Internetserver-programmierung

Projektbericht im Rahmen der Portfolio-Prüfung SoSe 2020

**2. Zeitraum: 06.07.2020 – 02.08.2020**

**Gruppe 16**

**Gruppenmitglied: Felicitas Yeboah**

**Gruppenmitglied: Christian Till**

1. **Spezifikation**

Unsere Anwendung verwaltet Termine in einem Kalender. Der Anwender kann neue Termine anlegen.

Termine können einzelnen Kategorien mit unterschiedlichen Farben zugeordnet und bei Bedarf können individuelle Kategorien angelegt werden. Der Terminkalender beinhaltet eine Monatsansicht und eine Wochenansicht. Termine können an der Kalenderoberfläche nach Kategorien gefiltert und angezeigt werden. Es können neue Kategorien angelegt und bestehende Kategorien bearbeitet und gelöscht werden. Termine können ebenfalls bearbeitet und gelöscht werden.

* 1. **Benutzergruppen**

Typische Benutzer sind Personen, die Termine zentral verwalten möchten. Unsere Anwendung Terminkalender kann von jeder Benutzergruppen verwendet werden, die seine Termine zentral verwalten möchte. Da Termin unterschiedlichen Kategorien zugeordnet werden können und bei Bedarf ebenfalls individuelle Kategorien erstellt werden können, ist die Anwendung sehr universell.

* 1. **Funktionale und nicht-funktionale Anforderungen**

Im Laufe der Projektarbeit wurde ein Kalender angefertigt, zu welchem Termine erfasst, bearbeitet und gelöscht werden können. Die Benutzereingaben der einzelnen Termine werden mit Regex-geprüft. Die Oberfläche besteht aus der Kalender-Ansicht mit den einzelnen Terminen, sowie einer Bearbeitungsseite für Termine. Die Kalenderansicht ist auswählbar für Wochenansicht und Monatsansicht.

Weiterhin können die einzelnen Termine einer Kategorie zugeordnet werden. Unterschiedliche Kategorien werden in unterschiedlichen Farben an der Oberfläche dargestellt. Zusätzlich ist es mittels eines Filters möglich, nur Termine einer bestimmten Kategorie an der Oberfläche anzuzeigen.

Beim Start der Anwendung sind per Default bereits acht verschiedene Kategorien vorhanden, aus welchen der Anwender auswählen.

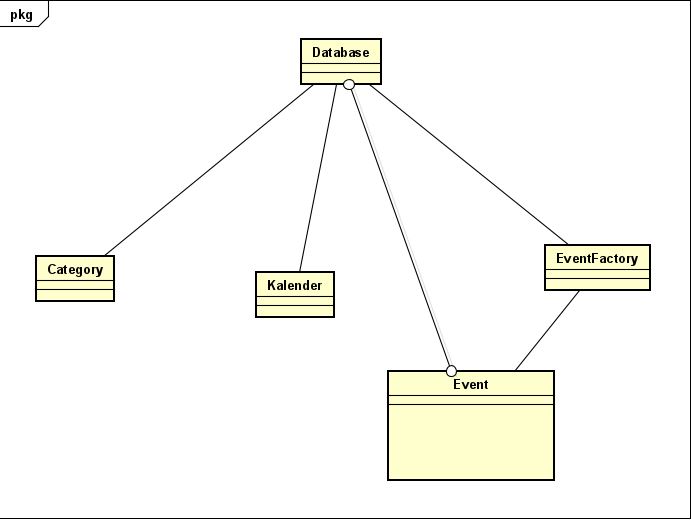
Eine eigene Kategorie inklusive Farbangabe ist bei der Erstellung eines Termins möglich.

Des Weiteren können in einer Kategoriebearbeitungsmaske Kategorien geändert, gelöscht und neu erstellt werden. Kategorien können individuellen Farben zugeordnet werden.

Einzelne Termine können über Detailansicht bearbeitet und gelöscht werden.

Auf die Umsetzung von Terminen in der Tagesansicht mussten wir leider verzichten, da das Anzeigen bzw. Auswerten von minutengenauen Terminen aus Zeitgründen nicht umgesetzt werden konnte. Eigentlich sollte eine Tagesübersicht angezeigt werden, auf der alle Termine des Tages.

1. **Entwurf**
   1. **Klassendiagramm**



Die Klasse „Database“ regelt den reinen Datenbankzugriff. Im Konstruktor wird die Verbindung hergestellt und eine Instanz der Klasse wird in einer globalen Variable gespeichert, welche von anderen Klassen genutzt werden kann. Die Klasse stellt grundlegende Operationen bereit, um mit der Datenbank physisch zu interagieren.

Des Weiteren gibt es die Klasse „Kalender“, welches die einzelnen Ansichten, wie Monats- und Wochenansicht, erstellt und als HTML-Code zurück gibt.

In den einzelnen Ansichten sollen auch die Termine in der gewählten Zeitspanne angezeigt werden. Dazu wird auf Terminobjekte der Klasse „Event“ zurückgegriffen.

Diese Event-Objekte stellt sämtliche Funktionen bereit, die sowohl der Kalender zur Anzeige benötigt (z.B. Ausgabe des HTML-Codes zur Darstellung) als auch diejenigen, die zur Kommunikation mit der Datenbank benötigt werden (Löschen genau dieses Objekts aus der Datenbank, Ändern…, usw.).

Da eine der funktionalen Anforderungen das Hinzufügen einer Kategorie zu einem Termin war und eine Kategorie potenziell zu mehreren Terminen gehören kann, gibt es auch hierfür eine eigene Klasse „Category“. In der Datenbank hat jede Kategorie eine eindeutige ID. Die Zuordnung einer Kategorie zum Termin erfolgt über diese ID. Die ID der Kategorie wird in einem Feld der Tabelle Termin gespeichert.

Die Klasse „Category“ enthält Funktionen zur Kommunikation mit der Datenbank.

Da das Erstellen der Termin-Objekte auf unterschiedliche Arten erfolgen kann, z.B. über unterschiedliche Suchprofile auf der Datenbank oder auch als neues Objekt, noch ohne Datenbankbezug, haben wir das Entwurfsmuster Factory eingesetzt und eine Klasse „EventFactory“ erstellt. Diese Klasse implementiert mehrere Methoden zur Erzeugung von Termin-Objekten, abhängig von der jeweiligen Anforderung an diese Objekte.

* 1. **Entwurfsmustereinsatz**

**Observer**

Wenn auf der Bearbeitungsseite einer Kategorie ebendiese gelöscht wird, wird ein Update an alle Events, welche dieser Kategorie angehören, gesendet und aus den bestehenden Datensätzen entfernt.

Eine weitere Möglichkeit wäre es gewesen, das Observer-Pattern beim Löschen eines Events einer Gruppe zu benutzen. Allerdings erschien es uns einfacher hierfür die Datensätze mittels der Gruppen-ID auf der Datenbank direkt zu löschen.

**Fabrik**

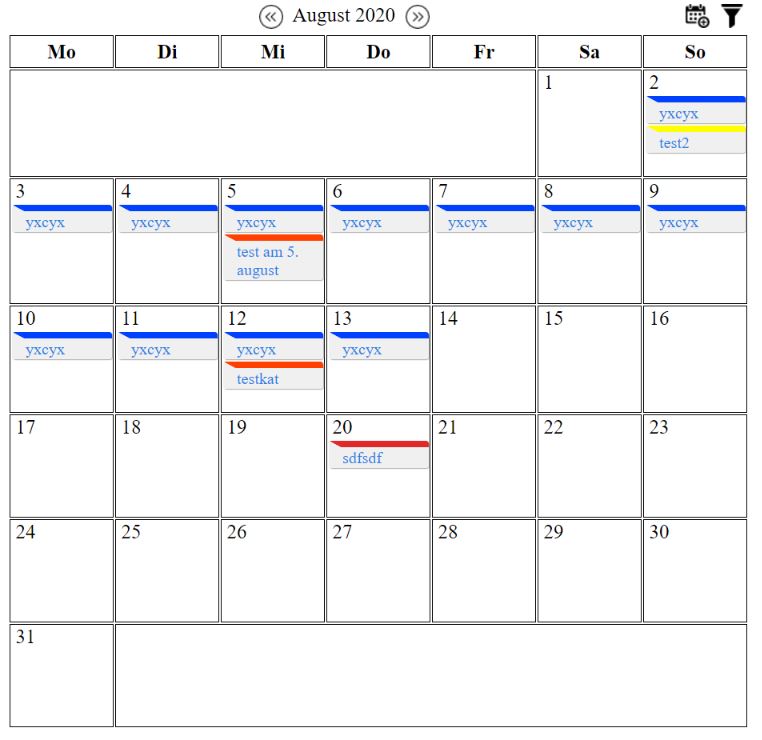
Terminobjekte werden mittels einer Factory-Klasse erstellt. Die Factory-Klasse stellt verschiedene Methoden bereit, womit Termin-Objekte erzeugt werden können. Wobei diese Aussage rein technisch nicht komplett richtig ist.

Die Erzeugung der Objekte geschieht eigentlich mittels der Schnittstelle der PHP Data Objects (PDO). Die Factory-Klasse bietet Methoden an, welche dann mit einem Suchprofil die Datenbank (über die Datenbank-Klasse) kontaktiert. Die Datenbank-Klasse erstellt über die Option „FetchClass“ aus den gefunden Datensätzen automatisch Objekte der Klasse „Event“ und gibt diese an die Factory-Klasse zurück.

Es gibt allerdings auch eine Methode, welche ein neues Terminobjekt erzeugt. Diese wird genutzt, wenn ein neuer Termin angelegt wird oder ein bestehender Termin bearbeitet wird (bevor er in die Datenbank geschrieben wird).

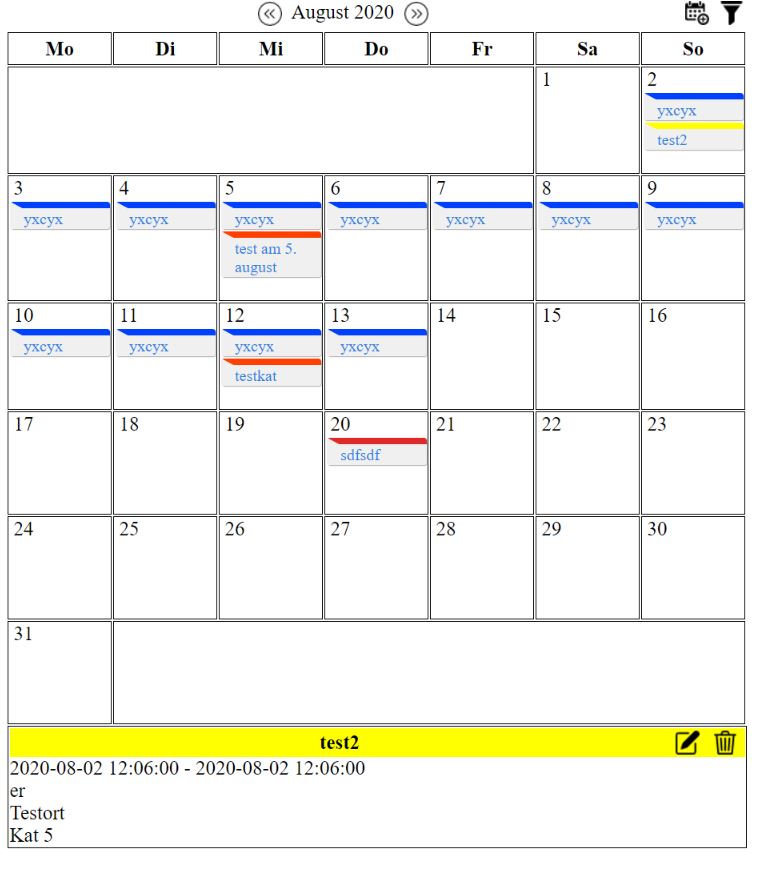
1. **Implementierung**
   1. **Screenshots**

**Monatsansicht**



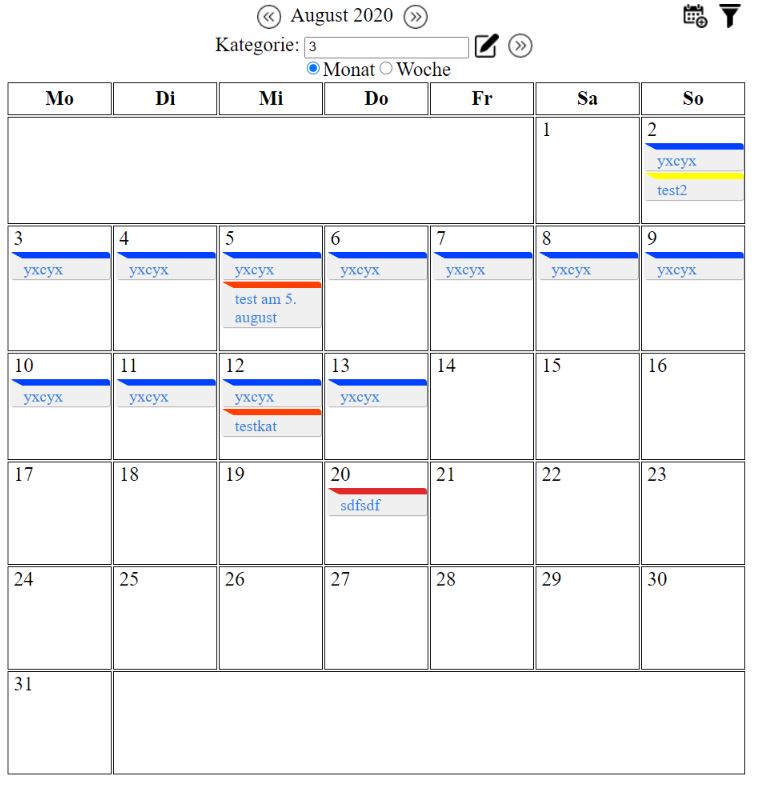
Monatsansicht mit eingetragenen Terminen unterschiedlicher Kategorien, welche anhand der Farbgebung unterschieden werden können. Im Kopfbereich können die Monate durchgeschaltet, ein neuer Termin angelegt oder eine Leiste mit Filteroptionen angezeigt werden. Ein Klick auf einen Termin öffnet eine Details-Ansicht.

**Detailansicht**:



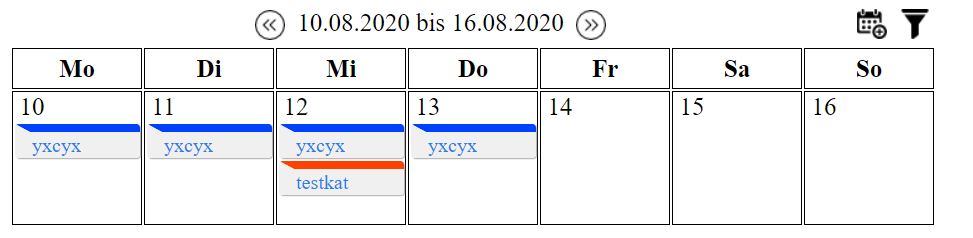
Unter dem Kalender wird die Detailansicht des angeklickten Termins angezeigt. Im Kopfbereich der Detailansicht gibt es die Möglichkeit den Termin zu löschen oder zu bearbeiten.

**Filter Kategorie, Monats-/Wochenansicht::**

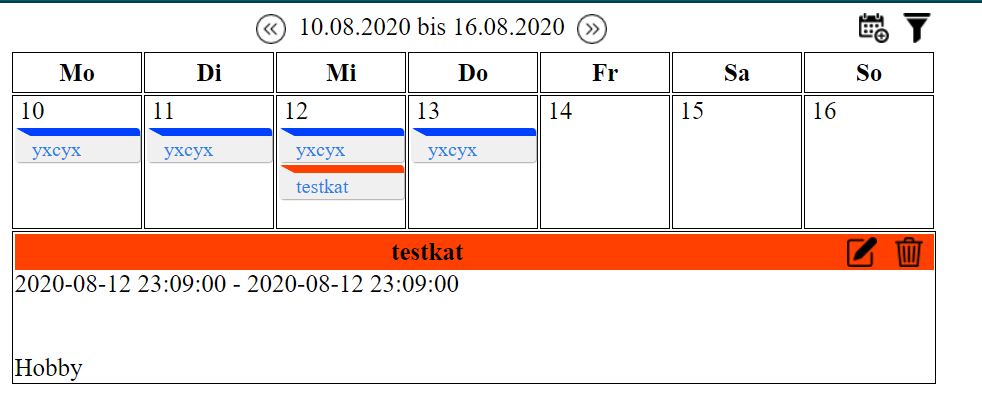


Eingeblendete Filterleiste mit der Möglichkeit die Wochenansicht zu aktivieren oder nur Termine einer bestimmten Kategorie anzuzeigen. Ein erneuter Klick auf das Filtersymbol blendet die Leiste wieder aus.

**Wochenansicht:**

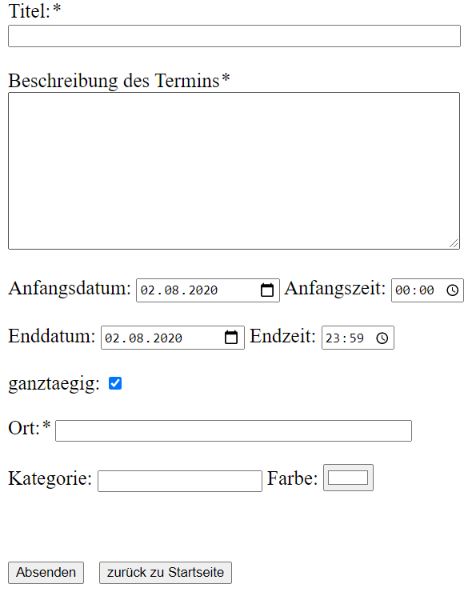


**Wochenansicht mit Termin Details:**



Wochenansicht des Kalenders. Die Steuerung unterscheidet sich nicht von der Monatsansicht.

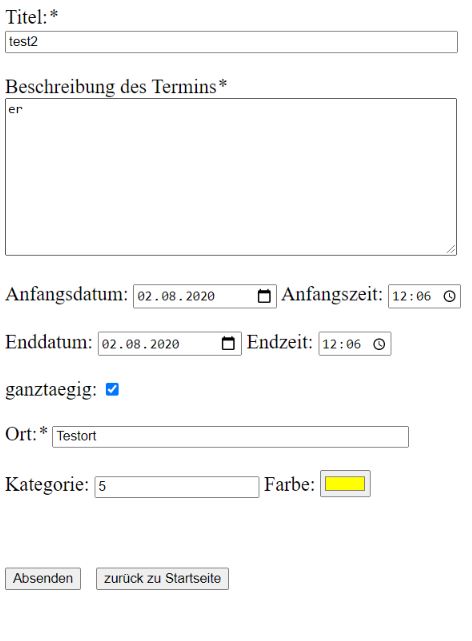
**Neuen Termin anlegen:**



Über das Icon zum Termin anlegen gelangt man in die oben gezeigte Ansicht. Die Daten des Termins können eingegeben werden und werden RegEx geprüft.

Eine Besonderheit stellt das Feld Kategorie dar. Hier werden vorhandene Kategorien in einer DropDown-Liste angezeigt und können ausgewählt werden. Es kann aber auch eine neue Kategorie eingegeben werden. Diese wird beim Speichern des Termins ebenfalls in der Datenbank abgelegt.

**Termin bearbeiten:**

****

Die Ansicht zum Bearbeiten eines Termins wird mit den Daten des ausgewählten Termins vorbesetzt.

**Kategorie bearbeiten und neu anlegen:**

****

Die Änderung einer bestehenden Kategorie kann nach Auswahl im Filter-Feld erfolgen. Es kann jeweils der Name und die Farbe geändert werden. Auch kann die Kategorie gelöscht werden. Dies hat automatisch zur Folge, dass die Kategorie auch aus den Terminen entfernt wird, die diese referenzieren (Entwurfsmuster Observer).

* 1. **Genutzte Frameworks**

Wir haben auf die Verwendung von Frameworks verzichtet.

1. **Zusammenfassung**
   1. **Aufgabenteilung**

**Christian:**

* Implementierung Entwurfsmuster
* Filterfunktion
* Wochenansicht des Kalenders
* Termine anzeigen in der Kalenderoberfläche
* Detailansicht Termine
* Ajax für zeigeEvent
* Projektdokumentation

**Felicitas**:

* Monatsansicht des Kalenders
* Termin hinzufügen Seite/Termin bearbeiten Seite
* Kategorien bearbeiten-Seite / Kategorie hinzufügen Seite
* Ajax calViews, Ajax Color
* Klassendiagramm
* Projektdokumentation

Der Rest wurde gemeinsam bearbeitet.

* 1. **Ausblick**

Zu Beginn des Projekts haben wir uns die Datenbanktabellen und benötigten Klassen überlegt. Diese haben wir erstmal nur oberflächlich auf die Anforderungen abgestimmt. Dies führte in der Folge dazu, dass wir Probleme hatten spezielle Szenarien umzusetzen bzw. diese konnten nur mit erheblichem Mehraufwand in das Programm eingebracht werden (z.B. Darstellung von Terminen eines längeren Zeitraums.

Ebenso haben wir uns ein Entwurfsmuster herausgesucht (Data Mapper), welches DTO-Objekte für der Termine in Kombination mit der Kategorie erschaffen soll. Da dieses Entwurfsmuster nicht im Skript beschrieben ist, sollten wir dieses nicht verwenden. Daher haben wir wiederum Zeit verloren bei der Implementierung eines gültigen Entwurfsmusters. Die Projektanforderungen müssen bei Unklarheiten, am besten vor Projektbeginn, geklärt werden.

Des weiteren ist es sinnvoll, Funktion sofort zu kommentieren und die Struktur des Codes (auch bei Tests) direkt anzupassen. Man könnte einen Testbereich in Klassen schaffen, welche entsprechende Funktionen enthält. Dieses werden dann im Quellcode verschoben, vervollständigt und kommentiert, sobald die korrekte Funktion sichergestellt ist.

Des weiteren ist es nicht sinnvoll, die Dateien, welche Klassen enthalten mit einem direkten Aufruf anzusprechen und dort entsprechende Auswertungen vorzunehmen. Durch die vielen Includes kam es dadurch zu schwerwiegenden Fehlern und Seiteneffekten, die nur sehr schwer ausfindig zu machen sind. In zukünftigen Projekten ist hier eine klare Trennung nach Klassen/Objekten und Prozessen vorzunehmen.