

PFLICHTENHEFT

Das Klassenverwaltungssystem



24.04.2022
IMS I2A
Modularbeit 326

Inhalt

Modulararbeit 326 Klassenverwaltungssystem	2
Einleitung:	2
Zielbestimmung:.....	2
Musskriterien:	2
Wunschkriterien	2
Produkteinsatz:	2
Anwendungsbereich.....	2
Zielgruppe:	2
Betriebsbedingungen:	2
Produktübersicht:.....	3
Produktfunktionen:	4
Musskriterien Use Cases 1-4 Schüler:.....	4
Wunschkriterien Use Cases 5-16 Klasse, Einträge, Login:.....	8
Produktdaten:	15
Alle Attribute für die Musskriterien:.....	15
Alle Attribute für die Wunschkriterien:	15
Benutzungsoberfläche:	16
Technische Produktumgebung:.....	17
Software:	17
Hardware:	17
Orgware:	17
Spezielle Anforderungen an die Entwicklungsumgebung:.....	17
Hardware:	17
Software:	17
Orgware:	17
Ergänzungen:.....	17

Modularbeit 326 Klassenverwaltungssystem

Einleitung:

Im Rahmen dieser Modularbeit soll eine von mir gewählte Software entstehen.

Ich habe mich dazu entschieden, eine Klassenverwaltungssystem zu planen und später dieses auch zu programmieren, beziehungsweise die einzelne Use Cases zu implementieren.

Zielbestimmung:

Musskriterien:

- Der Benutzer soll ein Schüler erstellen können.
- Der Benutzer soll ein Schüler ansehen können.
- Der Benutzer soll ein Schüler bearbeiten können.
- Der Benutzer soll ein Schüler löschen können.

Wunschkriterien:

- Der Benutzer muss sich zuerst Authentifizieren, bevor Aktionen getätigt werden können, dies dient dem zusätzlichen Schutz der Daten.
- Der Benutzer kann dem Schüler eine Absenz geben.
- Der Benutzer kann eine Absenz von dem Schüler entfernen.
- Der Benutzer soll eine Klasse erstellen können.
- Der Benutzer soll die erstellten Schüler, einer Klasse zuweisen können.
- Das Usability soll ermöglichen, dass der Benutzer die Software ohne Instruktion nutzen kann.

Produkteinsatz:

Anwendungsbereich:

Bei dem Produkt handelt es sich um eine Verwaltungssoftware.

Zielgruppe:

Die Software ist ausschliesslich für Mitarbeiter der Schule gedacht.

Ziel ist es den Administrativen Aufwand zu verkürzen und damit die Lehrpersonen, insbesondere das Sekretariat zu entlasten.

Betriebsbedingungen:

Das Produkt ist mit der IT-Infrastruktur der Schule verbunden und kann ausschliesslich in dem internen Lehrernetzwerk der Schule genutzt werden können.

Das Produkt soll immer in der Schule zur Verfügung stehen.

Produktübersicht:



Produktfunktionen:

Musskriterien Use Cases 1-4 Schüler:

Use Case 1; Schüler erstellen:

Ziel:

Eine Möglichkeit den Akteuren bieten, ein neuer Datenbankeintrag zu erstellen, mit den [unten](#) genannten Attributen.

Fokus bei der Erreichung dieses Ziel ist nicht nur eine performante Lösung, sondern dass auch Wert auf die Usability gelegt wird.

Beschreibung:

Der Nutzer füllt Pflichtfelder aus, Daten des Formulars werden gefiltert, sowie server- und clientseitig überprüft und dann in die Datenbank geschrieben.

Schlägt die Validierung fehl, wird dies dem Nutzer mitgeteilt.

Vorbedingung:

Der Akteur muss in dem gleichen Netzwerk sein, wie der Webserver, sowie sollte der Port 80, nicht freigegeben werden. Es muss eine aktive Verbindung zur DB bestehen.

Nachbedingung:

Die serverseitige Validierung sowie die clientseitige Validierung überprüfen die Formfelder, nur dann dürfen die Daten in die DB eingetragen werden.

Auslösendes Ereignis:

Nutzer betätigt "Senden" Button.

Kategorie:

Primär

Alternative:

Nutzer klickt auf Abbrechen, danach wird der Nutzer zur Hauptseite weitergeleitet.

Use Case 2; Schüler ansehen:

Ziel:

Ein System, dass es ermöglicht, automatisiert die Daten der DB abzurufen und diese Daten dann in einer Web-Page Tabelle anzuzeigen.

Zu jedem generierten Element sollen jeweils zwei Button dazu generiert werden, diese Buttons sollen dem Akteur die Möglichkeit bieten, einen Schüler Eintrag zu bearbeiten oder zu löschen.

Beschreibung:

Die automatisch erstellte Tabelle und die dazugehörigen Buttons, werden in zwei Funktionen gegliedert. Eine Funktion, die Einträge in die Tabelle einträgt, sowie dann der zweiten Funktion die ID des Eintrages übergibt.

Diese Funktion erstellt dann zu jedem Element zwei dazugehöriger Buttons.

Diese Buttons beinhalten ein GET-Request zu der Webpage «manageStudent», zusammen mit der dazugehörigen ID.

Vorbedingung:

Akteur muss in dem gleichen Netzwerk sein wie der Webserver sowie der Port 80 sollte nicht freigegeben werden.

Es muss eine aktive Verbindung zur DB bestehen.

Nachbedingung:

Es müssen Einträge vorhanden sein.

Auslösendes Ereignis:

Durch das Aufrufen der Schülerliste.

Kategorie:

Primär

Alternativen:

Falls keine Einträge vorhanden sind, wird dies dem Akteur mitgeteilt.

Use Case 3; Schüler bearbeiten:

Ziel:

Ein Schülereintrag sollte von dem Akteur bearbeitet werden können.

Beschreibung:

Eine ID wird unter der Variable mit dem Namen «EDIT» per GET-Request an «manageStudent» gesendet. «manageStudent» kontrolliert nun, ob die ID in der DB vorhanden ist, falls ja werden die Formfelder, die für das erstellen eines Schülers verantwortlichen sind, mit den Daten der DB geladen. Wird das Formfeld nun abgesendet und besteht die Validierung, werden die Daten in der DB aktualisiert.

Vorbedingung:

Akteur muss in dem gleichen Netzwerk sein wie der Webserver und der Port 80 sollte nicht freigegeben werden. Es muss eine aktive Verbindung zur DB bestehen.

Nachbedingung:

Die ID muss existieren.

Auslösendes Ereignis:

Durch das Betätigen eines editiere Button auf der Schüler ansehen Seite.

Kategorie:

Sekundär

Alternative:

Nutzer hat ungültige ID eingegeben -> Fehlermeldung auf der Seite.

Use Case 4; Schüler löschen:

Ziel:

Ein Eintrag sollte von dem Akteur gelöscht werden können.

Beschreibung:

Eine ID wird unter der Variabel mit dem Namen «DEL» per GET-Request an «manageStudent» gesendet. «manageStudent» kontrolliert nun, ob die ID in der DB vorhanden ist, falls ja wird der Eintrag gelöscht, falls nicht wird eine Fehlermeldung dem Nutzer zurückgegeben.

Vorbedingung:

Akteur muss in dem gleichen Netzwerk sein wie der Webserver und der Port 80 sollte nicht freigegeben werden. Es muss eine aktive Verbindung zur DB bestehen.

Nachbedingung:

Die ID muss existieren.

Auslösendes Ereignis:

Durch das betätigen eines Löschens Button auf der Startseite.

Kategorie:

Sekundär

Alternative:

Nutzer hat ungültige ID eingegeben -> Fehlermeldung auf der Seite.

Wunschkriterien Use Cases 5-16 Klasse, Einträge, Login:

Use Case 5; Klasse erstellen:

Ziel:

Eine Möglichkeit den Akteuren bieten, ein neuer Datenbankeintrag zu erstellen, mit den [unten](#) genannten Attributen.

Fokus bei der Erreichung dieses Ziel ist nicht nur eine performante Lösung, sondern dass auch Wert auf die Usability gelegt wird.

Beschreibung:

Die Web-Page heisst hier «manageClass».

Der Nutzer füllt Pflichtfelder aus, Daten des Formulars werden gefiltert, sowie server- und clientseitig überprüft und dann in die Datenbank geschrieben.

Schlägt die Validierung fehl, wird dies dem Nutzer mitgeteilt.

Vorbedingung:

Der Akteur muss in dem gleichen Netzwerk sein, wie der Webserver, sowie sollte der Port 80, nicht freigegeben werden. Es muss eine aktive Verbindung zur DB bestehen.

Nachbedingung:

Die serverseitige Validierung sowie die clientseitige Validierung überprüfen die Formfelder, nur dann dürfen die Daten in die DB eingetragen werden.

Auslösendes Ereignis:

Nutzer betätigt "Senden" Button.

Kategorie:

Primär

Use Case 6; Klasse ansehen:

Ziel:

Ein System, dass es ermöglicht, automatisiert die Daten der DB abzurufen und diese Daten dann in einer Web-Page Tabelle anzuzeigen. Zu jedem generiertem Element, sollen jeweils zwei Button dazu generiert werden. Diese Buttons sollen dem Akteur die Möglichkeit bieten, eine Klasseneintrag zu bearbeiten oder zu löschen.

Beschreibung:

Die automatisch erstellte Tabelle und die dazugehörigen Buttons, werden in zwei Funktionen gegliedert. Eine Funktion, die Einträge in die Tabelle einträgt, sowie dann der zweiten Funktion die ID des Eintrages übergibt.

Diese Funktion erstellt dann zu jedem Element zwei dazugehöriger Buttons. Diese Buttons beinhalten ein GET-Request zu der Webpage «mangeClass», zusammen mit der dazugehörigen ID. Beim Löschen heisst die GET Variable «DEL» und beim Editieren heisst die GET Variable «EDIT».

Vorbedingung:

Akteur muss in dem gleichen Netzwerk sein wie der Webserver und der Port 80 sollte nicht freigegeben werden.

Es muss eine aktive Verbindung zur DB bestehen.

Nachbedingung:

Es müssen Einträge vorhanden sein.

Auslösendes Ereignis:

Durch das Aufrufen der Klassen ansehen Seite.

Kategorie:

Primär

Alternativen:

Falls keine Einträge vorhanden sind, wird dies dem Akteur mitgeteilt.

Use Case 7; Klasse bearbeiten:

Ziel:

Ein Klasseneintrag sollte von dem Akteur bearbeitet werden können.

Beschreibung:

Eine ID wird unter der Variable mit dem Namen «EDIT» per GET-Request an «manageClass» gesendet. «manageClass» kontrolliert nun, ob die ID in der DB vorhanden ist, falls ja werden die Formfelder, die für das Erstellen einer Klasse verantwortlichen sind, mit den Daten der DB geladen. Wird das Formfeld nun abgesendet und besteht die Validierung, werden die Daten in der DB aktualisiert.

Vorbedingung:

Akteur muss in dem gleichen Netzwerk sein wie der Webserver und der Port 80 sollte nicht freigegeben werden. Es muss eine aktive Verbindung zur DB bestehen.

Nachbedingung:

Die ID muss existieren.

Auslösendes Ereignis:

Durch das Betätigen eines editiere Button auf der Klassen ansehen Seite.

Kategorie:

Sekundär

Use Case 8; Klasse löschen:

Ziel:

Ein Klasseneintrag sollte von dem Akteur gelöscht werden können.

Beschreibung:

Eine ID wird unter der Variable mit dem Namen «DEL» per GET-Request an «mangeClass» gesendet. «mangeClass» kontrolliert nun, ob die ID in der DB vorhanden ist, falls ja wird der Eintrag gelöscht, falls nicht wird eine Fehlermeldung dem Nutzer zurückgegeben.

Vorbedingung:

Akteur muss in dem gleichen Netzwerk sein wie der Webserver und sowie der Port 80 sollte nicht freigegeben werden. Es muss eine aktive Verbindung zur DB bestehen.

Nachbedingung:

Die ID muss existieren.

Auslösendes Ereignis:

Durch das Betätigen eines Löschens Button auf der Klassen ansehen Seite.

Kategorie:

Sekundär

Use Case 9; Eintrag erstellen:

Ziel:

Es soll den Akteuren bieten, ein neuer Datenbankeintrag zu einem Schüler zu erstellen, mit den unten genannten Attributen.

In diesem Dateneintrag sollte es möglich sein eine Absenzen oder Verspätungen zu erfassen oder einen eigenen Eintrag zu erstellen.

Fokus bei der Erreichung dieses Ziel ist nicht nur eine performante Lösung, sondern dass auch Wert auf die Usability gelegt wird.

Beschreibung:

Der Nutzer füllt Pflichtfelder aus, Daten des Formulars werden gefiltert, sowie server- und clientseitig überprüft und dann in die Datenbank geschrieben.

Schlägt die Validierung fehl, wird dies dem Nutzer mitgeteilt.

Vorbedingung:

Der Akteur muss in dem gleichen Netzwerk sein, wie der Webserver, sowie sollte der Port 80, nicht freigegeben werden. Es muss eine aktive Verbindung zur DB bestehen.

Nachbedingung:

Die serverseitige Validierung sowie die clientseitige Validierung überprüfen die Formfelder, nur dann dürfen die Daten in die DB eingetragen werden.

Auslösendes Ereignis:

Nutzer betätigt "Senden" Button.

Kategorie:

Primär

Use Case 10; Eintrag ansehen:

Ziel:

Ziel ist es, dass der Akteur die Einträge eines bestimmten Schülers aufrufen kann, so dass alle Einträge eines bestimmten Schüler zentral gelistet werden können.

Fokus bei der Erreichung dieses Ziel ist nicht nur eine performante Lösung, sondern dass auch Wert auf die Usability gelegt wird.

Beschreibung:

Der Eintrag besteht aus einem Text und einer Option, ob der Eintrag entschuldigt ist. Diese Daten sollten ausführlich dargestellt werden.

Ein Eintrag wird aufgerufen in dem beim Betätigen des Buttons die Schüler ID mitgegeben wird und die Daten werden dann als Antwort zurückgegeben.

Vorbedingung:

Der Akteur muss in dem gleichen Netzwerk sein, wie der Webserver, sowie sollte der Port 80, nicht freigegeben werden. Es muss eine aktive Verbindung zur DB bestehen.

Nachbedingung:

Die serverseitige Validierung sowie die clientseitige Validierung überprüfen die Formfelder, nur dann dürfen die Daten in die DB eingetragen werden.

Auslösendes Ereignis:

Nutzer klickt auf «ansehen» Button eines bestimmten Schülers.

Kategorie:

Primär

Use Case 11; Eintrag bearbeiten:

Ziel:

Ein Schüler Eintrag sollte von dem Akteur bearbeitet werden können.

Beschreibung:

Eine Eintrags ID wird unter der Variable mit dem Namen «EDIT» per GET-Request an «manageClass» gesendet. «manageClass» kontrolliert nun, ob die ID in der DB vorhanden ist, falls ja werden die Formfelder, die für das Erstellen eines Eintrages verantwortlichen sind, mit den Daten der DB geladen. Wird das Formfeld nun abgesendet und besteht die Validierung, werden die Daten in der DB aktualisiert.

Vorbedingung:

Akteur muss in dem gleichen Netzwerk sein wie der Webserver und der Port 80 sollte nicht freigegeben werden. Es muss eine aktive Verbindung zur DB bestehen.

Nachbedingung:

Der Eintrag muss existieren.

Auslösendes Ereignis:

Durch das Betätigen eines editiere Button auf der Eintrags ansehen Seite.

Kategorie:

Sekundär

Use Case 12; Eintrag löschen:

Ziel:

Ein Schüler Eintrag sollte von dem Akteur gelöscht werden können.

Beschreibung:

Eine ID wird unter der Variable per GET-Request gesendet. «manageClass» kontrolliert nun, ob die ID in der DB vorhanden ist, falls ja wird der Eintrag gelöscht, falls nicht wird eine Fehlermeldung dem Nutzer zurückgegeben.

Vorbedingung:

Akteur muss in dem gleichen Netzwerk sein wie der Webserver und der Port 80 sollte nicht freigegeben werden. Es muss eine aktive Verbindung zur DB bestehen.

Nachbedingung:

Die ID muss existieren.

Auslösendes Ereignis:

Durch das Betätigen eines Löschens Button auf der Klassen ansehen Seite.

Kategorie:

Sekundär

Use Case 13; Asp Benutzer Login:

Ziel:

Ein Akteur sollte sich einloggen können

Beschreibung:

Dazu wird das Microsoft eigene Login System verwendet, welche es ermöglicht einen Nutzer einzuloggen.

Vorbedingung:

Akteur muss in dem gleichen Netzwerk sein wie der Webserver und der Port 80 sollte nicht freigegeben werden. Es muss eine aktive Verbindung zur DB bestehen.

Nachbedingung:

Der Benutzer muss existieren.

Auslösendes Ereignis:

Durch das Eingeben der korrekten Anmeldedaten auf der Login Seite.

Kategorie:

Sekundär

Use Case 14; Asp Benutzer Ausloggen:

Ziel:

Ein Akteur sollte sich ausloggen können.

Beschreibung:

Dazu wird das Microsoft eigene Login System verwendet, welche es ermöglicht einen Nutzer auszuloggen.

Vorbedingung:

Akteur muss in dem gleichen Netzwerk sein wie der Webserver und der Port 80 sollte nicht freigegeben werden. Es muss eine aktive Verbindung zur DB bestehen.

Nachbedingung:

Der Benutzer muss eingeloggt sein.

Auslösendes Ereignis:

Durch das Betätigen des Ausloggen Button. Dieser Button steht allen Seiten zur Verfügung.

Kategorie:

Sekundär

Use Case 15; Asp Benutzer erstellen:

Ziel:

Ein eingeloggter Akteur sollte die Möglichkeit dazu haben, weitere Benutzer anzulegen.

Beschreibung:

Dazu wird das Microsoft eigene Login System verwendet, welche es ermöglicht, nur einem eingeloggten Benutzer die Berechtigung dazu geben, ein Benutzer zu erstellen.

Vorbedingung:

Akteur muss in dem gleichen Netzwerk sein wie der Webserver und der Port 80 sollte nicht freigegeben werden. Es muss eine aktive Verbindung zur DB bestehen.

Nachbedingung:

Der Benutzer muss eingeloggt sein. Die Daten müssen den Validierungsbestimmungen entsprechen.

Auslösendes Ereignis:

Durch das Betätigen des Benutzer anlegen Button auf der Anlegen Seite.

Kategorie:

Sekundär

Use Case 16; Asp Benutzer Bearbeiten:

Ziel:

Ein eingeloggter Akteur sollte die Möglichkeit dazu haben seinen eigenen Username, sowie Password zu ändern.

Beschreibung:

Dazu wird das Microsoft eigene Login System verwendet, welche es ermöglicht, innerhalb eines vorgefertigtem Grundgerüst das eigene Benutzerprofil anzusehen. Dort sind die beiden Funktionalitäten zu finden, welche es ermöglicht, das Password oder den Benutzernamen zu ändern.

Vorbedingung:

Akteur muss in dem gleichen Netzwerk sein wie der Webserver und der Port 80 sollte nicht freigegeben werden. Es muss eine aktive Verbindung zur DB bestehen.

Nachbedingung:

Der Benutzer muss eingeloggt sein. Die Daten müssen den Validierungsbestimmungen entsprechen.

Auslösendes Ereignis:

Durch das Betätigen des Benutzer speichern Button auf der Profilseite.

Kategorie:

Sekundär

Produktdaten:

Alle Attribute für die Musskriterien:

Attribute eines Schülers:

- firstname
- lastname
- street + nr
- PLZ
- city
- class_Id
- ID wird automatisch generiert.

Alle Attribute für die Wunschkriterien:

Attribute einer Klasse:

- name
- teachername
- ID wird automatisch generiert.

Attribute eines Benutzers:

- username
- password
- ID wird automatisch generiert.

Attribute eines Eintrages:

- type
- TimeOfAction
- student_ID
- excused
- ID wird automatisch generiert.

Genutzte ASP.NET Login Attribute:

- PasswordHash
- UserName

Benutzungsoberfläche:

Mit Bootstrap und HTML habe ich ein Mockup für das UI gestaltet.



Die aus meiner Arbeit resultierenden Benutzeroberfläche kann von diesem Modell möglicherweise stark abweichen.

Ziel:

Nutzer sollte alle Use Cases mit zwei oder weniger Klicks aufrufen können.

Hier sieht man den Entwurf von dem UI vom Use Case «Schüler ansehen».

The mockup shows a web interface for a student management system. At the top, there is a navigation bar with a 'Home | Schülerverwaltung' link and buttons for 'Klassenverwaltung', 'Klasse erstellen', 'Schüler erstellen', and 'Log out'. Below the navigation bar, a header bar displays 'Schülerverwaltungssystem Prototyp V.001'. The main content area features a green welcome message: 'Herzlich Willkommen Vorname + Nachname #ID'. Below this is a table with columns: ID, Klasse, Vorname, Nachname, and Aktion. The table contains two rows of student data. Each row has three icons in the 'Aktion' column: a speech bubble, a pencil, and a trash can.

ID:	Klasse	Vorname:	Nachname:	Aktion:
3	I2a	Valentine	Wilkerson	  
12	I2a	Tobias	Odermatt	  

Hier sieht man den Entwurf von dem Wunschkriterium Klasse erstellen.

The mockup shows a web interface for creating a new class. At the top, there is a navigation bar with a 'Home | Schülerverwaltung' link and buttons for 'Klassenverwaltung', 'Klassen erstellen', 'Schüler erstellen', and 'Log out'. Below the navigation bar, a header bar displays 'Klasse hinzufügen. Prototyp V.001'. The main content area features a light blue box with the text 'Hier können Sie eine neue Klasse anlegen.' Below this is a form with a label 'Klassennamen *' and a text input field with a placeholder 'Geben Sie eine Klasse ein, maximal 25 Zeichen.' At the bottom of the form are two buttons: 'anlegen' (blue) and 'Löschen' (orange).

Technische Produktumgebung:

Software:

Jeder Benutzer dieser Software muss einen Browser haben, um diese Software zu verwenden.

Hardware:

Ein Webserver muss physisch in einem geschützten Bereich aufgestellt werden.

Zusätzlich muss er mit einer physischen Verbindung, in dem Netzwerk integriert werden.

Orgware:

Die Benutzer müssen virtuell oder physisch in dem Schulnetzwerk integriert werden, damit die Software zu benutzen ist.

Spezielle Anforderungen an die Entwicklungsumgebung:

Hardware:

Mein Arbeitslaptop werde ich dafür verwenden, um diese Applikation zu entwickeln und zu testen.

Software:

Die Programme Visual Studio sowie Microsoft Internet Information Service werde ich nutzen, um meine Applikation zu entwickeln und zu testen.

Orgware:

Um den administrativen Überblick zu halten, werde ich die Software während der Entwicklung, in meine Use Cases unterteilen und diese Use Cases in Form von Todos abarbeiten.

Ergänzungen:

Falls es der Auftraggeber wünscht, könnte mit Absprache des Entwicklers ein Softwarepflegevertrag entstehen. Nach Anfrage würde der Entwickler die Konditionen eines Softwarepflegevertrag genau erläutern.