DHBW
Duale Hochschule
Baden-Württemberg
Mannheim

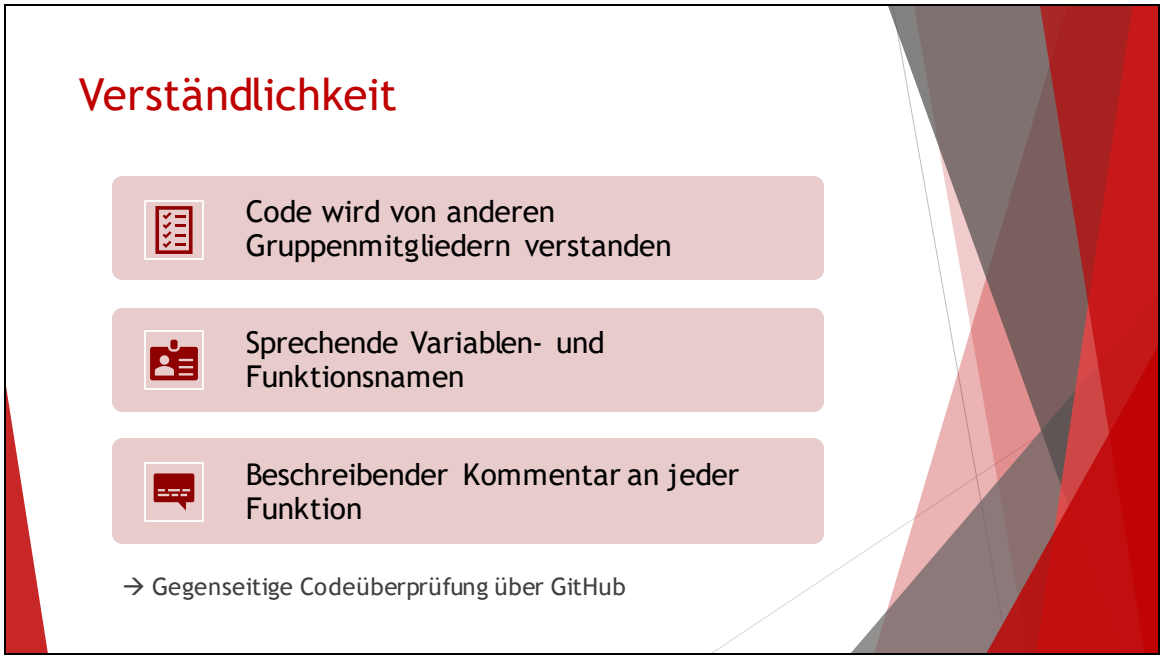
Server Beta

Backend Gruppe 2

Projektkonzeption und -realisierung

Mannheim, Dezember 2023


Qualitätsmaßnahmen



- Kein Merge ohne Review eines anderen: mit GitHub Restriktionen setzen
- Nur abgenommener Code wird weiter genutzt
- Festlegung, wer welchen Code liest: wöchentliche Auslosung wer wen "überprüft". Jeder bekommt eine bestimmte Person zugewiesen, dessen Code er überprüfen muss
- Zeiten:
  - Iterationsplanung: Freitag in der Vorlesung
  - Bearbeitung zugewiesener User Stories: bis Dienstag 18 Uhr
  - Code Reviews: bis Mittwoch 18 Uhr
  - Ggf. Überarbeitung nach Code Review Mitteilung: bis Donnerstag Abend/Freitag Morgen

Checkliste für Reviewer

- Wurden die Variablen sprechend benannt?
- Wurden die Funktionen sprechend benannt?
- Sind Tests vorhanden?
- Tut das Programm das was es soll? (Output) --> Vergleiche Spezifikationen
- Ist der Code verständlich geschrieben?

Checkliste für Reviewer

Vollständigkeit

Akzeptanzkriterien sind erfüllt

Spezifikation der Schnittstelle ist erfüllt

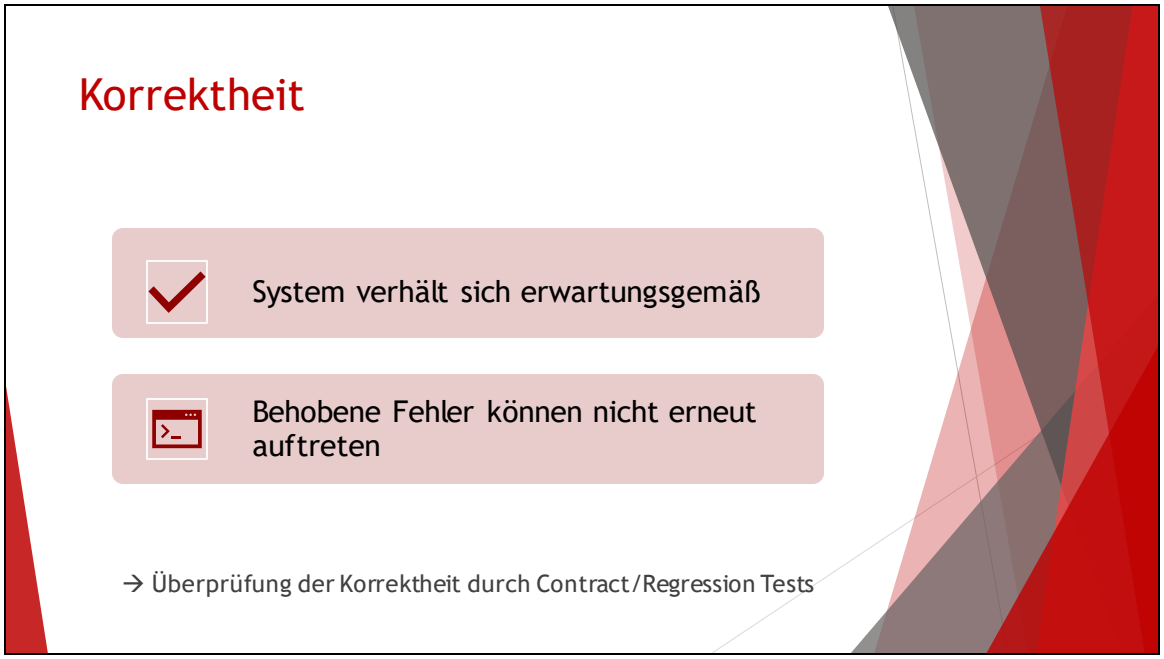
→ "Sprint Reviews" zur Überprüfung der Vollständigkeit

- Sprint Reviews: Donnerstag, gemeinsames Treffen der Gruppe
- Präsentation des Arbeit nach folgenden Fragen:
  - Was habe ich umgesetzt?
  - Was gab/gibt es für Probleme?
  - Was habe ich nicht geschafft?
  - Wo habe ich noch Fragen?
  - Wünsche über nächste Iteration/Beschwerden über vergangene Iteration
- Product Owner überprüft User Stories auf Erfüllung/Akzeptanzkriterien
- Gemeinsames Produktinkrement für Weiterarbeit

Fragen für Entwickler

- Was habe ich umgesetzt?
- Was gab/gibt es für Probleme?
- Was habe ich nicht geschafft?
- Wo habe ich noch Fragen?
- Wünsche über nächste Iteration/Beschwerden über vergangene Iteration

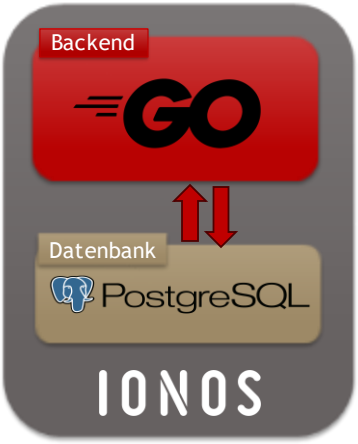
Product Owner überprüft User Stories auf Erfüllung/Akzeptanzkriterien

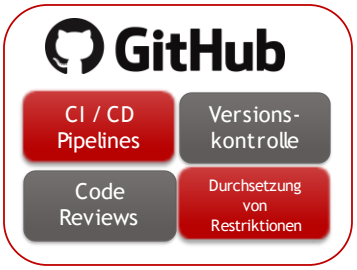


- Jeder schreibt Tests zu den eigens umgesetzten Funktionen
- "Wichtige" Tests werden geschrieben:
  - Contract Tests: Tests, ob die Spezifikationen erfüllt sind
  - Überprüfung mit Code Coverage: Gibt es Abschnitte, die nicht getestet wurden? --> Erweiterung der Tests oder Entfernung des Codes, wenn dieser nicht benötigt wird
  - Regression Tests: Für aufgetretene Bugs werden Tests geschrieben, sodass dauerhaft überprüft wird, dass diese nicht erneut auftreten

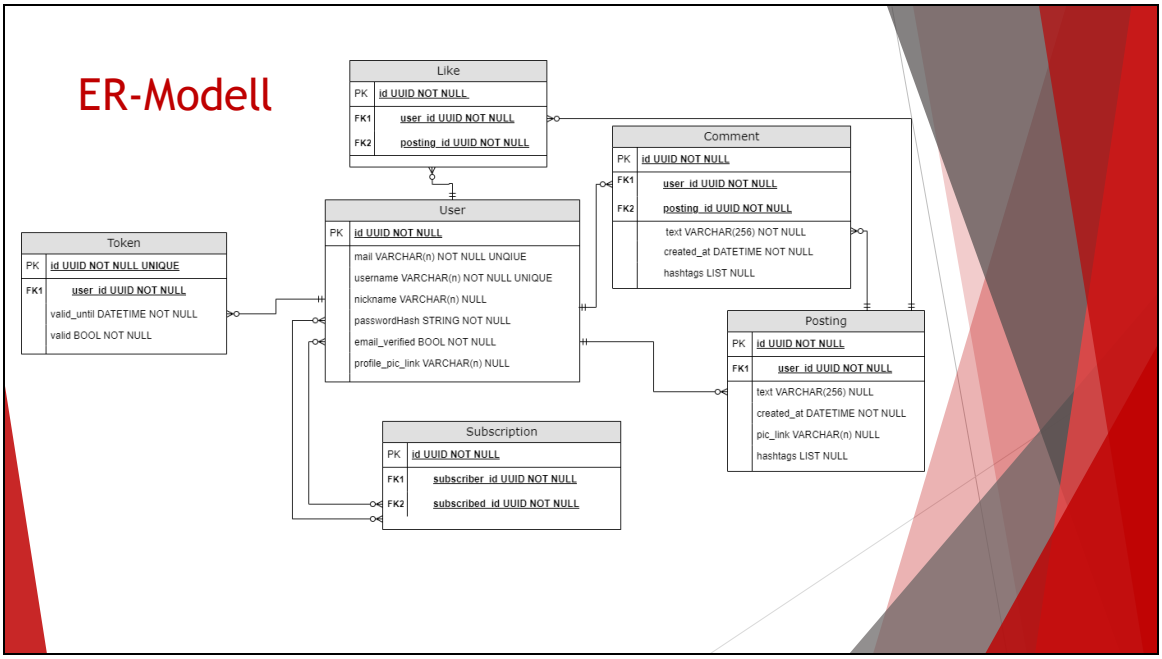

Technologien

Technologiestack

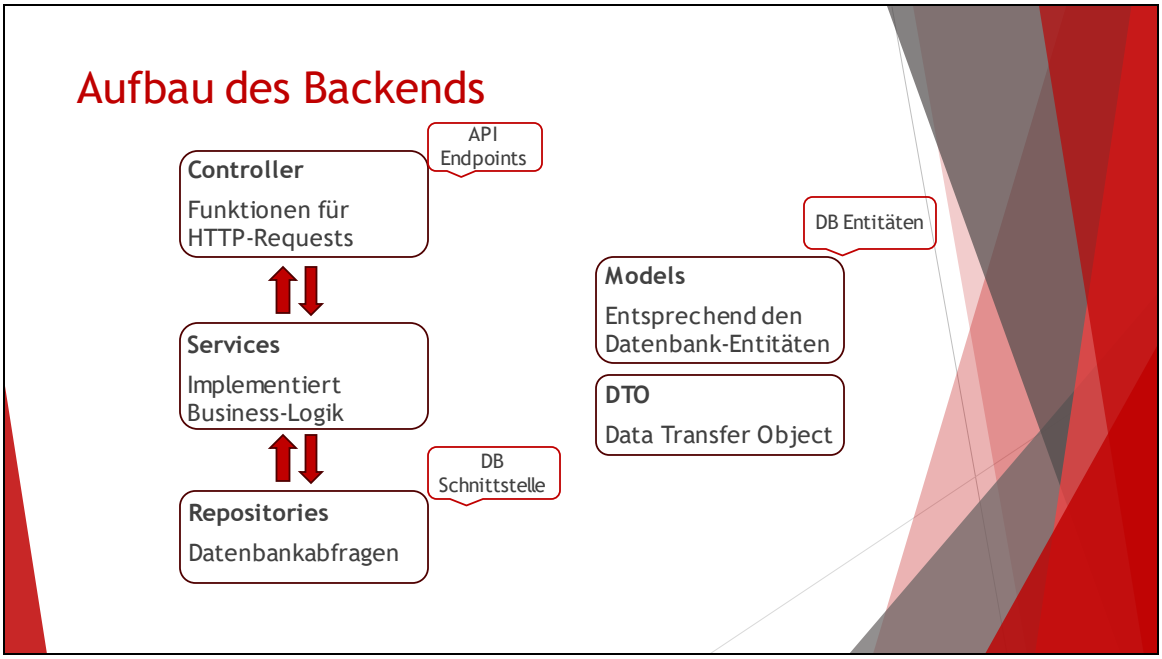




ER-Modell


Like
PK id UUID NOT NULL
FK1 user_id UUID NOT NULL
FK2 posting_id UUID NOT NULL
Comment
PK id UUID NOT NULL
FK1 user_id UUID NOT NULL
FK2 posting_id UUID NOT NULL
text VARCHAR(256) NOT NULL
created_at DATETIME NOT NULL
hashtags LIST NULL
User
PK id UUID NOT NULL
mail VARCHAR(n) NOT NULL UNQIUE
username VARCHAR(n) NOT NULL UNIQUE
nickname VARCHAR(n) NULL
passwordHash STRING NOT NULL
email_verified BOOL NOT NULL
profile_pic_link VARCHAR(n) NULL
Token
PK id UUID NOT NULL UNIQUE
FK1 user_id UUID NOT NULL
valid_until DATETIME NOT NULL
valid BOOL NOT NULL
Posting
PK id UUID NOT NULL
FK1 user_id UUID NOT NULL
text VARCHAR(256) NULL
created_at DATETIME NOT NULL
pic_link VARCHAR(n) NULL
hashtags LIST NULL
Subscription
PK id UUID NOT NULL
FK1 subscriber_id UUID NOT NULL
FK2 subscribed_id UUID NOT NULL

Aufbau des Backends



Build-Prozess - Makefile

- Makefile
- go.mod zum Verwalten aller Abhängigkeiten
  - Verwendete Module
  - Versionen

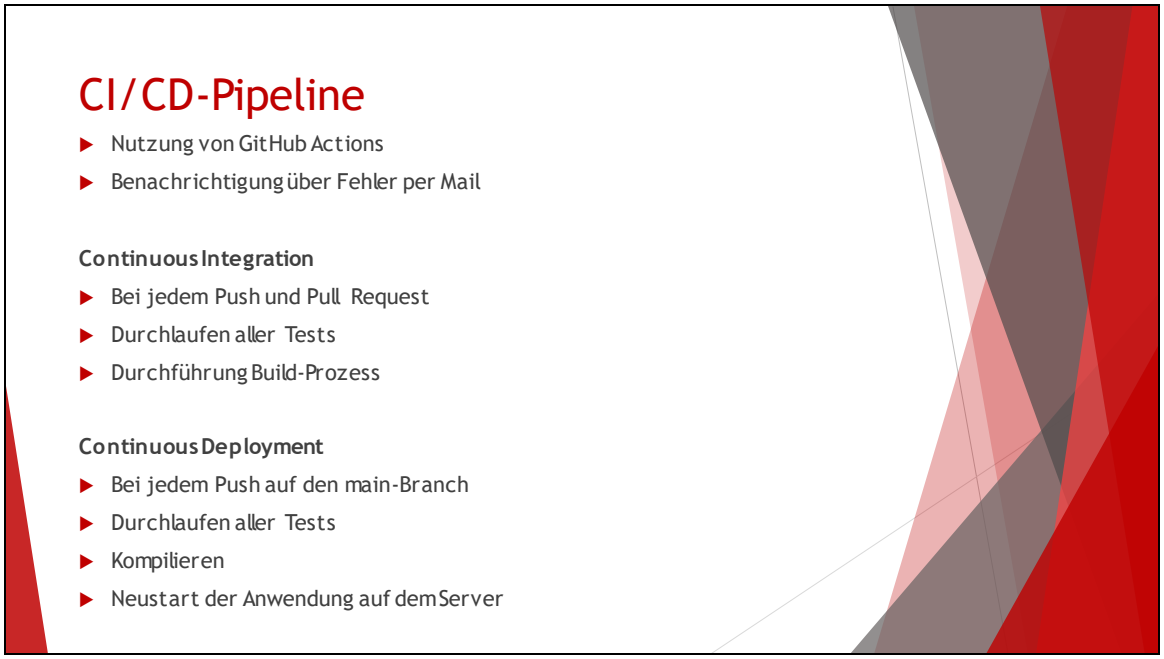
```
all: clean update build test

BUILD
go build –o ./bin/server-beta ./cmd/server-beta

CLEAN
rm –f ./bin/server-beta

UPDATE
go mod tidy

TEST
go test ./..
```



- Fehlerbehebung bis zum Ende der Iteration
- Wer untersucht Fehler?
    - CI: Derjenige der den Push/Pull Request angestoßen hat, ggf. Übertragung von Code-Änderungen auf andere Gruppen Mitglieder, wenn sie für den Fehler verantwortlich sind
    - CD: Marc

Danke für Ihre Aufmerksamkeit!

Gruppe

Lina Groth, 9220827
Klara Kulinna, 4379580
Jennifer Tielke, 6181218
Holger Theis, 4585051
Marc Buddemeier, 8466584