Protocol

[README.md to find on GITHUB TourPlannerSWENII]

Technical Steps:

Using WPF; it was an requirement of the course SoftwareEngineeringII, so it was quite clear, that we use WPF with programming language C#. As everyone of us already knew IDE Visual Studio, so we used Visual Studio 2022 within .NET 7.0 version for it.

As this course was ment for improving our skills in layered architecting, we firstly created an WPF project within some Subprojects like an BusinessLayer for the Logic, DataAcessLayer for acess to the database, Models for Handling our Objects correct.

Later there was an Subproject created within Utils and another one for Tests.

In the beginning, we took care of the basic framework of what the project should look like. Then it was about adding a view events and folders for the MVVM pattern.

Main focus on our technical steps was to stay on another branch than the Main branch, mostly all of us used their own feature branch, later on merging in the dev branch and in the end it was merged to the main.

In the future we consider beginning with TDD.

Decisions:

We decided to go for Observable pattern as our research came to the conclusion, that MVVM architecture and Design Pattern named Observable pattern matches quite good. For easier testing and maintaining.

Failures:

We recognized, that it is not possible to add a WalkingTour with too long Distance; for instance from Wien to Graz it is not possible to be calculated(as an Pedestrian- Input) at the moment.

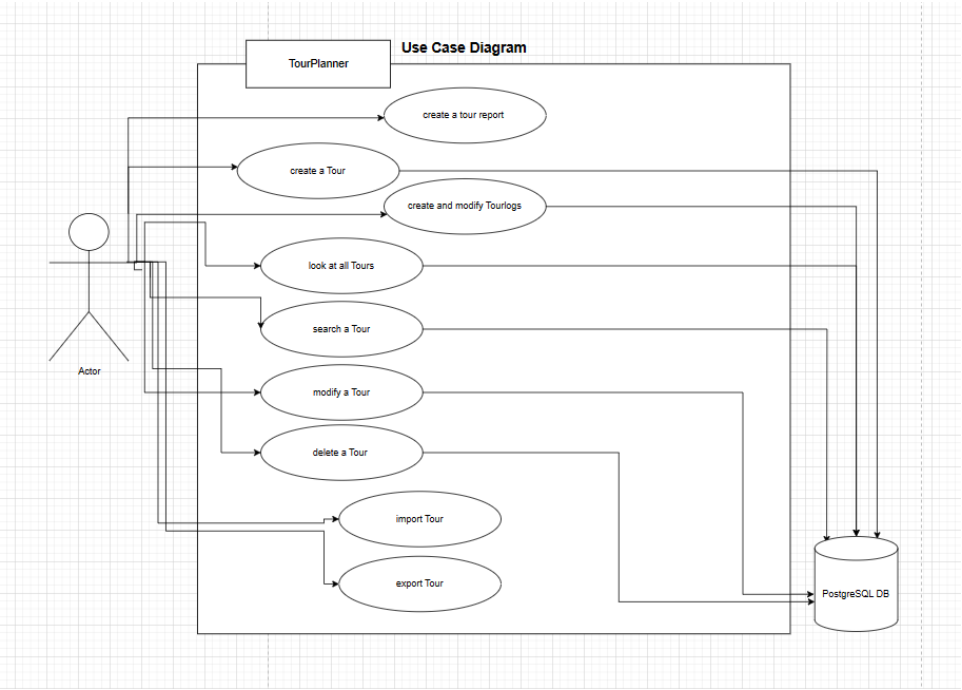
There was also once the failure, that the key was extracted but just in the RouteMethod and forgotten in the ImageMethod and then the Distance was not updated.

As our skills are not that improved like senior programmers we might failed a bit with the timing. So that we missed implementing the following things:

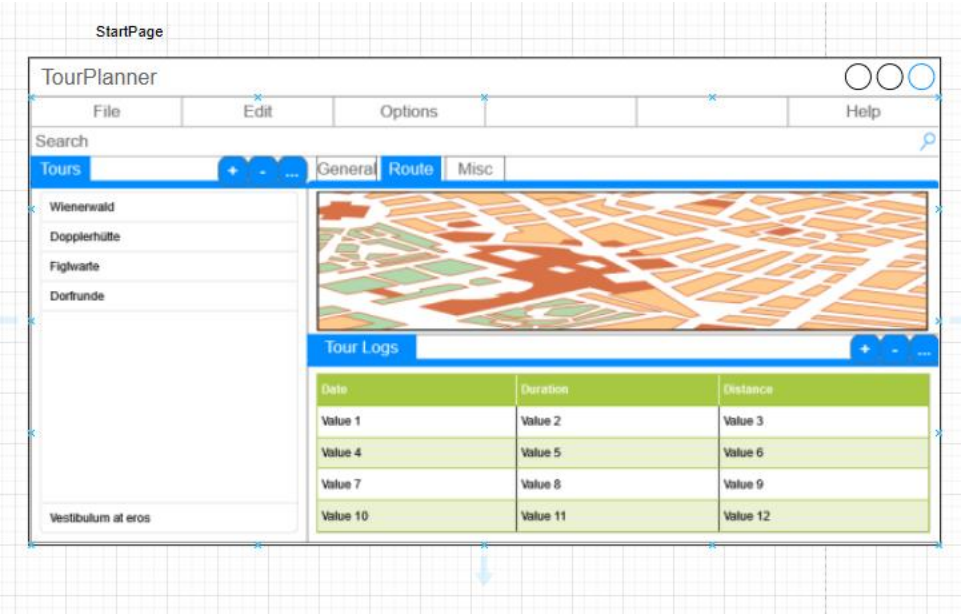
- Popularity calculated (computed Attribute)

Unfortunatelly we failed a bit by implementing the Unique Feature, becaucse we run a bit in a maze by seperating the SearchFunctions from searching just the Tourlogs and searching Tours separatly, firstly there is for instance an comment ‘muh’ which is added to an Tourlog, so if the user wants to search for a Tour he also will get the Tourlog with that comment added. So it is bounded together, if Tour is searched, it will also iterate through the Tourlogs.

UML use case diagram [see also in the github repo, there it is also linked.]



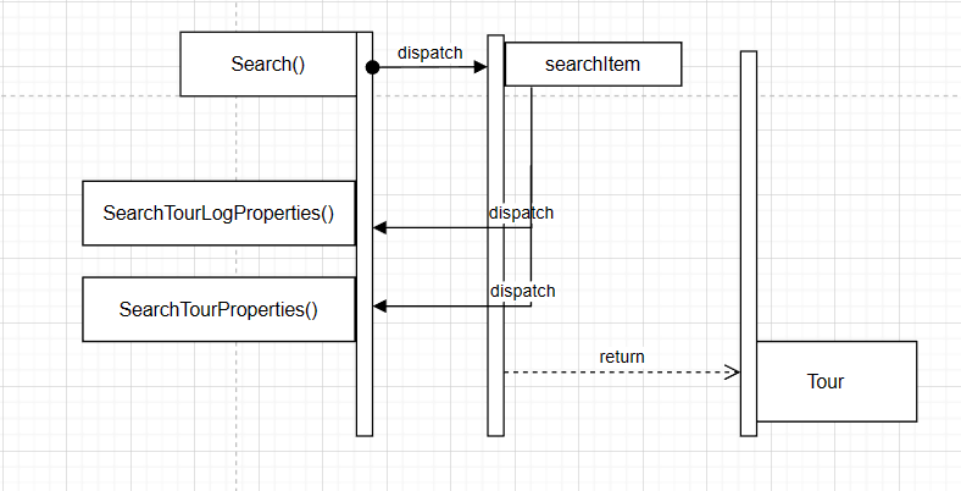
• document your UI-flow using wireframes



• document the application architecture using UML:

o class diagram

o sequence diagram for full-text search



• explain why these unit tests are chosen and why the tested code is critical

UNittests – Why we choosed this kind of Unittests; is that, the main focus on the Project is to add/ edit / delete Tours, so we tested out our Methods AddTour, GetTour, UpdateTour, EditTour, SearchTour, to check if the List of Tours is correctly responded.

Unique Feature:

In the application it is possible, to use the Searchfunction for searching in Tours. But the unique feature is, that it has a second function, which could be acgtivated by ticking on a RadioButton in the UI to search in the TourLogs.

Learning Outcomes:

We for sure got knowledge about the WPF and we have done a lot of work with git (different branches). Also we learned to how to treat with the MVVM and the observable pattern. For sure one step for learning outcomes in our group was to define things/ to name something well;so that searching for it will be better and everyone is talking about the same thing to get the same result of wanted implementation.

Time Tracked:

Miriam	Yousef	Johanna
80 hours	76 hours	72 hours

Link to GIT

https://github.com/Johanna0815/TourPlanner_4_SWENII