (transport, accommodation, activity, …)
  - Optional Location
  - Description
  - Price
  - …
- **Create/view/edit/delete itinerary items for each trip**
- **Ability to view a day overview for each day of the trip**
- **Automatically calculate/adjust total price for each trip**

Functional Requirements

Option 3: Travel Planner & Trip Overview Manager

- **Optional:**
  - Actual data persistence across sessions (use file or database)
  - You are allowed to add custom features (e.g. packing list)

Project

- Using code or diagrams of others:
  - From fellow students: **discouraged**
    Only grades on what is yours, not from others
    Code or diagrams from other groups = **plagiarism**
  - From the internet:
    Feel free to reuse code -> give credit!
    (credits: websites, names, YouTube-links, StackOverflow, …)
- Try to use Git and GitHub
  - Ideal for group projects, with features as branching and committing
  - **It's for your own safety**
  - In case you've never heard from Git: an introduction
    (https://www.youtube.com/watch?v=SWYqp7iY_Tc)
- Useful link: https://www.gofpattern.com/index.php