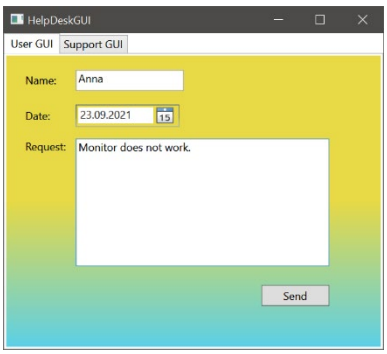
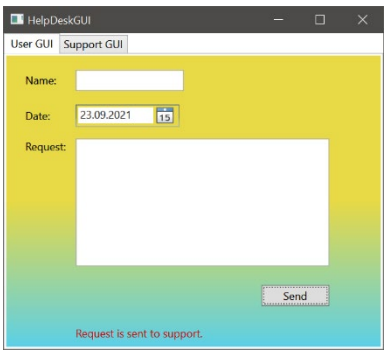
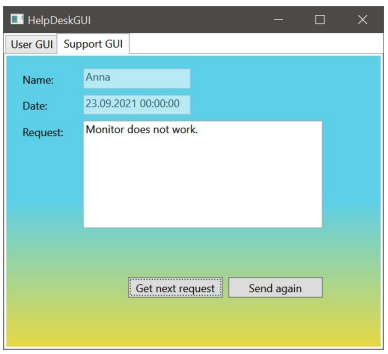
Entwickle mit C# und WPF eine HelpDesk Anwendung. Diese besteht aus zwei Teilen. Einerseits können BenutzerInnen beim HelpDesk einen HelpRequest erstellen und damit technische Hilfe anfordern, andererseits arbeiten die MitarbeiterInnen des HelpDesk-Centers die Liste der HelpRequests der Reihe nach ab.







Schritt 1: Die Klasse **HelpRequest**

Erstelle die Klasse in einer Klassenbibliothek.
Ein Objekt der Klasse **HelpRequest** besteht aus einem **Namen** (`string`) der Person, die die Anfrage stellt. Ein **Date** (`DateTime`) wann der Antrag gestellt wurde und einem **Request** (`string`) in dem das Problem beschrieben werden kann. Die Klasse hat einen Konstruktor, der die Werte für alle Properties als Parameter übergeben bekommt.

Erstelle eine `string` Property CSV. Beim `set` wird `value` dazu verwendet um die Properties Name, Date und Request zu setzten. Der `get` liefert die csv-Darstellung (mit den Werten Name, Date und Request) des Objekts.

Implementiere die Properties so, dass **Name** und **Date** nicht leer sein dürfen. Sollte dies der Fall sein, soll eine `ArgumentNullException` geworfen werden.

Schritt 2: Die WPF

Um die zwei Bereiche der Anwendung getrennt darstellen zu können, verwenden wir zwei TabControls, wie man oben in der Grafik sehen kann. Eines für die BenutzerInnen, die eine Request erstellen können und eines für die HelpDesk-MitarbeiterInnen, die die Requests der Reihe nach bearbeiten.

Wichtig: durch die TabControls sieht die Anwendung aus, als hätten wir zwei getrennte Fenster. Das ist aber nicht der Fall. Deshalb müssen die einzelnen WPF-Elemente (TextBox, Label, Button, etc.) so benannt werden, dass der Name im ganzen Fenster, also in beiden Tabs, eindeutig ist. Auch der C#-Code (und deshalb auch alle Felder) ist für beide Tabs in einer Klasse (in *MainWindow.xaml.cs*) beschrieben.

Meldungen an die BenutzerInnen bzw. an die HelpDesk-MitarbeiterInnen werden in einem Label im Fenster unten angezeigt.

Schritt 3: Die Implementierung in MainWindow

Die HelpRequests werden in einer **Queue** verwaltet. Hier werden die neuen HelpRequest-Objekte der BenutzerInnen gespeichert (Button: *Send*) und aus dieser Queue werden die HelpRequest-Objekte der zeitlichen Reihenfolge entsprechend (FIFO-Prinzip: First in – First out), von den HelpDesk-MitarbeiterInnen entnommen (Button: *Get next Request*).

Hinweis: Der Wert aus dem **DatePicker** kann wie folgt verwendet werden:

```
DateTime dt =  (DateTime)datePickerRequest.SelectedDate;  // get the value from date picker
datePickerRequest.SelectedDate = DateTime.Now;                // set the value for date picker
```

UserRequests

- Mit dem Button send wird die Eingabe in ein neues HelpRequest-Objekt gespeichert und in die Collection eingefügt.

HelpDesk Support

- Mit dem Button *Get next request* wird aus der Warteschlange das HelpRequest-Objekt genommen, das als nächstes bearbeitet werden soll. Dieses Objekt soll aus der Collection auch sofort gelöscht werden, damit nicht versehentlich eine andere Support-MitarbeiterIn (in einer Mehrbenutzerumgebung) an dem gleichen Problem zu arbeiten beginnt.
- Für den Fall, dass das Problem nicht sofort gelöst werden kann (z.B. fehlende Ersatzteile, etc.), kann der Request-Text erweitert, und mit dem Button *Send again* erneut in die Warteschlange gestellt werden.

Schritt 4: Daten permanent speichern

Erstelle eine **statische** Klasse **`DataManager`**, die dazu dient, die Requests permanent (also auf die Festplatte) zu speichern bzw. die gespeicherten Daten aus einer Datei zu lesen.

Methoden:

- `WriteAllRequests` – schreibt alle HelpRequest-Objekte aus der Queue, die als Parameter mitgegeben wurde, in eine .csv-Datei. Der Pfad zur Datei, wird ebenfalls als Parameter mitgegeben. Erstelle aus der Queue-Parameter eine List<string>, in der du die csv-Darstellung der Request-Objekte speicherst. Schreibe dann diese Liste in die csv-Datei.
- `ReadAllRequests` – bekommt den Pfad zur csv-Datei als Parameter und liest die Daten Zeile für Zeile. Jede Zeile enthält die Daten für ein HelpRequest-Objekt. Der Rückgabewert der Methode ist eine Queue, in der die eingelesenen HelpRequest-Objekte gespeichert wurden.

Daten in MainWindow:

- Im Konstruktor von MainWindow sollen die Daten aus der .csv-Datei geladen werden.
- Beim Schließen von MainWindow (Closing) sollen die Request-Objekte auf die Festplatte gespeichert werden.

Gutes Gelingen!