Leon-Vincent von Voss 4. Semester – Bachelor: Informatik Tagesform
Klemens Hamburger

TourPlanner- Protocol

App architecture

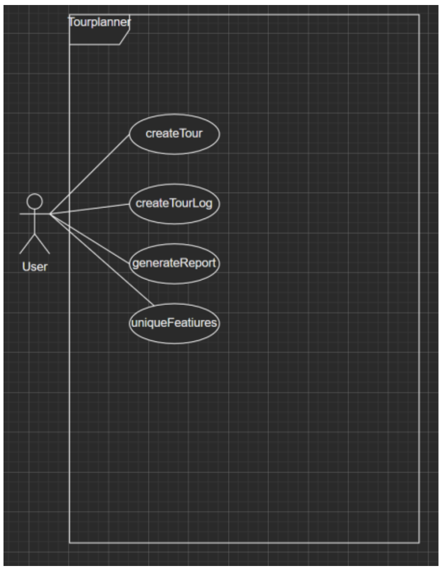
We had to use the MVVM pattern, so we implemented this pattern. Our project is structured in Model (business logic, dal), View (controllers), ViewModel (viewmodels).

Model (business logic, dal): Here we do CRUD operations on the postgres-database and the business logic controls all the requests of the view (controllers) with its implemented managers. Here are also the services 'ApiClient' (for map api request) and PDFGenerator (for user action to generate a report).

View (controllers): Here we represent the Data from the Viewmodels to the user.

ViewModel (viewmodels): Here we take all the user action and ether call the model or for simple tasks we do that in the controllers itself. The View is between the Model and the ViewModel and controls the UI.

Use cases



createTour:	A user can create a tour.
createTourLog:	A user can create a log for a specific tour.
generateReport:	A user can generate a report (PDF).
unqiueFeatures:	A user can get the information who programmed the project.
	A user can choose a random tour by click on the menu 'Feature'.

viewTourandLogs: A user can click and review a tour with the corresponding logs.

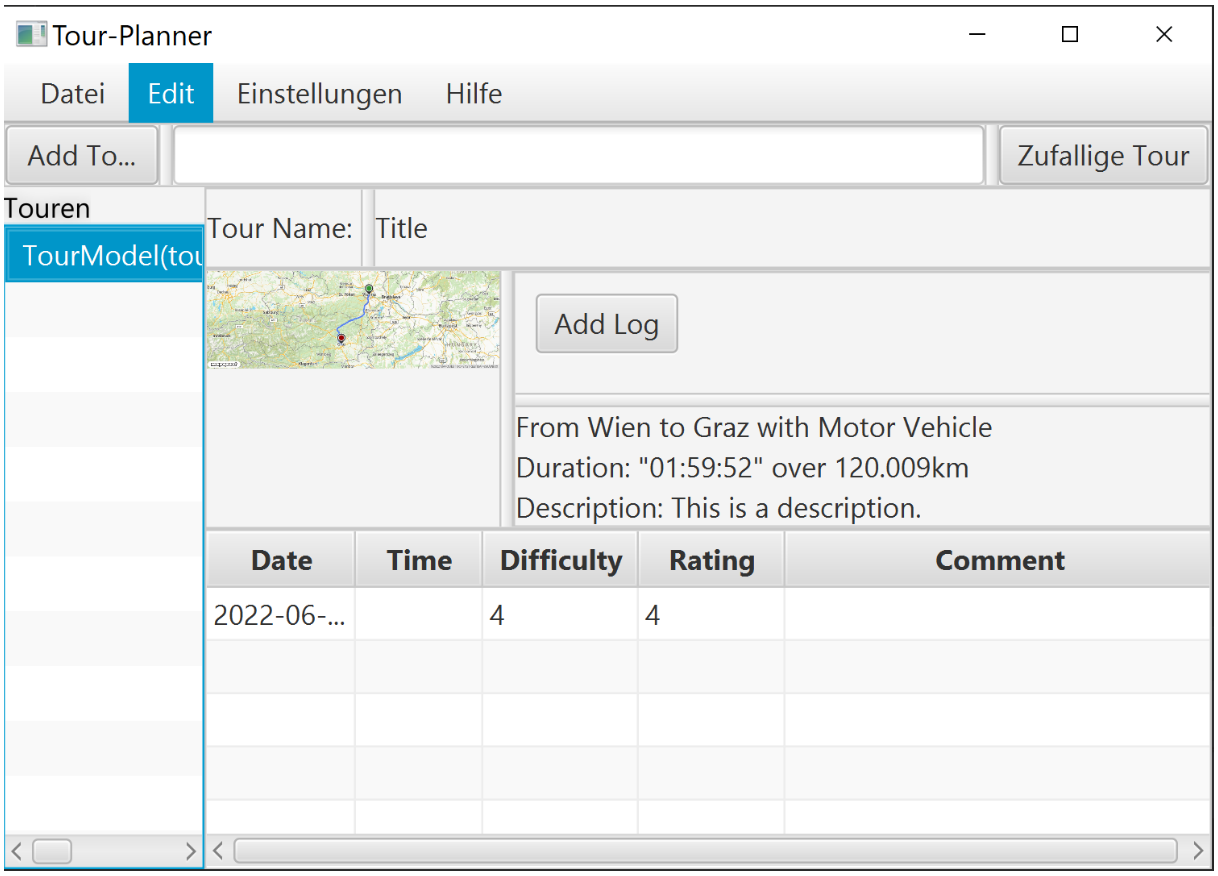
openRandomTour: A user can view a random tour.

UX

We decided to create a really simple design, for the user to understand quickly.

Here you can see the design:

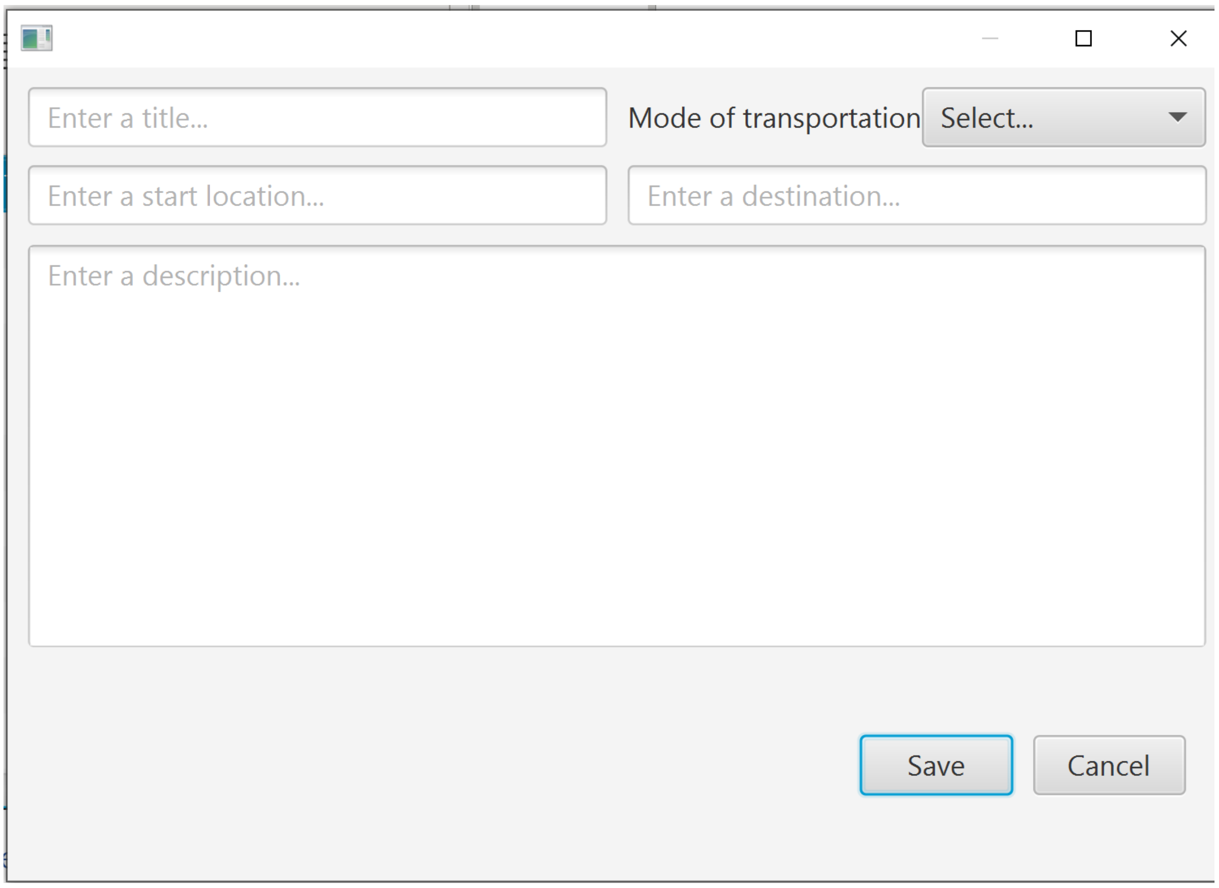
Homescreen:



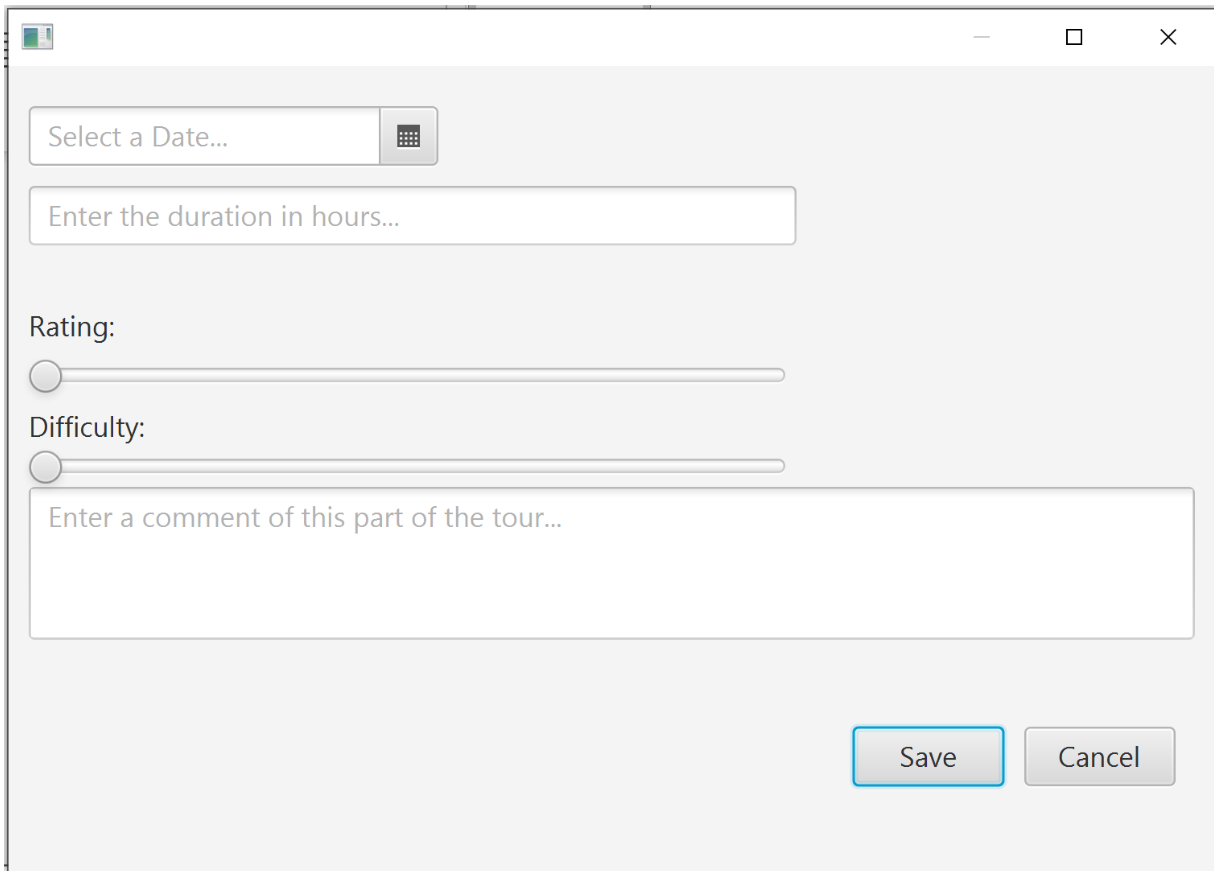
From Wien to Graz with Motor Vehicle
Duration: "01:59:52" over 120.009km
Description: This is a description.

Date	Time	Difficulty	Rating	Comment
2022-06-...		4	4	

AddTour Screen:



AddLog Screen:


Select a Date...
Enter the duration in hours...
Rating:
Difficulty:
Enter a comment of this part of the tour...
Save
Cancel

Leon-Vincent von Voss 4. Semester – Bachelor: Informatik Tagesform
Klemens Hamburger

Lessons learned

We learned how program a project with a javafx frontend. Also, how to work with GIT and data binding.

Unit tests

In the project we decided to write tests for the daos (TourDaoTest, TourLogDaoTest) and for the model (TourModelTest). We have chosen to test these classes because they are mandatory for the database. If the database classes do not work correctly, the hole program does not work, because it cannot save, delete, or select data.

Unique features

The unique features of our project are:

- If you click on the menu on 'Hilfe' and then on 'Über', a popup will appear and you are able to see, who programmed this project.
- If you click on the button "Zufallige Tour", a random tour, of all tours which exist, will be selected and shown to you and its logs too.

Tracked time

Leon von Voss: 42 Hours
Klemens Hamburger: 50 Hours

Git

https://github.com/Leonvonvoss/TourPlanner