SHREDDIT^3

by Rolf & Rolf

2Rolf FmbH

1. Inhalt

- Client
- Server
- Tests
- Erkenntnisse
- Fragen

2. Client – Frameworks

- Angular.JS
- Bootstrap
- jQuery

2. Client – Struktur

Hauptdatei

- Index.html

Template-Dateien

- about.html
- comments.html
- error.html
- login.html
- new.html
- new-comment.html
- postings.html
- register.html
- settings.html

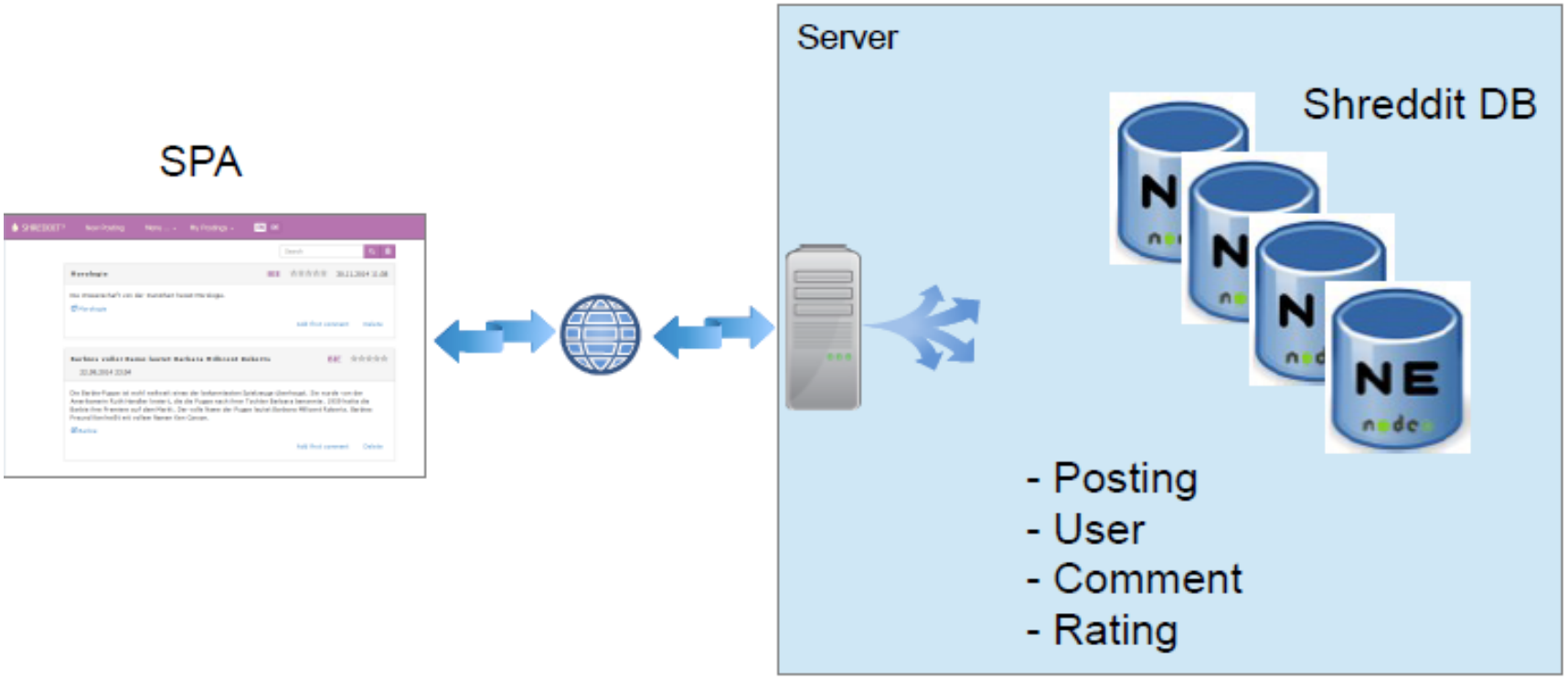
2. Client – Features

- Postings anzeigen und erfassen
- Sortieiren/Filtern der Postings
- Suchen nach Postings
- Kommentare anzeigen und erfassen
- Login
- Registrieren und anpassen der Einstellungen
- About
- Sprachumschaltung

2. Client – Fehlende Features

- Erfassen und anzeigen von Bildern
- Seitenweises anzeigen der Postings (scroll & reload)
- Onbeforeunload (bei nicht gespeicherten Daten)
- Tests (AngularJS)
- Security
- Accessability

3. Server – Übersicht

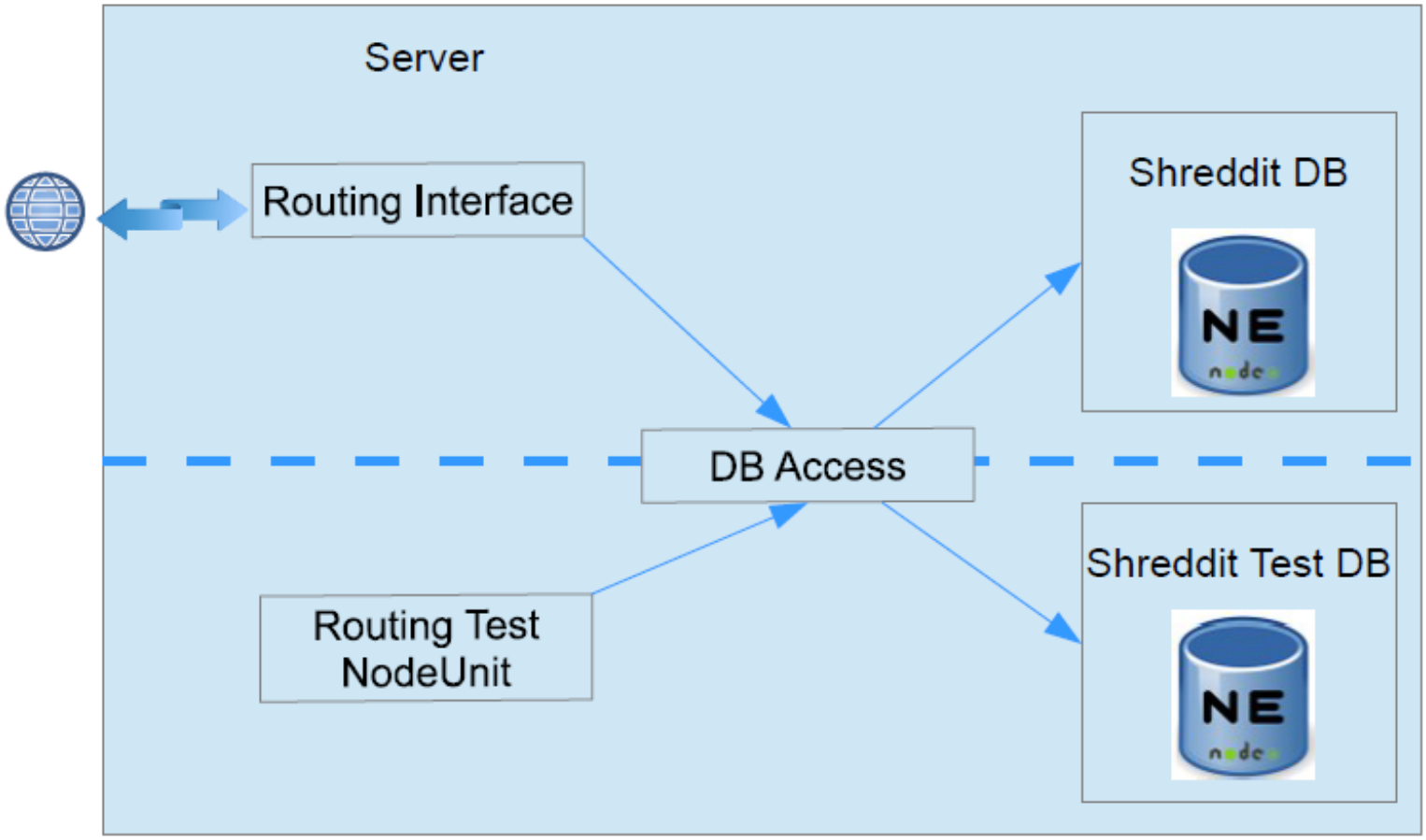


3. Server – Datenbank

Gründe

- Einfache Anwendung unter Node.js
- Einfaches API
- Funktionsumfang für unsere Anwendung ausreichend
- Einfacher Ausbau zur Mongo DB möglich

3. Server – Aufbau der Software



3. Server – DB-Struktur

User-DB

{"**_id**":"newUser",
"**password**":"7c4a8d0…494f8941b",
"**email**":"newUser@hsr.ch",
"**_version**":1,
"**since**":"2014-07-04T10:44:00.000Z",
"**locale**":"EN",
"**notify**":"true",
"**admin**":"false"}

Posting-DB

{"**title**":"Testing",
"**user**":"tester",
"**version**":1,
"**time**":"2014-11-30T12:02:29.245Z",
"**rating**":"0.00",
"**people**":"0",
"**link**":"Testing",
"**url**":"www.testing.ch",
"**commentCount**":0,
"**tags**":"test",
"**content**":"we test the shreddit3",
"**_id**":"BG3fY0gikybQrgi0"}

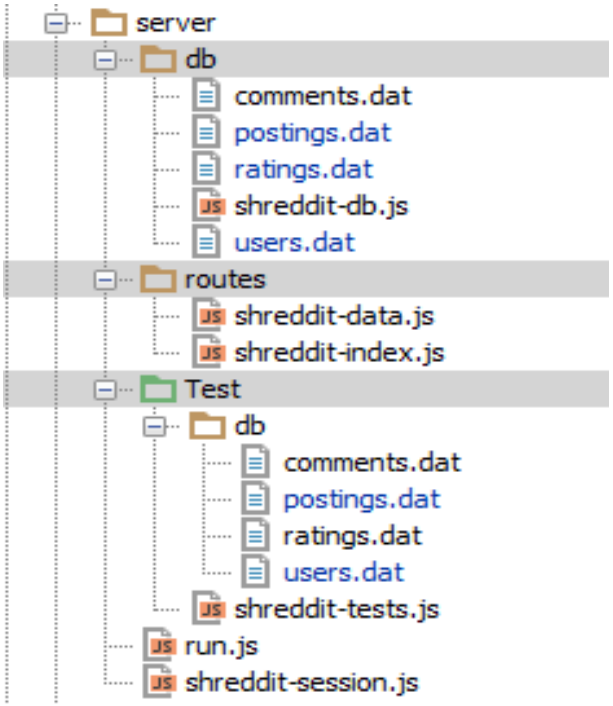
Comment-DB

{"**pid**":"BG3fY0gikybQrgi0",
"**user**":"tester",
"**time**":"2014-11-30T13:44:15.343Z",
"**comment**":"this is a comment for Morologie.",
"**response**":"bie@example.com",
"**_id**":"3DomEQPiDEiWLuIO"}

Rating-DB

{"**_id**":"ME4QWkUBSZ4dEZQx",
"**_count**":1,
"**_average**":4,
"**_version**":1,
"**tester**":4}

3. Server – Dateistruktur



3. Server – Übersicht

Vorteile:

- Der DB-Access führt alle Zugriffe auf die DB aus.
- Nach Festlegung des schmalen Routing-Interface könnte die SPA- und der Server-Code unabhängig entwickelt werden.
- Testing ist bereits bei der Entwicklung des Servercodes möglich.
- Spezielle DB Instanzen für das Testing (vordefinierte Muster in der DB).

4. Tests – NodeUnit

Wir haben NodeUnit verwendet.

Gründe

- Einfache Anwendung unter Node.js
- keine weiteren Abhängigkeiten zu weiteren Tools.

Befehl

nodeunit server/test/shreddit-tests.js --reporter html > c:/temp/shredditTest.html

4. Tests – Protokoll

Testbericht

```
shreddit-tests.js
  1. testUpdateRating
  2. testUserDB - getUser
  3. testUserDB - regExistUser
  4. testUserDB - regUser
  5. testPostingDB - onPosting
  6. testPostingDB - allPostings
  7. testPostingDB - topRatingSortPostings
  8. testPostingDB - postPosting
  9. testPostingDB - deletePosting
OK: 38 assertions (67ms)
```

Code-Beispiel

```
DB.getUserData(userExist, passw, function (err, user) {
    test.expect(5);
    test.deepEqual(err, null, "check user DB, no err");
    test.deepEqual(user._id, userExist, "check user DB, user name");
    test.deepEqual(user.password, true, "check user DB, user password is ok");
    test.deepEqual(user.email, "test@example.com", "check user DB, user's email-address");
    test.deepEqual(user.admin, "true", "check user DB, user is admin");
    test.done();
})
```

5. Erkenntnisse

- Qual der Wahl → Es gibt für jeden Bereich X Frameworks
- Frameworks haben auch Nachteile.
- Werbefenster?

6. Fragen