* Dürfen neue Querys nur abgesetzt werden wenn ein Editorfenster geschlossen wurde? (wie garantiere ich Konsistente Daten)

Design Patterns

* Schichtenmodel: eine Schicht kann nur den Dienst ihres unteren Nachbarn nutzen, sie darf keine Schichten überspringen
* Abbildung der Ideen auf Pakete, Klassen & Schnittstellen
* Klassen sollten nicht zu viel über andere Klassen wissen
* InterClass Kommunikation über wohldefinierte Schnittstellen so dass Änderungen mgl.

Zu klärende Fragen:

* Threads? Damit z.B. nach absetzten eines Querys weiter im Programm navigiert/gearbeitet werden kann auch wenn die Ausführung des Querys lange dauert
* Eigene Datenstruktur für das MySql ResultSet? Damit behalten wir Flexibilität der Datenbankversion / -firma gegenüber, da das jeweilige ResultSet Seviertfertig in unser Format übersetzt wird. Diese Übersetzungsschnittstelle müsste dann also nur ausgetauscht werden.

Konzepte:

* Database Klasse ist nach Singleton- und Observer-Pattern konstruiert (es gibt nur eine instanz, und diese wird von den Models überwacht), die Klasse nutzt eine Schnittstelle zum ansprechen der verschiedenen DB Ausprägungen

**DB Grundlage** (Rick)

* Schnittstellen definieren

**GUI** (SEPP,JAKOB)

* Welche Daten werden pro Bereich benötigt? 🡪 müssen diese aufbereitet werden
* 🡪 UseCase als Grundlage
* Über MVC belesen, werden Threads im GUI Controller gestartet
* Wie wird dem Nutzer symbolisiert das seine Aktion in Bearbeitung ist, das Programm darf nicht blockieren
* Gestaltung Ivo
* Escapetasten Funktionalität

**Import** (Sepp)

* Zuordnung von .xls zu Db Struktur

**Export / Druck** (Ivo)

* Schnittstelle definieren
* Vorlagen?

**Mail** (Jakob)

* Schnittstelle definieren
* Vorlagen?

**Suche** (Rick)

* 🡪 Funktionsweise
* Autovervollständigung
* Filter ?