chumm-uffa 25.11.2017

Entscheide

- Frontend und Backend werden separiert. Sie werden in unterschiedlichen Container ausgeführt.
- Das Styling wird erst am Schluss entschieden. Zuerst werden Routing und Funktionalität erstellt. (Design follow function)
- Schnittstelle zwischen Frontend und Backend werden via API-Projekt zur Verfügung gestellt. Somit wird sicher gestellt, dass die übertragenen Daten auf beiden Seiten verstanden werden.
- Rest Service URI beginnt immer mit /api. Somit kann das Routing von Frontend zu Backend sicher gestellt werden.
- Die Suche muss separat nochmals beschrieben werden, da diese zu kompliziert ist.
- Jede Seite wird als separates Modul umgesetzt.
- Pullrequest werden gegenseitig gemerged.
- Im Commit immer Issue nummer vermerken. Somit wir die Verbindung von Commit zu Issue hergestellt.
- Entwicklungsstrategie ist, möglichst dumme Komponenten zu haben und möglichst alles in Service aus zu lagern. Unittest werden nur für Services erstellt, aber nicht für Komponenten.
- HTTP Zugriffe mit Angula HTTP Module implementieren.

Main UseCase

- Einfacher Ansatz. Ein Benutzer registriert sich. Anschliessend kann er eigene Meetups erfassen oder sich auf bereits vorhandene Meetups registrieren. Der Owner des Meetup kann dann entscheiden, ob er einen anderen Benutzer mitnehmen möchte oder nicht.
- Die auf ein Meetup registrierten Benutzer können via Chat Informationen austauschen.

Vorgehen

- Alle Seiten mit Routing erstellen
- Seiten mit HTML Elementen füllen
- Services erstellen

Fragen

- Welche DB wollen wir nehmen. NoSql oder SQL
- NgBootstrap Framework oder anderes Styling Framework verwenden erlaubt? => Responsive!!