<Company Name>

<Deskflex>
<Iteration/ Master> Test Plan

Version <1.0>

[Note: The following template is provided for use with the Rational Unified Process. Text enclosed in square brackets and displayed in blue italics (style=InfoBlue) is included to provide guidance to the author and should be deleted before publishing the document. A paragraph entered following this style will automatically be set to normal (style=Body Text).]

[To customize automatic fields in Microsoft Word (which display a gray background when selected), select File>Properties and replace the Title, Subject and Company fields with the appropriate information for this document. After closing the dialog, automatic fields may be updated throughout the document by selecting Edit>Select All (or Ctrl-A) and pressing F9, or simply click on the field and press F9. This must be done separately for Headers and Footers. Alt-F9 will toggle between displaying the field names and the field contents. See Word help for more information on working with fields.]

Revision History

Date	Version	Description	Author
<dd/mmm/yy>	<x.x>	<details>	<name>

Table of Contents

<Iteration/ Master> Test Plan

1. Introduction

1.1 Purpose

Der Zweck des Iterationstestplans besteht darin, alle notwendigen Informationen zu sammeln, um die Testaktivitäten für eine bestimmte Iteration zu planen und zu steuern. Er beschreibt den Ansatz für das Testen der Software. Dieser Testplan für Deskflex verfolgt folgende Ziele:

- Identifikation der Elemente, die durch die Tests abgedeckt werden sollen.
- Erläuterung der Motivation und der Überlegungen hinter den ausgewählten Testbereichen.
- Darstellung des gewählten Testansatzes.
- Bestimmung der benötigten Ressourcen sowie Schätzung des Testaufwands.

1.2 Scope

Dieser Testplan beschreibt Tests zur Sicherstellung der Funktionalität des Frontends, des Backends sowie der Kommunikation zwischen beiden Komponenten der Anwendung. Folgende Testarten werden in diesem Dokument berücksichtigt:

- Unit-Tests

Tests im Bereich Performance, Skalierbarkeit sowie Usability werden in diesem Plan ausdrücklich **nicht** behandelt.

1.3 Intended Audience

Dieser Testplan dient in erster Linie der internen Dokumentation.

1.4 Document Terminology and Acronyms

n/a

1.5 References

n/a

1.6 Document Structure

n/a

2. Evaluation Mission and Test Motivation

2.1 Background

Deskflex ist eine Anwendung zur effizienten Verwaltung von Sitzplatzbuchungen in Unternehmen. Mitarbeitende können damit flexibel Arbeitsplätze reservieren und verwalten. Das System basiert auf einer klar strukturierten Softwarearchitektur nach dem MVC-Muster für Frontend und Backend.

Tests sind ein zentraler Bestandteil des Entwicklungsprozesses, um sicherzustellen, dass Deskflex zuverlässig funktioniert und alle Anforderungen erfüllt. Durch frühzeitiges Erkennen von Fehlern werden Qualität und Stabilität verbessert und spätere Probleme vermieden.

Da Deskflex für die Organisation von Arbeitsplätzen eine wichtige Rolle spielt, ist eine umfassende Testabdeckung unerlässlich. Fehlerhafte Buchungen oder Systemausfälle könnten erhebliche Auswirkungen auf den Unternehmensablauf haben. Deshalb ist es entscheidend, dass alle Komponenten reibungslos zusammenarbeiten.

2.2 Evaluation Mission

Wie bereits erwähnt, hilft Testen dabei, Fehler, Defekte und Bugs frühzeitig im Entwicklungsprozess zu erkennen, was die Behebung einfacher und kostengünstiger macht. Testen ist somit ein wesentlicher Bestandteil des Entwicklungsprozesses.
Der Schwerpunkt der Testaktivitäten liegt auf der Identifikation kritischer Fehler, die die Nutzererfahrung beeinträchtigen könnten, sowie auf der frühzeitigen Erkennung potenzieller Qualitätsrisiken, die die Zuverlässigkeit der Anwendung gefährden könnten.

2.3 Test Motivators

Die folgenden Punkte bilden die Grundlage für die Testschwerpunkte in dieser Iteration:

1. **Qualitätsrisiken und Use Cases:**
   Das Testen stellt sicher, dass die Anwendung die gewünschten Qualitäts- und Funktionsanforderungen erfüllt. Gleichzeitig sollen Schwachstellen identifiziert werden, bei denen Verbesserungen notwendig sind. Der Fokus liegt darauf, die Bedürfnisse und Anforderungen der Nutzer vollständig abzudecken.
2. **Funktionale Anforderungen:**
   Die Tests dienen dazu, zu überprüfen, ob die Anwendung sämtliche funktionalen Anforderungen erfüllt. Gleichzeitig werden Optimierungspotenziale identifiziert, um die Funktionalität der Anwendung weiter zu verbessern.

3. Target Test Items

Die folgende Liste identifiziert die Testobjekte: Software, Hardware und unterstützende Produktelemente, die als Ziele für die Tests festgelegt wurden. Diese Liste gibt an, welche Elemente getestet werden:

- **Server Backend**

4. Outline of Planned Tests

4.1 Outline of Test Inclusions

Backend: C# ASP.NET-Anwendung:

- Unittesting

Die Tests selbst werden nicht getestet und fließen nicht in die Code-Abdeckmailungsanalyse ein.

4.2 Outline of Other Candidates for Potential Inclusion

n/a

4.3 Outline of Test Exclusions

Aufgrund von Zeit- und Ressourcenbeschränkungen werden wir folgende Tests nicht durchführen:

- Stresstests
- Datenbanktests
- Usability-Tests

5. Test Approach

5.1 Initial Test-Idea Catalogs and Other Reference Sources

n/a

5.2 Testing Techniques and Types

5.2.1 Unit Testing

Unittests stellen sicher, dass der getestete Quellcode wie erwartet funktioniert. Dabei werden kleine Teile des Quellcodes unabhängig voneinander getestet.

Technique Objective:	Sicherstellen, dass jede Codeeinheit (Funktionen, Methoden, Klassen) wie vorgesehen funktioniert.
Technique:	Die genaue Vorgehensweise zur Implementierung und Durchführung der Unittests wird festgelegt.
Oracles:	Die Methode zur Überprüfung und Validierung der Testergebnisse wird definiert, z.B. durch Vergleich der Testergebnisse mit den erwarteten Ausgaben.
Required Tools:	Die spezifischen Tools und Bibliotheken, die für das Unittesting benötigt werden (z.B. Frameworks, Testbibliotheken, CI/CD-Tools), werden noch festgelegt.
Success Criteria:	Alle Tests bestehen. Die Abdeckungsquote liegt bei mindestens 20 % (Backend).
Special Considerations:	Die Anforderungen und die Abdeckung können je nach Projektfortschritt und Prioritäten variieren.

6. Entry and Exit Criteria

6.1 Test Plan

6.1.1 Test Plan Entry Criteria

n/a

6.1.2 Test Plan Exit Criteria

n/a

7. Deliverables

7.1 Test Evaluation Summaries

Das Projekt verfügt über Unittests im Backend

7.2 Reporting on Test Coverage

Für die Testabdeckung wurde kein spezielles Tool eingesetzt. Die Abdeckung wurde nicht automatisiert gemessen.

7.3 Perceived Quality Reports

Es wurden keine automatisierten Tools zur Codequalitätsanalyse verwendet. Die Qualitätssicherung erfolgte ausschließlich durch Unittests im Backend.

8. Testing Workflow

Lokales Testen in der IDE

9. Environmental Needs

9.1 Base System Hardware

The following table sets forth the system resources for the test effort presented in this *Test Plan*.

System Resources		
Resource	**Quantity**	**Name and Type**
Lokales Testgerät	1	Notebook

9.2 Base Software Elements in the Test Environment

The following base software elements are required in the test environment for this *Test Plan*.

Software Element Name	Type and Other Notes
Visual Studio Code, Visual Studio	Test Runner / IDE
XUnit	Unit testing library

10. Responsibilities, Staffing, and Training Needs

10.1 People and Roles

Human Resources		
Role	**Minimum Resources Recommended** (number of full-time roles allocated)	**Specific Responsibilities or Comments**
Testmanager	Christoph, Jannik	• Übernehmnen das Management und die Aufsicht.
Testdesigner	Moritz, Leon	• Definieren die technische Herangehensweise für die Umsetzung der Testaktivitäten.
Test System Administrator	Hagen	• Stellt sicher, dass die Testumgebung und Ressourcen verwaltet und gepflegt werden.

10.2 Staffing and Training Needs

n/a

11. Iteration Milestones

Wir wollen 20% Testabdeckung erreichen.

12. Risks, Dependencies, Assumptions, and Constraints

Risk	Mitigation Strategy	Contingency (Risk is realized)
Code hat viele Seiteneffekte	Code refaktorisieren (Clean Code Prinzipien)	• Neue refaktorisierte Tests veröffentlichen
Test Runner kann Tests nicht ausführen	Standardbibliotheken verwenden, die einen funktionierenden Test Runner enthalten	• Testausführungskonfiguration reparieren

13. Management Process and Procedures

n/a