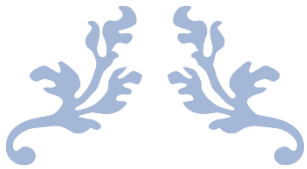


VOLLEY-RESERVIERUNGSTOOL

IT-Projekt: Dokumentation

28. MAI 2021

DOZENTEN: LUKAS FREY, PROF. DR. BRADLEY RICHARDS
Fachhochschule Nordwestschweiz, BSc Wirtschaftsinformatik

Autoren	
Robin Roth	+41 77 427 05 68 robin.roth@students.fhnw.ch
Luca Schädler	+41 79 501 51 95 luca.schaedler@students.fhnw.ch

Betreuende Dozenten	
Lukas Frey	+41 62 957 22 65 lukas.frey@fhnw.ch
Prof. Dr. Bradley Richards	+41 62 957 23 87 bradley.richards@fhnw.ch

Inhalt

Eckdaten

Die Applikation wurde mit den Entwicklungsumgebungen Spring Tool Suite 4 und Visual Studio Code programmiert. Das Backend wurde mit Spring Tool Suite programmiert, das Frontend mit Visual Studio Code. Die Unterstützung der Webtechnologien ist um einiges besser in VS Code, deshalb entschieden wir uns das Projekt mithilfe zweier Umgebungen zu realisieren. Dadurch, dass wir mit beiden Entwicklungsumgebungen den gleichen Folder anvisierten, mussten wir nur die Spring Tool Suite mit Gitlab verbinden. Somit wurden Veränderungen in VS Code durch das «Aktualisieren» in der Spring Tool Suite übernommen.

Die Klasse «PasswordHash» wurde aus dem Internet übernommen. Der Buchungskalender wurde mithilfe der Kalendervorlage aus dem Internet konstruiert. Der restliche Code wurde von den Backyardcoders erstellt.

Quellen:

PasswordHash Klasse: https://howtodoinjava.com/java/java-security/how-to-generate-secure-password-hash-md5-sha-pbkdf2-bcrypt-examples/

Kalendervorlage: https://codepen.io/oliverdunk/pen/pyPrBx

Projektlink

Das gesamte Projekt kann unter nachfolgendem Link eingesehen werden:

Gitlab: https://gitlab.fhnw.ch/luca.schaedler/backyardcodersproject.git

Installationsguide

Schritt 1: Download

Falls Sie über keine Spring Tool Suite verfügen, kann diese unter folgender URL heruntergeladen werden: https://spring.io/tools (Download > 300 MB)

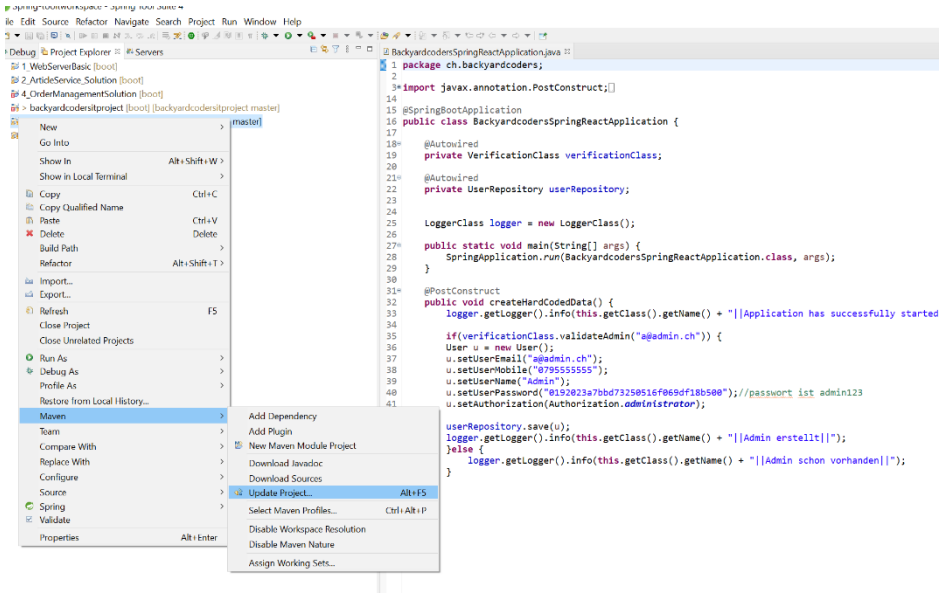
Schritt 2: Einrichtung

Wie bei Eclipse genügt das Herunterladen und Entpacken am geeigneten Verzeichnis. Für die Entwicklung wird ein Java Development Kit (JDK benötigt), dieses sollte jedoch schon von Eclipse vorhanden sein.
Beim Start von Spring Tool Suite wird ein Workspace wie ebenfalls bei Eclipse erwartet. (Spring Tool Suite funktioniert eigentlich von der Umgebung gleich wie Eclipse und sollte daher keine grossen Probleme bereiten.)

Schritt 3: Import des Projekts

Das Importieren funktioniert gleich wie bei Eclipse.

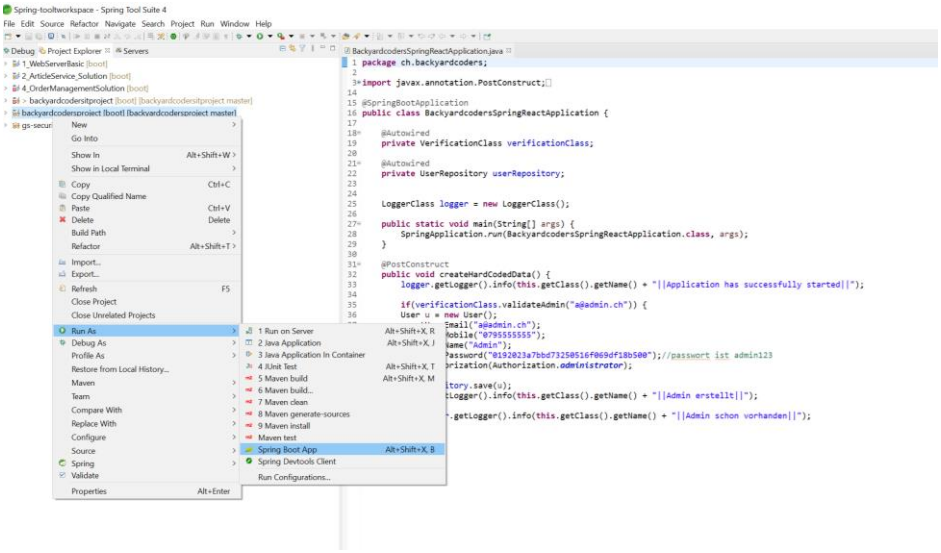
1. File → Import
2. Git → Smart Import
3. Gitlab-Link einfügen und im Workspace abspeichern.
4. Rechtsklick auf das Projekt → Maven → Update Project



Beim ersten Herunterladen, kann dies einige Sekunden dauern, da erst die Bibliotheken heruntergeladen werden müssen.

Schritt 4: Starten der Applikation

Die Applikation wird wie folgt gestartet: Rechtsklick auf das Projekt → Run As → Spring Boot App

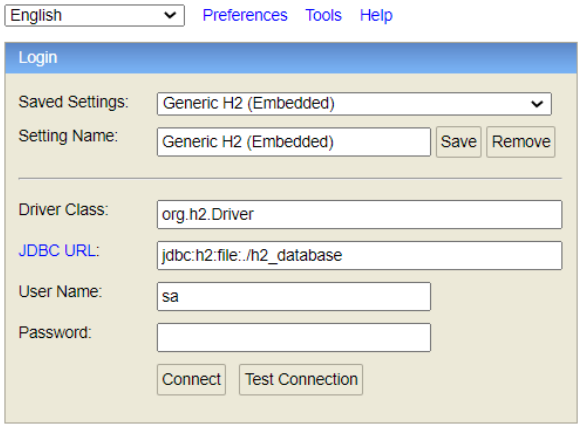


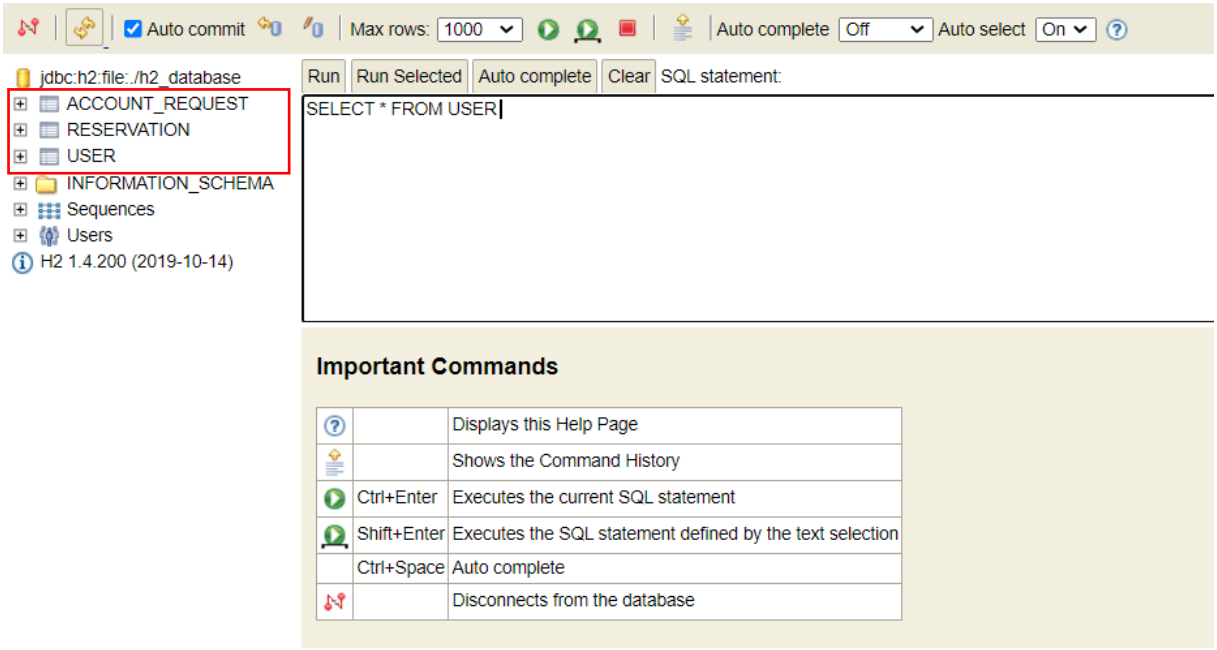
Der Server wird in Spring Tool Suite gestartet und loggt Einträge in Konsole und in eine Textdatei.

Nachdem erfolgreichen Start des Servers kann über den Browser die URL: http://localhost:8080/ (Port Nr. = 8080) aufgerufen werden. Das UI unserer Applikation erscheint nun im Browserfenster.

Die Datenbank, welche wir zum Testen verwenden, ist die integrierte H2 Datenbank von Spring Tool. Die Konsole der Datenbank ist unter der URL: http://localhost:8080/console/ zu erreichen.

Die Angabe des Usernamens «sa» reicht aus für das Einloggen in die Datenbank. Ein Passwort wird keines benötigt.





Rot umrahmt sind die verwendeten Tabellen (durch die Annotationen von Spring Tool werden diese automatisch erstellt).

Funktionen

Home

- Wenn der Benutzer über einen Login verfügt, kann sich dieser mittels Email und Passwort anmelden
- Wenn der Benutzer über keinen Login verfügt, kann dieser über den Link «Noch keinen Account» durch die Eingabe von Namen, E-Mail, Mobile, Passwort eine Accountanfrage erstellen.
    - Erst wenn der Administrator diese Accountanfrage in einen User umwandelt, kann sich der Benutzerüber das Login anmelden.

Buchungskalender

- Der Name des momentan angemeldeten Benutzers wird angezeigt (Hallo, Admin)
- Über die Buttons der Navigationsleiste kann zu den Sichten Userliste, Accountanfragenliste oder «Userangaben ändern» gewechselt werden
    - Die Buttons «Userliste» und «Accountanfragenliste»
- Über Logout kommt der Benutzer zur Homesicht und ist abgemeldet
- Frei Zeitfenster werden durch grüne Buttons angezeigt.
- Besetzte Zeitfenster werden durch rote Buttons angezeigt.
- Durch die Links «Vorherige Woche» und «Nächste Woche» kann der Benutzer durch den Kalender navigieren.

Reservation erstellen

- Wenn der Benutzer den Platz reservieren will, kann er den grünen Reservier-Button zur gewünschten Zeit anklicken
    - Die Details zur Reservation werden aufgelistet und der Benutzer wird aufgefordert die Namen der Spieler einzutragen.
    - Durch einen Klick auf den Button «Zurück» kommt der Benutzer zur Kalendersicht, ohne dass eine Reservation erstellt wurde.
    - Erst durch den Klick auf den Button «Reservieren» wird die Reservation vollzogen und auf der Datenbankabgelegt. Gleichzeitig wird der grüne Reservier-Button rot gefärbt und der Text ändert sich von «reservieren» auf «bearbeiten/löschen».

Reservation ändern/löschen

- Ein Benutzer kann seine eigens erstellten Reservationen ändern oder löschen, jedoch nicht Reservationen anderer Benutzer (ausser es handelt sich um einen Admin!)
- Wenn der Benutzer eine Reservation ändern oder löschen will, kann er den roten Button («bearbeiten/löschen») klicken
    - Die Details zur Reservation werden aufgelistet.
    - Durch einen Klick auf den Button «Zurück» kommt der Benutzer zur Kalendersicht, ohne dass eine Reservation erstellt wurde.
    - Durch die Eingabe der Spieler gefolgt von einem Klick auf den Button «Reservation ändern», kann der Benutzer seine Reservation ändern.
    - Durch einen Klick auf den Button «Reservation Löschen» kann der Benutzer seine Reservation löschen.

Userliste

- Kann nur durch die Administratoren aufgerufen werden.
- Administratoren können nicht gelöscht werden.
- Alle in der Datenbank gespeicherten User werden in einer Tabelle angezeigt.
- Durch einen Klick auf den Button «Zurück» kommt der Administrator zur Kalendersicht.
- Der Administrator kann durch einen Klick auf den Button «Löschen» den gewünschten User löschen.

Accountanfragenliste

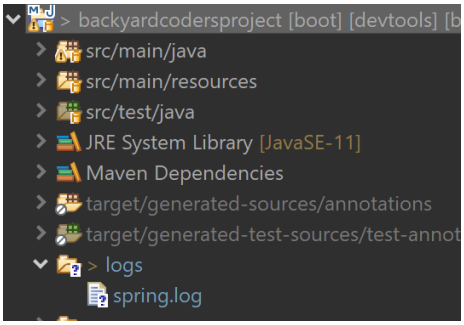
- Kann nur durch die Administratoren aufgerufen werden
- Alle in der Datenbank gespeicherten Accountanfragen werden in einer Tabelle angezeigt.
- Durch einen Klick auf den Button «Zurück» kommt der Benutzer zur Kalendersicht
- Der Administrator kann durch einen Klick auf den Button «Anfrage Löschen» die gewünschte Accountanfrage löschen.
- Der Administrator kann durch einen Klick auf den Button «Account Erstellen» einen User aus der Accountanfrage erstellen.

Userangaben ändern

- Kann durch jeden Benutzer aufgerufen werden.
- Durch einen Klick auf den Button «Zurück» kommt der Benutzer zur Kalendersicht.
- Der Benutzer kann durch das Ausfüllen aller Felder (Name, E-Mail, Mobile, Passwort, Passwort wiederholen) und das Klicken auf den Button «Änderungen Speichern» seine Accountdaten ändern.

Logging

- Ein Logger loggt Einträge (> Level Info) in der Konsole, sowie auf einer separaten Datei.
    - Die Datei befindet sich im Projekt-Folder unter logs/spring.log
- Bei einer Grösse von 10MB wird die Datei von oben nach unten überschrieben.
    - In anderen Worten: die ältesten Einträge werden durch die neusten ersetzt.



Testdaten

Um den Dozenten das Testen der Applikation zu vereinfachen, hat das Projektteam Testdaten erstellt.

Accountanfragen

SELECT * FROM ACCOUNT_REQUEST;

ACCOUNT_REQUEST_ID	ACCOUNT_REQUEST_EMAIL	ACCOUNT_REQUEST_MOBILE	ACCOUNT_REQUEST_NAME	ACCOUNT_REQUEST_PASSWORD
8	hans@hotmail.com	07989967542	Hans	573bd983f1a92bb6cf8b535919e3a728
10	hanna@hotmail.com	0762336879	Hanna	80b1087caacd60a17ffe976e8681ac3c
12	leonard@hotmail.com	0794112233	Leonard	e662b774bfbd8b7bf43c09ab5fd0c1a5
13	bertha@hotmail.com	0797885140	Bertha	1deefbe9b36cc6c3db2a4f1ee2cb2662
18	anja@hotmail.com	0794567890	Anja	2bcb242e395e9e513aa50657fe7f616b
19	gina@hotmail.com	0783337542	Gina	6561b48b67ac11a03bb406f1f98149e1
21	jana@hotmail.com	0765546572	Jana	bb61710601c4257d26169c6b0e690edc
23	hugo@hotmail.com	0554336879	Hugo	50ff26fc9409153e45d9647bfe4dfcc6

(8 rows, 2 ms)

User

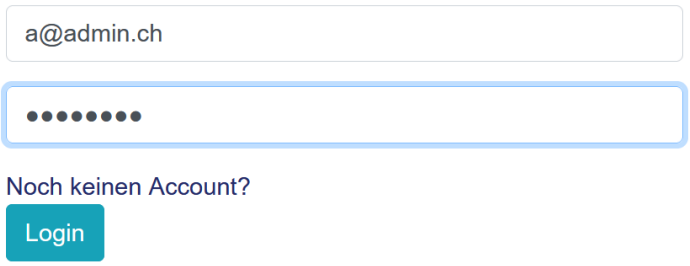
SELECT * FROM USER;

USER_ID	AUTHORIZATION	USER_EMAIL	USER_MOBILE	USER_NAME	USER_PASSWORD
1	0	a@admin.ch	0795555555	Admin	0192023a7bbd73250516f069df18b500
15	1	paul@hotmail.com	0786549345	Paul	2a6df21e82a3b929df6a7076c4be1d06
16	1	chloe@hotmail.com	0787771198	Chloe	5f596fde55cdcb2d67b77b58136d933f
17	1	bastian@hotmail.com	0773469087	Bastian	7c95314b122422875836cb30795f4231
22	1	inga@hotmail.com	0776543210	Inga	72c70c64a1122c6255bf62b701ac0c6a

(5 rows, 2 ms)

Die Accountanfragen sowie die User wurden alle nach dem gleichen Muster erstellt. D.h. die Passwörter bestehen jeweils aus dem doppelten, kleingeschriebenen Namen

- Z.B. Name = Hans → Passwort = «hanshans»



Der Zugang zum Administratorenaccount erfolgt durch die Eingabe von:

- Email a@admin.ch
- Passwort «admin123»

Verlauf

Das Projekt konnte gut anhand der Anforderungsanalyse realisiert werden. In der Applikation sind alle Use Cases, welche in der Anforderungsanalyse definiert wurden, enthalten. Die HTTP-Methoden des Backend haben wir alle mit Hilfe von Postman während der Programmierung getestet, so konnten wir davon ausgehen, wenn später im Client Probleme auftreten, dass das Problem im Frontend liegt. Die definierten Test-Cases haben wir am Ende des Projektes nochmals komplett durchgespielt, um allfällige Bugs zu erkennen.

In der Systemspezifikation, mussten wird bei den Entitäten der Datenbank kleine Änderungen vornehmen. Wir haben nur eine User-Entität erstellt und als Attribut die Rolle (Bsp. Admin) definiert, anstatt 3 Entitäten. Des Weiteren haben wir komplett auf einen IT-Verantwortlichen verzichtet, da sich während des Projekts nicht als sinnvoll herausgestellt hatte, eine andere Berechtigung zu implementieren. Das Klassendiagram weicht leicht von unserer Applikation ab, bezüglich den Klassennamen und den Namen der implementieren Methoden. Wir haben ebenfalls noch einige Klassen mehr Programmiert (z.B. Die Klasse PasswordHash, um das Password zu hashen vor dem Abspeichern in der Datenbank).

Hürden während der Programmierung

Bei der Backendprogrammierung hatten wir keine größeren Schwierigkeiten oder Probleme. Da wir in der Vergangenheit bereits Projekte mit Java realisieren konnten. Auch fiel uns dadurch die Planung des Backends erheblich leichter.

Die Realisierung des Frontends lief nicht immer wunschgemäss. Zu Beginn erstellten wir pro Sicht (Home, Buchungskalender, Userliste, etc.) je eine HTML, JS und CSS Datei. Dies, um den Code zu trennen und übersichtlicher zu gestalten. Jedoch merkten wir bald, dass so das Login leicht umgangen werden kann, indem man in der Adressleiste des Browsers die URL der Kalendersicht eingibt. Auch wussten wir nicht wie wir das Userobjekt, des momentan angemeldeten Benutzers, zwischen den Sichten übergeben konnten. Nach vielen Diskussionen kamen wir auf drei Lösungsansätze:

- Je eine HTML,CSS und JS Datei für das gesamte Projekt
    - Sehr unübersichtlich
- Eine Lösung mit einem PHP-Skript
    - Wäre die sicherste Lösung, jedoch haben beide Backyardcoders keinerlei Erfahrungen mit PHP
- Eine Lösung mit JS-Framework React
    - Einfaches Testing der Webanwendung
    - Ermöglicht eine bessere, übersichtlichere Struktur als mit Vanilla JS

Wir haben uns schliesslich entschieden eine SPA (Single-Page Applikation) mit React zu erstellen. Im Zuge der Entscheidung haben wir unsere Applikation und die vorhandenen HTML und JavaScript-Files nochmals neu mit Hilfe vom Framework React programmiert. Dies stellte sich jedoch als sehr kompliziert heraus, da das Framework an Komplexität nicht zu überbieten ist. Wir versuchten 1 bis 2 Wochen unser gesamter Client mit React zu realisieren. Jedoch mussten wir wohl oder übel einsehen, dass dieses JS-Framework nicht so schnell zu erlernen ist. Nach vielen Hindernissen und wenigen Fortschritten haben wir uns dazu entschieden das Projekt nochmals umzustellen. Am Ende entschieden wir uns nur eine HTML und JS Datei zu verwendeten. Die Folge davon ist eine grosse Einbusse von Übersichtlichkeit, die wir letztendlich in Kauf nahmen.

Fazit

Dieses Projekt war für uns äusserst lehrreich. Es hat uns klipp und klar aufgezeigt wie bedeutend eine ordentliche Planung ist. Durch unser Knowhow in Java konnten wir das Backend gut durchplanen, demnach hatten wir bei der Implementierung auch keine grossen Hindernisse zu bewältigen. Wohingegen die Planung des Clients eher kurz ausfiel, somit war die Realisierung eher problematisch. Durch gute Teamarbeit, Ehrgeiz und Geduld konnten jedoch alle Hindernisse bewältigt werden.