PS Softwareentwicklung und Projektmanagement
Sommersemester 2016

Konzept-Review

Reviewteam:

Daniel Linter: 1418148

Julia Wanker: 1315695

Softwareteam: Projekt 2 – Come&Go

Proseminargruppe: Gruppe 2

Datum: 04.04.2016

Hinweis: Gestalten Sie den Review in konstruktiver Art und Weise. Konkrete Vorschläge können anhand von Kommentaren und Vorschläge im Konzeptpapier selbst gemacht werden. Bitte beachten Sie, dass das Projektteam auf Ihren Review schriftlich Stellung nehmen muss.

1. Zusammenfassung

- *Fassen Sie das Systemkonzept in eigenen Worten zusammen*
  Die Qualität von Come&Go zeichnet sich durch responsitives Design und User-Freundlichkeit aus. Zudem ist das System sehr Leistungsfähig, auch bei Stoßzeiten. Das System wird stetig weiterentwickelt und in Stand gehalten.
  Jeder Mitarbeiter kann sich zu Beginn seines Arbeitstages einloggen und registriert den Beginn seiner Arbeitszeit. Jeder Mitarbeiter kann während des Arbeitstages Pausen eintragen. Am Ende des Tages kann der Mitarbeiter durch einen einfachen Klick seinen Arbeitstag beenden und sich ausloggen. Im Falle einer Krankheit bzw. eines Urlaubantrages kann jeder Mitarbeiter dies im System erfassen. Die Benachrichtigung wird automatisch an den Vorgesetzten gesendet. Jeder Mitarbeiter kann Anträge erstellen, dazu gehören Überstundenauszahlung, Urlaubsauszahlung und seinen eigenen Mitarbeiter-Bericht, welcher sämtliche Arbeitszeiten und Abwesenheiten beinhaltet.
  Der Boss ist zuständig für die Entscheidung aller gesendeten Anträge von Mitarbeitern – er kann diese akzeptieren bzw. ablehnen. Zudem kann der Boss über das System die Arbeitszeiten und mögliche Verstöße dagegen überprüfen und gegebenen falls einen Mitarbeiter feuern.
  Der Personal-Manager kann das Gehalt sowie die Arbeitsstunden eines jeden Mitarbeiters ändern. Zusätzlich kann er für die gesamte Firma eine Übersicht der Arbeitszeiten aller Mitarbeiter erstellen.
  Der Administrator kann für Mitarbeiter einen Account anlegen bzw. löschen und kann Mitarbeiterdaten ändern.

  Das System besteht aus drei Stakeholdern:
    - die Firma selbst mit all ihren Mitarbeitern, Personalchefs, dem Boss und dem Administrator
    - die Gesetzgebung, welche von der Firma eingehalten werden muss
    - die Systementwickler (Softwarearchitekt und -entwickler)

  Zur Entwicklung des Systems wird das MVC-Pattern herangezogen:
    - Models: Hauptklasse User, mit mehreren Assoziationen zu Klassen, welche die Funktionalität eines Mitarbeiters beschreiben.
    - View: ist zuständig für die Datenrepräsentation und überprüft die eingegeben Daten auf ihre Richtigkeit. Die Eingaben werden konvertiert und an die darunter liegenden Objekte geschickt.
    - Controller: Zu jeder View existiert ein eigener Controller um zwischen Models und View kommunizieren zu können.

  Entwicklungsprozess: Zur Entwicklung des Systems wird das Play-Framework verwendet. Software-Architekt und Software-Entwickler dokumentieren beide die Systementwicklung auf GitLab-Wiki. Der Software-Entwickler arbeitet an der Implementierung mit IntelliJ und schickt Änderungen an das GitLab-Repository. Der CI-Server bemerkt Änderungen und kompiliert das Projekt. Das Projekt wird an den Test-Server geschickt, wo der Entwickler das Projekt testen kann.

- *Fassen Sie das Ergebnis des Reviews kurz zusammen und gehen Sie dabei auf Stärken und Schwächen ein*
  Das Systemkonzept wurde sehr detailliert ausgearbeitet. Es sind alle Szenarien, Funktionen und Aufgaben abgedeckt. Die Architektur und der Systemaufbau sind übersichtlich strukturiert.

2. Systemüberblick

- *Ist die Zielgruppe klar definiert?*
  Ja. Die Aufgaben sind klar verteilt. Jede Gruppe hat eindeutig ihre Funktionen zugeteilt bekommen
- *Passen Zielgruppe, geplante Funktionalitäten und GUI Prototyp zusammen?*
  Ja.

3. Use Cases

- *Sind die Use Cases vollständig? Achten Sie dabei auch auf die Kriterien, die in der Vorlesung besprochen wurden.*
  Ja. Es wurden alle Akteure berücksichtigt und die Anwendungsfälle wurden korrekt zugewiesen.
- *Sind die Use Cases verständlich beschrieben oder gibt es Unklarheiten? Wenn ja, welche?*
  Bei den Reports (User-Report, Company-Report) wären Unterpunkte zur genaueren Klarstellung angebracht. Ansonsten sind alle Fälle klar und verständlich definiert.
- *Werden die in den Use Cases verwendeten fachlichen Begriffe konsistent verwendet?*
  Ja.

4. Fachliches Klassendiagramm

- *Ist das Klassendiagramm frei von technischen Konzepten, d.h. für Fachexperten verständlich?*
  Das Klassendiagramm ist klar und verständlich modelliert. Bei der Assoziation User-zu-User sollten die Akteure klar definiert werden.
- *Sind die Konzepte im Klassendiagramm konsistent mit den in den Use Case-Beschreibungen verwendeten Begriffen?*
  Ja.
- *Setzen Use Cases und Klassendiagramm die Vorgaben des Kollektivvertrags um?*
  Ja, die Vorgaben des Kollektivvertrags werden klar umgesetzt bzw. werden Abweichungen explizit angegeben (Bsp.: Verweigerung der Auszahlung des Urlaubs-Geldes).
- *Ist das Klassendiagramm modellierungstechnisch von guter Qualität?*
  Das Klassendiagramm ist stark vereinfacht, da die genaue Ausführung im Zusammenhang mit dem Kontextdiagramm modelliert wurde. Im Allgemeinen ist das Klassendiagramm gut modelliert.

5. GUI-Prototyp

- *Sind GUI-Prototyp, Use Cases und fachliches Klassendiagramm konsistent?*
  Ja.

- *Ist erkennbar, dass die Benutzeroberfläche Usability-Konzepte umsetzt? Wenn ja, welche?*
  Instant Gratification, Satisficing (außer bei den Spezial-Views für Boss, Personalchef und Administrator), Changes in midstream, Deferred Choices, Spatial Memory
- *Stellen die GUI-Prototypen die geforderten Informationen aus dem Kollektivvertrag dar?*
  Nicht ganz, da Urlaubsauszahlungen ablehnbar sind, dies ist aber lt. KV nicht erlaubt.

6. Projektplan

- *Sind die Verantwortlichkeiten klar verteilt?*
  Die Aufgaben sind klar verteilt. Die Verantwortlichen wurden für uns nicht offensichtlich zugewiesen.
- *Ist der Funktionsumfang der Inkremente verständlich und realistisch?*
  Ja.
- *Erscheint der Zeitplan realistisch?*
  Ja.