Dieses RDD Template stellt eine grobe Struktur des Dokuments dar.
Sie können es nach Belieben verändern.

Der rot-kursive Text soll durch Ihren Text ersetzt werden.

Bitte aktualisieren Sie bei jeder Meilenstein-Abnahme das Inhaltsverzeichnis.

Requirements and Design Documentation

(RDD)

Version xxx

SE2P – Praktikum – *Semester Jahr*

Name, Vorname, MatrikelNr, email
Name, Vorname, MatrikelNr, email
Name, Vorname, MatrikelNr, email
Name, Vorname, MatrikelNr, email

Änderungshistorie:

Version	Author	Datum	Anmerkungen
xxx	*xxx*	*xxx*	*xxx*

Inhalt

1. Motivation

Kurze Beschreibung der Aufgabenstellung und des Ziels.

Stakeholder ermitteln.

2. Randbedingungen

2.1. Entwicklungsumgebung

Auflistung der Entwicklungsumgebung (Simulator, Hardware, Betriebssystem etc.)

2.2. Werkzeuge

Auflistung von Werkzeugen inkl. ihrer Versionen

2.3. Sprachen

Auflistung der Programmiersprachen und Bibliotheken

3. Requirements und Use Cases

3.1. Anforderungen

Ermittelung aller Anforderungen an das System und die wichtigsten Anwendungsszenarien mit ihren Vor-/Nach- und Randbedingungen.

Unterscheidung von funktionalen und nicht-funktionalen Anforderungen. Erklären Sie, wie Sie diesen Anforderungen in Ihrer Realisierung gerecht werden wollen.

Mögliche Fehlbedienung und Fehlverhalten des Systems, Ermittlung von späteren Testfällen.

3.2. Use-Case-Diagramm

Spezifikation der Anforderungen in einem UML Use-Case-Diagram mit Use-Case-Details.

Darstellung des Grob-Verhaltens für die Anforderungen

4. Design

Anmerkung: Die Implementierung MUSS mit Ihrem Design-Modell korrespondieren. Daher ist ein wohlüberlegtes Design wichtig.

4.1. System Architektur

Erstellung der System-Architektur. Geben Sie eine kurze Beschreibung Ihrer Architektur mit den dazugehörenden Komponenten und Schnittstellen.

Spezifikation der Architektur und Definition der System-Schnittstellen in einem UML Komponentendiagramm.

4.2. Datenmodell

Bestimmung des Datenmodells mit Hilfe von UML Klassendiagrammen unter Beachtung der Designprinzipien.

Kurze textuelle Beschreibung des Datenmodells und deren wichtigsten Klassen und Methoden.

4.3. Verhaltensmodell

Spezifikation der wichtigsten System-Szenarien anhand von Verhaltensdiagrammen.

Sie können für die Spezifikation der Prozess-Lenkung entweder Petri-Netze oder hierarchische Automaten nehmen.

5. Implementierung

Anmerkung: Wichtige Implementierungsdetails sollen hier erklärt werden. Code-Beispiele (snippets) können hier aufgelistet werden, um der Erklärung zu dienen.

Anmerkung: Bitte KEINE ganze Programme hierhin kopieren!

5.1. Algorithmen

Wichtige Algorithmen, die Sie hier benutzt haben.

5.2. Patterns

Wichtige Patterns, die Sie implementiert haben.

5.3. Mapping Rules

Wichtige Mapping Rules, die Sie benutzt haben, z.B. um aus Ihrem Design entsprechenden Code zu erstellen.

6. Testen

Machen Sie sich Gedanken über Unit-Test, Komponententest, Integrationtest, Systemtest, Regressionstest und Abnahmetest.

6.1. Unit Test/Komponenten Test

Test Szenario eines Laufbands.

6.2. Integration Test/System Test

Test Szenarien mit beiden Laufbändern

6.3. Regressionstest

Welche Szenarien müssen immer wieder abgetestet werden? Automatisieren Sie Ihre Tests nach Möglichkeit

6.4. Abnahmetest

Leiten Sie die Abnahmebedingungen aus den Kunden-Anforderungen her.

Geben Sie an, welche Anforderungen erfolgreich und eventuell nicht erfolgreich implementiert sind.

6.5. Testplan

Zeitpunkte für die jeweiligen Teststufen in Ihrer Projektplanung setzen. Dazu können Sie die Meilensteine zu Hilfe nehmen.

6.6. Testprotokolle und Auswertungen

Hier fügen Sie die Test Protokolle bei, auch wenn Fehler bereits beseitigt worden sind, ist es schön zu wissen, welche Fehler einst aufgetaucht sind. Eventuelle Anmerkung zur Fehlerbehandlung kann für weitere Entwicklungen hilfreich sein.

Das letzte Testprotokoll ist das Abnahmeprotokoll, das bei der abschließenden Vorführung erstellt wird. Es enthält eine Auflistung der erfolgreich vorgeführten Funktionen des Systems sowie eine Mängelliste mit Erklärungen der Ursachen der Fehlfunktionen und Vorschlägen zur Abhilfe

7. Projektplan

7.1. Verantwortlichkeiten

Verantwortliche innerhalb des Projekts (Projektleiter, Tester, Implementierer, etc.) benennen.

7.2. PSP und Zeitplan

Projektstrukturplan, Ressourcenplan, Zeitplan, Abhängigkeiten von Arbeitspaketen, Eventueller Zeitverzug, etc.

8. Lessons Learned

Was lief gut, was lief schlecht in diesem Projekt (technisch und organisatorisch)?

Was haben Sie gelernt?

Weitere Anregungen und Erkenntnisse durch das Projekt.

Glossar

Eindeutige Begriffserklärungen

Abkürzungen

Listen Sie alle Abkürzungen auf, die Sie in diesem Dokument benutzt haben.

Anhänge

Auflistung aller Artefakten dieses Projekts

- Alle Modell-Dateien (Visual Paradigm, Petri-Netze etc.)
- Source Code und Code Dokumentationen (z.B. Doxygen)
- Test Protokolle
- Meeting Protokolle
- Projektplan
- etc.