Reflektion

In diesem Projekt habe ich mich komplett um das Frontend gekümmert. Ich habe mich im Projekt zwar nicht unbedingt mit Java Spring Boot beschäftigt, dafür aber eine mir noch mehr unbekannte Technologie verwendet. Für das Frontend verwenden wir Kotlin/JS. Kotlin/JS bietet die Möglichkeit, Applikationen in Kotlin zu schreiben, die am Ende in JavaScript kompiliert werden. Warum haben wir uns für Kotlin/JS entschieden? Da wir uns entschieden haben, Kotlin als Programmiersprache für das Backend zu verwenden, kam mir der Gedanke, dass wir dadurch Model-Klassen durch Kotlin Multiplatform im Backend und Frontend verwenden könnten. Dadurch wären beispielsweise die Klassen, welche als Objekte von Java Spring Boot für den Controller versendet werden, die selben wie für das Frontend. Leider kam das Problem auf, dass die aktuelle Version von den JS-Wrapper Klassen für Kotlin/JS (react & mui) noch nicht vollständig kompatibel mit dem neuen Kotlin/JS Compiler IR sind, welchen wir benötigen um Kotlin Multiplatform zu verwenden.

Im Frontend habe ich viel mit StateFlows gearbeitet, um die UI mit dem ViewModel zu synchronisieren oder auch um auf asynchrone Anfragen zu reagieren. StateFlows sind wie Observer von Kotlin Coroutines. Wird eine Anfrage gesendet, wird zuerst der State Loading in den Flow geschmissen. Das ViewModel lauscht auf dem Flow und sagt der UI, dass die App lädt und einen Ladebalken anzeigen soll. Kommt eine Antwort von dem Server zurück, wird ein entsprechendes Event (Erfolgreich oder Fehler) in den Flow geschissen und das ViewModel bzw. die UI reagiert darauf.

Wir haben Jira verwendet, um die agile Entwicklungsmethode SCRUM zu verwenden. Für die Aufgabenplanung hat dies gut funktioniert und ich habe versucht, mich an die Aufgaben für den aktuellen Sprint zu halten.

Zudem habe ich mich mit Continuous Integration befasst. Leider war die Zeit zu knapp, um dies vollständig zu integrieren. Ich weiß das wir uns bei Ihnen bezüglich eines Servers melden sollten, aber die Zeit hat dies nicht hergegeben. Hier habe ich einen Raspberry Pi so konfiguriert, dass dieser bei mir zuhause als GitLab Runner für das Repository fungiert. Ich bin an dem Punkt hängen geblieben, die gebaute Applikation auf Docker Hub hochzuladen und diese auf dem Pi als Server zu deployen. Diesen Schritt hätte ich gerne noch vollendet.

Durch dieses Projekt habe ich sehr viel gelernt. Ich bin zwar manchmal ziemlich an die Grenzen von Kotlin/JS bzw. den Wrapper Klassen gestoßen, hatte aber immer nach ein bisschen rumprobieren eine Lösung parat. Durch mache sehr merkwürdigen Fehlern überlege ich mir, diese genauer anzuschauen auf GitHub zu melden, so dass vielleicht auch andere etwas von meinen Problemen haben.