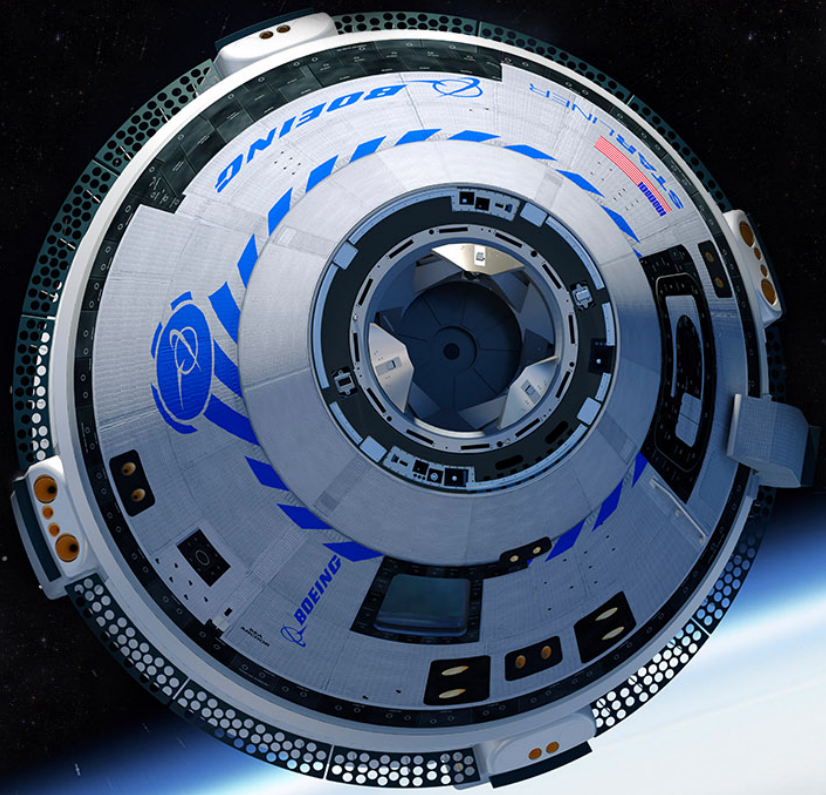
End-to-end testing für Web Appliaktionen

E2E Tests sind langsam, kompliziert, teuer und haben keinen Nutzen, wenn bereits Unit- und Integrationstests verwendet werden!

BOEING

STARLINER

YOUR RIDE IS HERE.



Der Starliner ist eine Raumkapsel der nächsten Generation, die Menschen in und aus dem Orbit bringen wird.

– Boeing

20.12.2019

Am 20. Dezember des Jahres 2019 befindet sich der Starliner auf seinem ersten Flug zur Internationalen Raumstation.

Der Start war ein voller Erfolg.

bis …

Was ist passiert?

Aufgrund eines Programmierfehlers wurden fehlerhafte Daten der “Mission Clock” verwendet, was wiederum einen höheren Treibstoffverbrauch verursachte und der Starliner nicht mehr genug Treibstoff hatte, um es in die Umlaufbahn zu schaffen

Ok… aber wie konnte das passieren?

- Die Ingenieure bei Boeing führten eine enorme Anzahl an Tests durch
-
-

- Die Ingenieure bei Boeing führten eine enorme Anzahl an Tests durch
- Aber die Tests wurden in einzelne Module aufgeteilt
-

- Die Ingenieure bei Boeing führten eine enorme Anzahl an Tests durch
- Aber die Tests wurden in einzelne Module aufgeteilt
- Um Boeing zu zitieren: dies geschah, "weil die Ingenieure es für logisch hielten, nicht den gesamten Flug auf einmal durchzutesten”.

- Die Untersuchungen zeigten auch zusätzliche Softwareprobleme, die einen noch katastrophaleren Ausfall hätten verursachen können.
- Trotz des Problems sagten Boeing und die NASA, die Testmission habe die meisten ihrer Ziele erreicht. Die Daten zeigten, dass die Kapsel eine sichere Umgebung für die Beförderung von Personen darstellte.

Weitere Untersuchungen ergaben, dass nicht nur der Fehler mit der Mission Clock sondern noch zahlreiche weitere kritische Fehler durch einen vollständigen End-to-End Test rechtzeitig entdeckt worden wären

Also - wenn es absolut sinnvoll ist, den gesamten Raketenflug in einem Durchgang zu testen - warum sollte dein Software-Projekt anders sein?

End-to-end testing für Web Appliaktionen

E2E-Tests sind langsam, kompliziert, teuer, aber für die Softwarequalität absolut notwendig, auch wenn bereits Unit- und Integrationstests verwendet werden!

Die NASA wird wahrscheinlich etwa 90 Millionen Dollar für jeden Astronauten zahlen, der an Bord der CST-100 Starliner-Kapsel von Boeing auf Missionen der Internationalen Raumstation (ISS) fliegt

Die End-to-End-Tests hätten ~25 Stunden gedauert.

End-to-end testing für Web Appliaktionen

E2E Tests sind langsam und kompliziert, aber sie sparen bares Geld und sind für die Softwarequalität absolut notwendig, auch wenn bereits Unit- und Integrationstests verwendet werden!

```
I.amOnPage('/user/login');
I.fillField('Username', 'Markus');
I.fillField('Passwort', '1234');
I.click('Anmelden');
I.see('Willkommen');
```

End-to-end testing für Web Appliaktionen

E2E Tests sind langsam, aber einfach zu schreiben, sparen bares Geld und sind für die Softwarequalität absolut notwendig, auch wenn bereits Unit- und Integrationstests verwendet werden!

Aber ja, sie sind langsam.

End-to-end testing für Web Appliaktionen

E2E Tests sind langsam, aber einfach zu schreiben, sparen bares Geld und sind für die Softwarequalität absolut notwendig, auch wenn bereits Unit- und Integrationstests verwendet werden!

E2E für Web Applikationen

Anforderungen

- Web-Browser automatisieren
- Tests in allen Browsern ausführen, die von der Applikation unterstützt werden
- Authentifizierung, z. B. im Intranet
- Nach der anfänglichen Einrichtung sollte es einfach sein, Tests zu schreiben und zu aktualisieren

Bekannte E2E Tools

- Selenium
- Jest
- Protractor
- Cypress.io
- CodeceptJS

Selenium

Pros

- Gibt es schon seit vielen Jahren
- Unterstützt alle gängigen Browser

Cons

- Sehr langsam in der Ausführung
- Tests sind kompliziert zu schreiben und nicht einfach zu lesen

Jest

Pros

- Kann Selenium im Hintergrund nutzen und somit alle gängigen Browser unterstützen
- Nutzt Appium für das Testen auf Mobilgeräten

Cons

- “One catch all” Lösung. Focus liegt zu sehr auf Unit- und Integrationtesting

Protractor

Pros

- Kann Selenium im Hintergrund nutzen und somit alle gängigen Browser unterstützen
- Nutzt Appium für das Testen auf Mobilgeräten

Cons

- -

Cypress.io

Pros

- Super GUI die einen Schritt für Schritt durch den komplett Test durchgehen lässt und an jeder Stelle die Möglichkeit bietet die Applikation zu debuggen

Cons

- Läuft nur in Chrome

CodeceptJS

Pros

- Kann Selenium im Hintergrund nutzen, so dass auf allen gängigen Browsern getestet werden kann, aber unterstützt zusätzlich Puppeteer and Playwright, die sehr viel schneller sind
- Nutzt Appium für das Testen auf Mobilgeräten
- GUI (Beta) ähnlich zu der von cypress.io
- Sehr einfache Syntax um Tests zu schreiben

Cons

- -

– Live coding part –

Fazit:

Sei klüger als Boeing, auch wenn du keine Raketen baust