Student: Maciej Mateusz Madejsza  
ID: MAD21541198  
Date: 21/12/2023  
Project: Coursework - Sprint 3  
Target: Higher marks - enhanced version

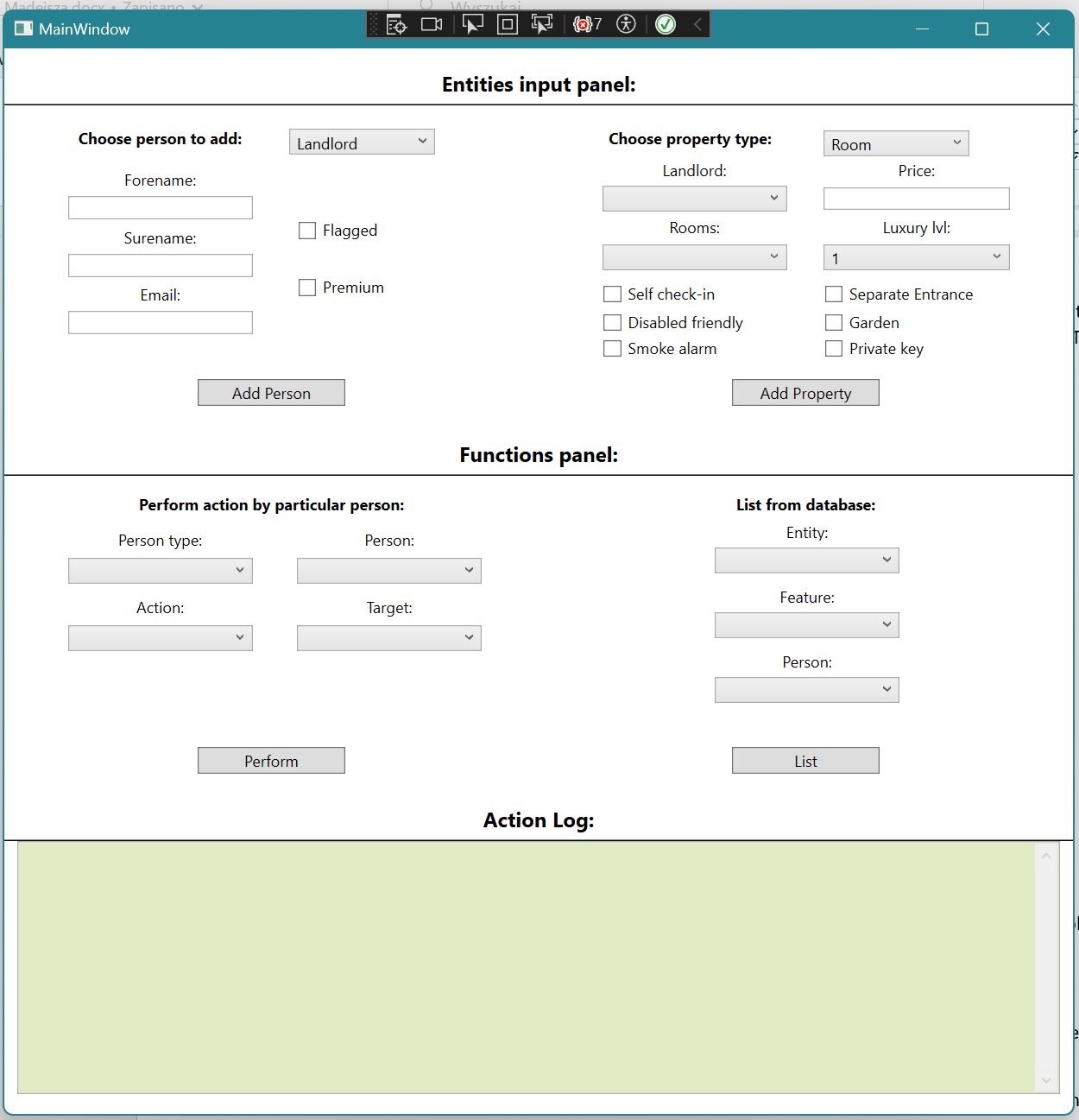
In the following document, I’ll do my best to walk you through all coursework requirements I’ve met with outcome screenshots and descriptions with screenshots of the enhanced version. To notice all extra features and robustness It’ll be required however to play around with the program – It’s a bit too extended for documentation with screenshots here.

**Interface Layout:**

Area to input entities to the program **like in Sprint 2** – users and bookable items

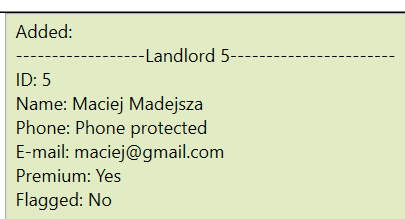
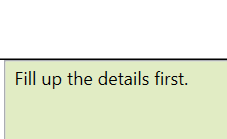
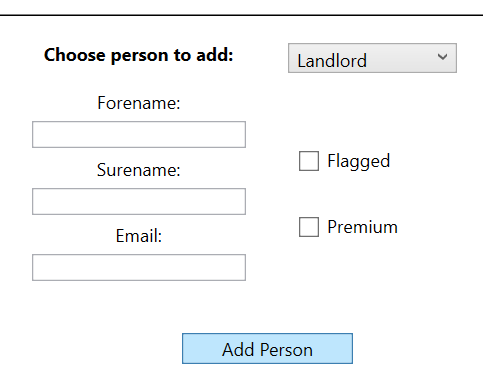
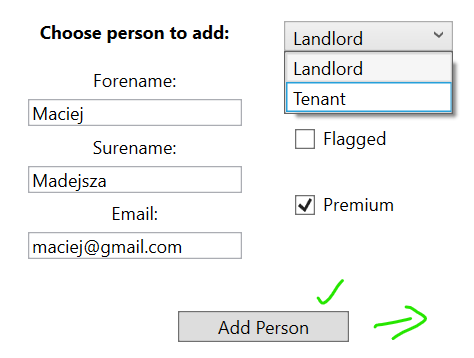
Area to perform more complex actions inc. **booking**

Area displaying actions outcome and instructions when needed.



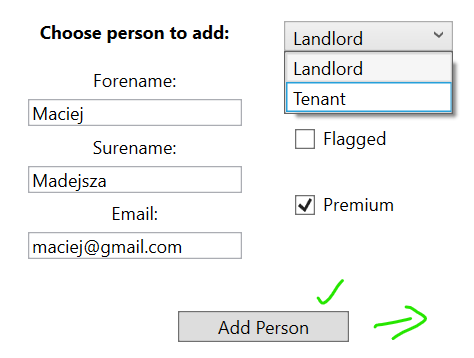
**Sprint 3 Basic requirements (3rd – 2.2)**

• Button that can respond to a click

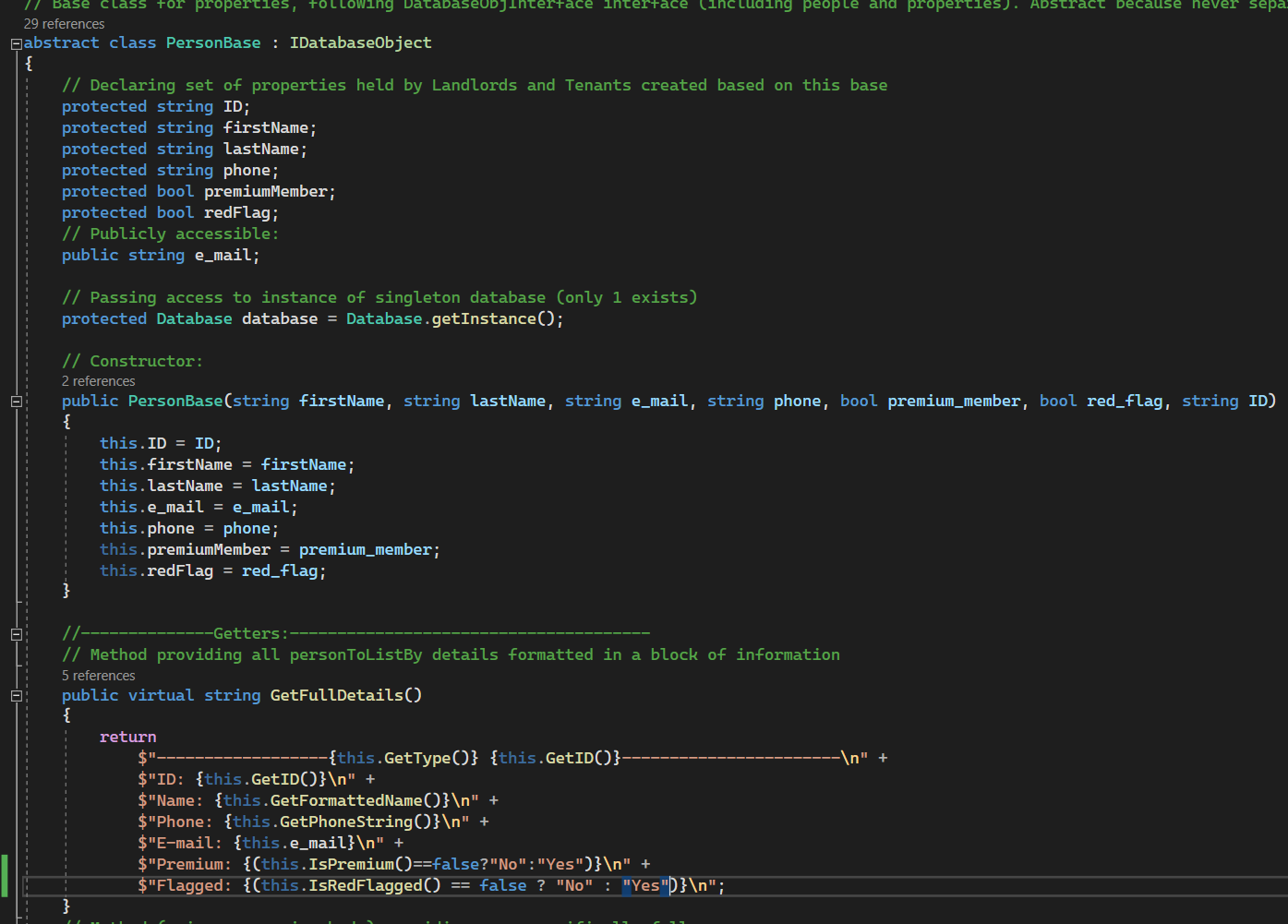
* Phone restricted to limit inputs amount for coursework assessment purposes

Empty form

validation

• List box implemented

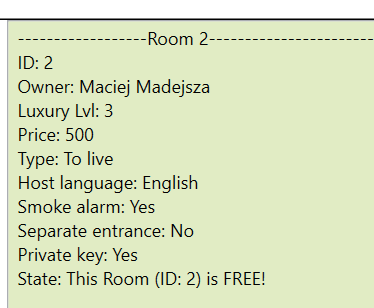
* Plus many other visible everywhere – good way to validate by limiting certain actions based on certain choices

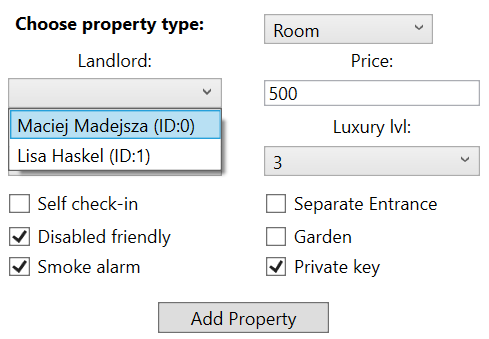
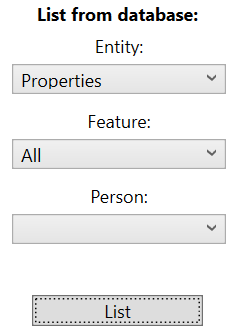
• Use of classes from sprint 1 in ‘code behind’

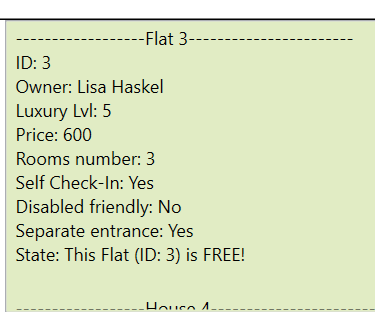
Part of the Landlord base class with method responsible for printing out the details like above after adding to the database

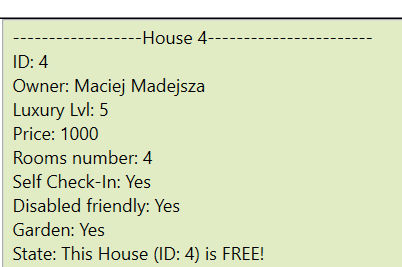
• List at least three bookable items displayed in the GUI generated by your sprint c# OOP code

* Like in sprint 2 – we have this ability to input them manually – not generated randomly





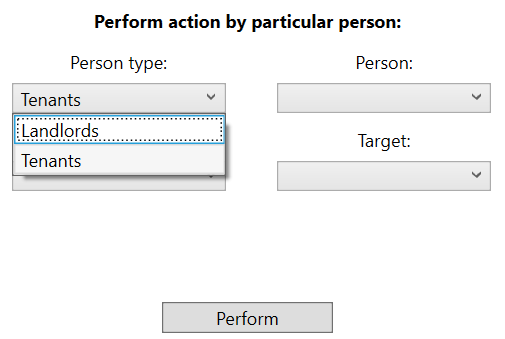
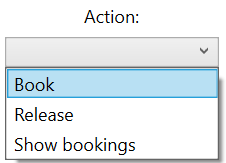
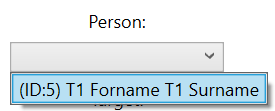
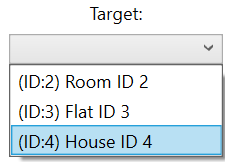
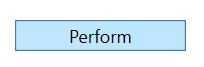
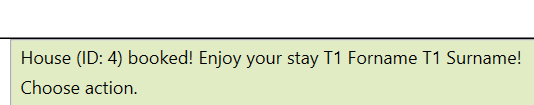
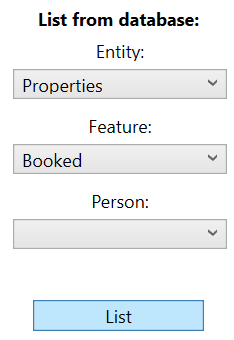
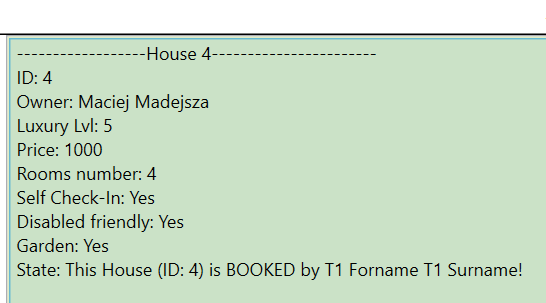




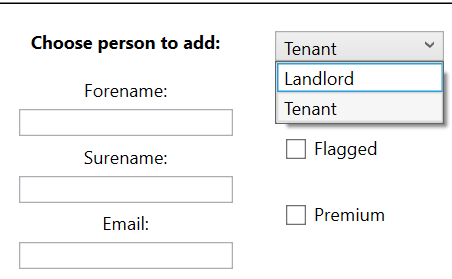
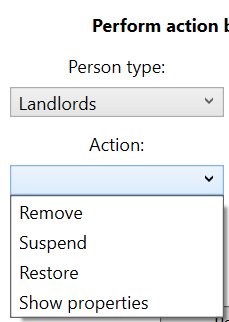
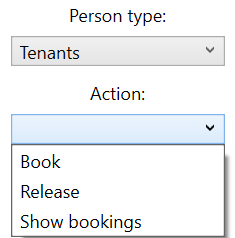
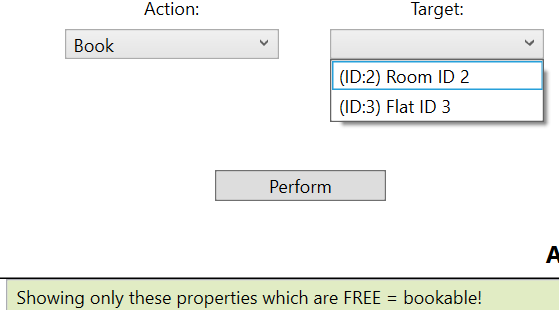
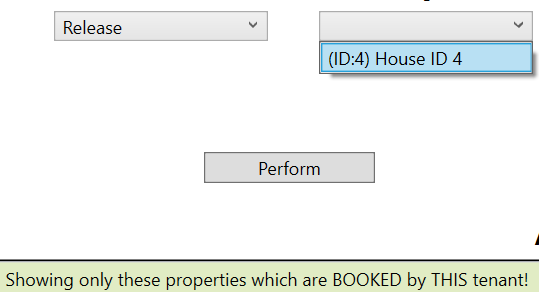
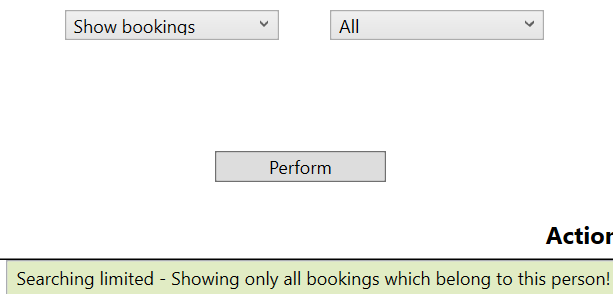
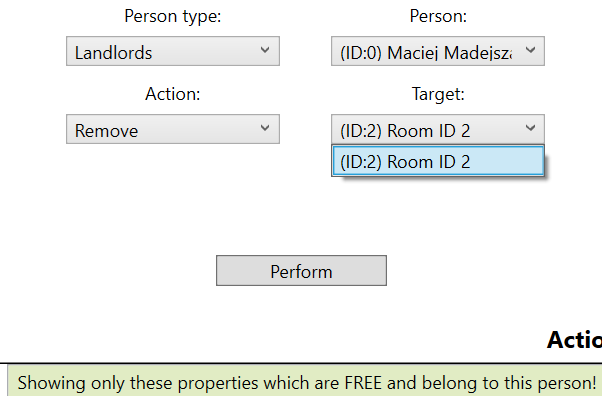
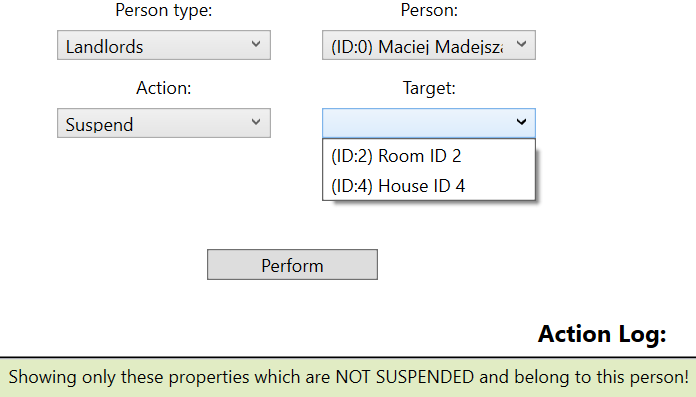
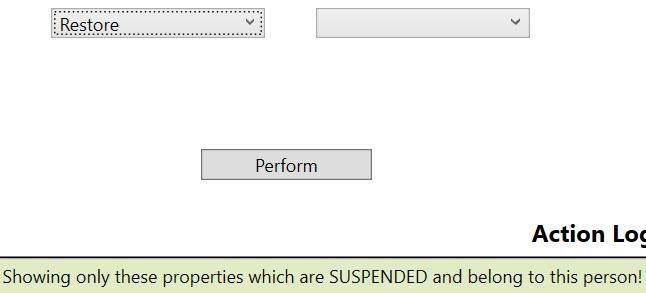
**3x**

**Scrolled**

**Advanced requirements (2.1 and above) The above plus...**

• User can select a bookable item and an action is performed

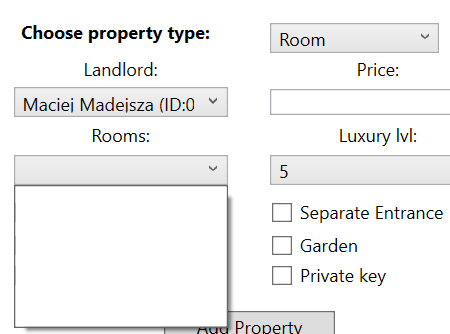
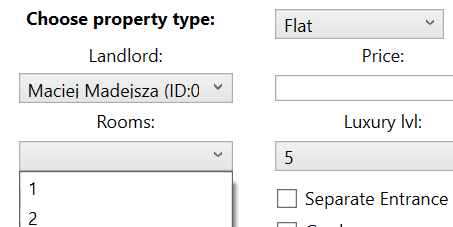
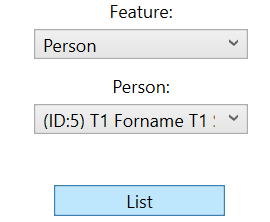
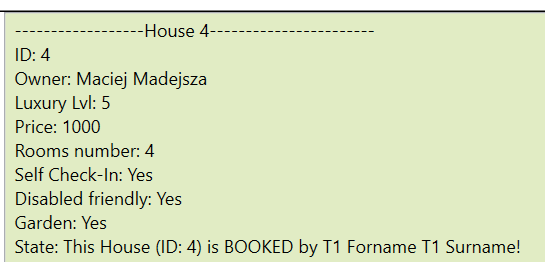
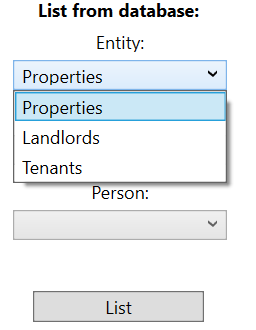
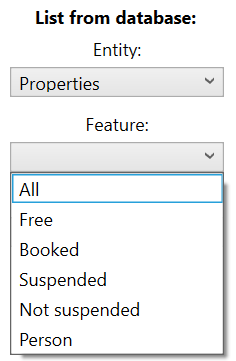
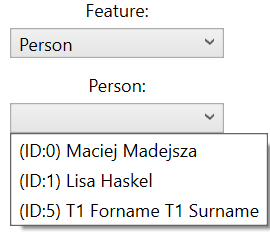
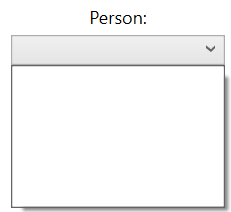
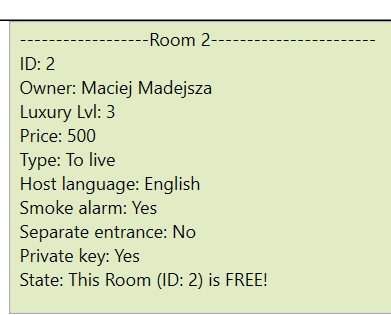
• Advanced features of your choice:

* User - there are 3 types of users with different privileges **in code behind**: Admin, Landlord and Tenant. To keep interface reasonable simple for assessment – only Landlord and Tenant are pulled out to interface in ‘functions panel’
* Design patterns used:
* State pattern: to achieve smooth state instances of the properties and to be able to flexibly implement booking and **suspension features (extra).**
* Booking, **releasing, and suspending change property state**

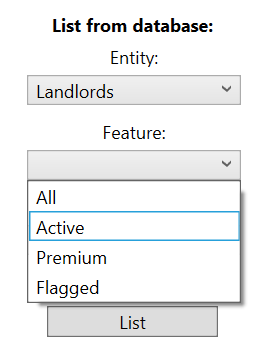
Nothing suspended yet

May be booked (4)

Tenant

* Singleton: to achieve 1 common database and monitor the occurrence of objects. That allowed to implement **listing of particular landlord or tenant properties**. It allowed to populate conditionally drop lists based on chosen person or property to allow/restrict different actions.

Scroll down



The same features set for Landlords and Tenants.

Person drop list deactivated

* Various validation – based on conditional access to different actions based on user and property type + simple inputs validation.