IPCA-EST-MEI-PCC

<Insert Project Name>

Software Architecture Document (SAD)

CONTENT OWNER

<owner name>

|  |  |  |
| --- | --- | --- |
| DOCUMENT NUMBER: | RELEASE/REVISION: | RELEASE/REVISION DATE: |
|  |  |  |
|  |  |  |
|  |  |  |

|  |
| --- |
| **BACKGROUND (Delete this page)**  This template is based on the Software Engineering Institute’s “View and Beyond” method for documenting software architectures, as described in Clements, et al., [*Documenting Software Architecture: Views and Beyond*](http://www.sei.cmu.edu/architecture/books.html) (Addison Wesley, 2002), as well as in *Software Architecture for Developers* (Brown, 2019).  **TIPS FOR USING THIS TEMPLATE**  To create an instance of this document:  Insert relevant information on cover sheet and in placeholders throughout.  Insert relevant information in page header: Move to a page of the body of the report, select *View > Header and Footer* from the main menu, and then replace relevant information in the header box at the top of the page.  To update the contents and page numbers in the Table of Contents, List of Figures, and List of Tables:  Position the cursor anywhere in the table to be updated.  Click the *F9* function key.  Answer “Update entire table”.  To insert a figure or table caption:  From the main menu, choose *Insert > Reference > Caption* and then either *Figure* or *Table* as needed.  Click the OK button.  Add a colon and a tab stop after the figure number in the caption itself.  The caption should use the *Caption* style.  Add a colon and a tab stop after the table/figure number in the caption itself.  **TIPS FOR MAKING YOUR DOCUMENT MORE READABLE**  A gray box containing *CONTENTS OF THIS SECTION* is provided at the beginning of most sections and subsections. After determining what specific information will be included in your document, you can remove this gray box or leave it to serve as a quick-reference section overview for your readers. In the case that text has been provided in the template, inspect it for relevance and revised as necessary.  Consider hyperlinking key words used in the document with their entries in the [Glossary](#_Glossary) or other location in which they are defined. Choose *Insert > Hyperlink*.  Don’t leave blank sections in the document. Mark them “To be determined” (ideally with a promise of a date or release number by which the information will be provided) or “Not applicable.”  Consider packaging your SAD as a multi-volume set of documentation. It is often helpful to break your documentation into more than one volume so that the document does not become unwieldy. There are many ways that this can be accomplished. The structuring of the document must support the needs of the intended audience and must be determined in the context of the project. Each document that you produce should include the date of issue and status; draft, baseline, version number, name of issuing organization; change history; and a summary. A few decomposition options are:  *A 2-Volume approach:* Separate the documentation into two volumes; one that contains the views of the software architecture and one that contains everything else. A common variant of this approach has one volume per view, and one volume for everything else.  *A 3-Volume approach:* Document organizational policies, procedures, and the directory in one volume, system specific overview material in a second, and view documentation in a third.  *A 4-Volume approach:* Create one volume for each viewtype (Brown, 2019) (module, component-and-connector, allocation) that contains the documentation for the relevant [views](#view). Include all of the other information in the fourth volume.  Software interfaces are often documented in a separate volume.  In *any* case, the information should be arranged so that readers begin with the volume containing the Documentation Roadmap (Section 1 in this template). |

Table of Contents

[1 Introduction and Goals 3](#_Toc125153349)

[1.1Requirements Overview 3](#_Toc125153350)

[1.2Quality goals 3](#_Toc125153351)

[1.3Stakeholders 3](#_Toc125153352)

[2 Constraints 5](#_Toc125153353)

[2.1Technical Constraints 5](#_Toc125153354)

[2.2Organizational Constraints 5](#_Toc125153355)

[2.3Conventions 5](#_Toc125153356)

[3 Context 6](#_Toc125153357)

[3.1Business Context 6](#_Toc125153358)

[3.2Deployment Context 6](#_Toc125153359)

[4 Solution strategy 7](#_Toc125153360)

[4.1Introduction to the Strategy 7](#_Toc125153361)

[4.2Structure of the system 7](#_Toc125153362)

[4.3System Strategy 7](#_Toc125153363)

[4.4The Connection of the System 7](#_Toc125153364)

[5 Building Block View 8](#_Toc125153365)

[5.1Level 1 8](#_Toc125153366)

[5.2Level 2 8](#_Toc125153367)

[6 Runtime View 9](#_Toc125153368)

[6.1“Feature X” 9](#_Toc125153369)

[7 Deployment View 10](#_Toc125153370)

[7.1Operating System Infrastructure (Windows, IOS, others) 10](#_Toc125153371)

[8 Technical and crosscutting concepts 11](#_Toc125153372)

[8.1Dependencies Between Modules 11](#_Toc125153373)

[8.2Domain Model 11](#_Toc125153374)

[8.3User Interface 11](#_Toc125153375)

[8.4Exception and Error Handling 11](#_Toc125153376)

[8.5Logging and Tracing 11](#_Toc125153377)

[8.6Testability 11](#_Toc125153378)

[9 Design Decisions 12](#_Toc125153379)

[9.1How Does the Engine Communicate with the Outside World? 12](#_Toc125153380)

[9.2Are Position Objects Changeable or Not? 12](#_Toc125153381)

[10 Quality Requirements 13](#_Toc125153382)

[10.1Utility Tree 13](#_Toc125153383)

[10.2Quality Scenarios 13](#_Toc125153384)

[11 Glossary 14](#_Toc125153385)

[11.1Terms 14](#_Toc125153386)

[12 Source Code 15](#_Toc125153387)

[13 Sample Figures & Tables 16](#_Toc125153388)

List of Figures

[Figure 1: Sample Figure 22](#_Toc51125206)

List of Tables

[Table 1 - Quality Goals 3](#_Toc125153420)

[Table 2 - Stackeholders 4](#_Toc125153421)

[Table 3 - Technical constraints 5](#_Toc125153422)

[Table 4 - Organizational constraints 5](#_Toc125153423)

[Table 5 - Conventions 5](#_Toc125153424)

[Table 6 - Quality goals and Architectures 7](#_Toc125153425)

[Table 7 - Logical components cescription – Level 1 8](#_Toc125153426)

[Table 8 - Logical sub-components description – Level 2 8](#_Toc125153427)

[Table 9 - Quality Scenarios 13](#_Toc125153428)

[Table 10 – Glossary terms 14](#_Toc125153429)

[Table 11: Sample Table 16](#_Toc125153430)

# Introduction and Goals

Atualmente, em Portugal a obesidade e o excesso de peso é um problema que tem vindo a aumentar com o decorrer dos anos. Estima-se que cerca de 50% da população portuguesa tenha excesso de peso ou obesidade e que 30% das crianças tenham o mesmo problema. O objetivo da nossa aplicação é promover o exercício físico fornecendo uma plataforma que facilita o agendamento de uma atividade que a maioria dos portugueses gosta de fazer: jogar futebol.

## Requirements Overview

Antes de iniciar o desenvolvimento de uma aplicação, é importante definir claramente o que se espera da aplicação e como ela deve funcionar. Isso inclui considerações sobre o público-alvo, as funcionalidades desejadas, as restrições técnicas e os requisitos de desempenho.

Nesta secção, discutiremos os principais requisitos da aplicação.

### Requisitos funcionais

Os requisitos funcionais são uma parte fundamental do processo de desenvolvimento de software. São utilizados para definir as funcionalidades que a aplicação deve ter e servem como uma ponte entre as expectativas dos utilizadores e a implementação técnica.

Para a Kickoff foram definidos os seguintes requisitos funcionais:

* RF01 - Serviço de autenticação – A aplicação deve possuir um serviço de autenticação com registo, login e recuperação de palavra-passe.
* RF02 – Criar equipa – A aplicação deve possuir a funcionalidade de criação de equipas.
* RF03 – Editar equipa – A aplicação deve possuir a funcionalidade de edição de equipas. Isto implica, alterar as informações básica da equipa (nome, foto) e gestão de membros da equipa.
* RF04 – Listagem de equipas – A aplicação deve possuir a funcionalidade de listagem de equipas.
  + RF05 – Criar publicações – A aplicação deve permitir que os utilizadores consigam escrever publicações.
  + RF06 – Editar publicações – A aplicação deve permitir a edição do conteúdo das próprias publicações.
  + RF07 – Remover publicações – A aplicação deve permitir ao utilizador remover as próprias publicações.
  + RF08 – Agendar jogos – A aplicação deve permitir o agendamento de jogos de futebol.
  + RF09 – Editar jogos – A aplicação deve permitir a edição de jogos agendados durante um certo período.
  + RF10 – Listar jogos – A aplicação deve permitir a listagem dos jogos de futebol a que um certo utilizador está associado.
  + RF11 – Adicionar amigos – A aplicação deve possuir a funcionalidade de adicionar outros utilizadores como amigos.
  + RF12 – Remover amigos – A aplicação deve possuir a funcionalidade de remover a amizade de utilizadores adicionados previamente.
  + RF13 – Listar amigos – A aplicação deve permitir ao utilizador listar todos os seus amigos.
  + RF14 – Editar perfil – A aplicação deve permitir ao utilizador editar as suas informações básicas.
  + RF15 – Notificações – A aplicação deve possuir um conjunto de notificações que aparecem após o utilizador realizar uma certa ação.

### Requisitos não funcionais

Os requisitos não funcionais são especificações que descrevem aspetos da aplicação que não são diretamente relacionados às suas funcionalidades, mas que afetam o seu desempenho, segurança, usabilidade, compatibilidade e escalabilidade.

* RNF01 – Disponibilidade - A aplicação deve estar disponível 24 horas por dia, 7 dias por semana.
* RNF02 – Segurança - A aplicação deve ser segura e proteger os dados pessoais de todos os seus utilizadores.
* RNF03 – Desempenho - A aplicação deve ser rápida a carregar e responsiva ao toque do utilizador.
* RNF04 – Escalabilidade - A aplicação deve ser escalável para lidar com um grande número de utilizadores.
* RNF05 – Compatibilidade - A aplicação deve ser compatível com diferentes dispositivos e sistemas operativos.
* RNF06 – Usabilidade - A aplicação deve ser fácil de utilizar e navegar.

## Quality goals

A definição de metas de qualidades são uma das principais preocupações em qualquer projeto, especialmente em aplicações para dispositivos móveis e web. As metas de qualidade definem os padrões a serem seguidos e as expectativas a serem atendidas ao desenvolver uma aplicação. Estas metas ajudam a garantir que a aplicação seja fácil de usar, confiável, segura e escalável, entre outras coisas.

No caso de uma aplicação para agendar jogos amigáveis, as metas de qualidade são ainda mais importantes, pois é fundamental que a aplicação funcione de forma eficiente e sem falhas, para garantir a satisfação dos utilizadores. Além disso, é importante levar em consideração questões como segurança de dados, privacidade e compatibilidade com diferentes tipos de dispositivos, com a finalidade de proteger as informações pessoais dos utilizadores e garantir a melhor experiência possível.

Nesta seção, discutiremos as metas de qualidade que devem ser consideradas ao desenvolver uma aplicação para agendar jogos amigáveis.

|  |  |
| --- | --- |
| Quality Goal | Motivation/description |
| Usabilidade | A aplicação deve ser intuitiva e fácil de usar, com um design simples e elegante. |
| Confiabilidade | A aplicação deve funcionar de forma confiável, sem bugs ou falhas frequentes. |
| Desempenho | A aplicação deve ser rápida e responsiva, com tempo de carregamento rápido. |
| Segurança | A aplicação deve proteger as informações pessoais dos utilizadores de acordo com as melhores práticas de segurança. |
| Escalabilidade | A aplicação deve ser projetada para escalar com o crescimento do número de utilizadores, sem comprometer a performance. |
| Compatibilidade | A aplicação deve ser compatível com diferentes tipos de dispositivos, incluindo smartphones, tablets e computadores desktop. |
| Testabilidade | A aplicação deve ser projetada para ser facilmente testável, para que possa ser melhorada e corrigida rapidamente. |
| Facilidade de manutenção | A aplicação deve ser projetada de forma a ser fácil de manter e atualizar, sem necessidade de grandes esforços. |

Table 1 - Quality Goals

## Stakeholders

Os stakeholders são as pessoas ou grupos que têm interesses ou impactos no projeto ou empresa. Os stakeholders incluem desde os usuários finais até os investidores, passando por desenvolvedores, empresas de tecnologia, autoridades regulatórias e parceiros comerciais.

Identificar e compreender os stakeholders é fundamental para o sucesso de um projeto. Cada stakeholder tem necessidades e expectativas diferentes, e é importante garantir que essas necessidades sejam atendidas durante o desenvolvimento da aplicação. Além disso, é importante estabelecer uma comunicação clara e efetiva com cada stakeholder para garantir que todas as questões relevantes sejam abordadas e resolvidas de maneira adequada.

No caso de uma aplicação para agendar jogos amigáveis, os stakeholders incluem utilizadores finais, desenvolvedores, proprietários de campos de futebol, autoridades reguladoras e a comunidade em geral. Nesta seção, discutiremos os principais stakeholders de uma aplicação para agendar jogos amigáveis e exploraremos as suas necessidades e expectativas.

|  |  |
| --- | --- |
| Who? | Matters and concern |
| Developer | São responsáveis por criar e manter a aplicação, garantindo que atenda às necessi-dades e expectativas dos utilizadores. |
| Utilizadores | São os jogadores que usam a aplicação para agendar e jogar jogos amigáveis com seus amigos. |
| Parceiros comerciais | São empresas que trabalham em parceria com a aplicação para oferecer produtos ou serviços adicionais aos utilizadores. |
| Autoridades reguladoras | São responsáveis por regulamentar as aplicações, incluindo questões de segurança de dados, privacidade, etc. |
| Proprietários de campos de futebol | São os proprietários de campos de futebol que podem ser alugados pelos utilizadores da aplicação para jogar jogos amigáveis |
| Comunidade | São pessoas interessadas na aplicação, incluindo jogadores, entusiastas de desportos e defensores da vida saudável. |

Table 2 - Stackeholders

# Constraints

As restrições são fatores que limitam ou influenciam o desenvolvimento, estas incluem considerações técnicas, legais, de tempo e de recursos. É importante identificar e compreender as restrições desde o início do projeto para garantir que o projeto seja realizável e alcance seus objetivos.

As restrições técnicas são especialmente importantes pois podem afetar diretamente a funcionalidade e o desempenho da aplicação. Essas restrições incluem limitações no hardware, no software e na conectividade de rede, entre outras.

Nesta seção, discutiremos as restrições técnicas específicas da aplicação.

## Technical Constraints

|  |  |
| --- | --- |
| Constraint | Background and/or motivation |
| Hardware | A aplicação deve ser compatível com diferentes tipos de dispositivos, como smartphones, tablets e computadores. Isso significa que a aplicação deve ser otimizada para diferentes tamanhos de tela e configurações de hardware. |
| Software | A aplicação deve funcionar em diferentes sistemas operacionais, incluindo iOS, Android e Windows. Além disso, a aplicação deve ser compatível com diferentes versões dos sistemas operacionais para garantir a melhor experiência possível para os usuários. |
| Rede | A aplicação deve ser capaz de funcionar de forma eficiente em diferentes níveis de conectividade de rede, incluindo conexões Wi-Fi e dados móveis. |
| Segurança e privacidade | A aplicação deve proteger as informações pessoais dos utilizadores, incluindo nomes, endereços de e-mail, etc. Além disso, a aplicação deve ser projetada para ser segura contra-ataques cibernéticos e outras ameaças à segurança. |
| Integração com outros sistemas | A aplicação deve ser capaz de se integrar a outros sistemas, como calendários, mensagens de texto e notificações push, para fornecer uma experiência ao utilizador completa e integrada. |

Table 3 - Technical constraints

## Organizational Constraints

As restrições organizacionais são fatores externos que afetam o desenvolvimento de uma aplicação. No caso da aplicação a ser desenvolvida, algumas restrições organizacionais importantes a serem consideradas:

|  |  |
| --- | --- |
| Constraint | Background and/or motivation |
| Orçamento | O orçamento disponível para o projeto pode afetar a escala e o alcance da aplicação. Por exemplo, se o orçamento for limitado, pode ser necessário limitar o número de recursos incluídos na aplicação. |
| Prazo | O prazo para o projeto pode ser uma restrição importante, especialmente se houver uma data específica para o lançamento da aplicação. É importante garantir que o projeto esteja no cronograma e que as atividades sejam concluídas dentro do prazo estabelecido. |
| Recursos | A disponibilidade de recursos, incluindo equipe de desenvolvimento, pode afetar o desenvolvimento da aplicação. Por exemplo, se a equipe for pequena, pode ser necessário limitar o número de recursos incluídos na aplicação. |

Table 4 - Organizational constraints

## Conventions

|  |  |
| --- | --- |
| Constraint | Background and/or motivation |
| Design | Existem convenções de design para aplicativos móveis, incluindo padrões de navegação, cores e tipografia. É importante considerar essas convenções para garantir que a aplicação seja fácil de usar e atenda às expectativas do utilizador. |
| Tecnologia | Algumas tecnologias podem ser consideradas padrão em uma indústria ou organização. É importante considerar essas tecnologias ao desenvolver a aplicação, pois elas podem ser mais fáceis de implementar e manter. |
| Regulamentos e leis | Algumas regulamentações e leis podem estabelecer convenções para a coleta e armazenamento de dados. É importante garantir que a aplicação atenda a essas regulamentações e leis. |
| Padrões de segurança | Alguns padrões de segurança são aceitos como convenção em uma indústria ou organização. É importante considerar esses padrões ao desenvolver a aplicação para garantir a segurança dos dados do utilizador. |

Table 5 - Conventions

# Context

A aplicação a desenvolver permitirá a marcação de jogos de futebol com mais facilidade. Esta facilidade será garantida através das várias funcionalidades que a aplicação disponibilizará, das quais, um chat geral onde os utilizadores podem conhecer novas pessoas e porventura combinar jogos entre si. O objetivo é que este chat permita, então, que utilizadores descubram outros utilizadores dispostos a marcar jogos de futebol e por conseguinte enviar-lhes um pedido de amizade.

Com o referido anteriormente, os utilizadores podem enviar pedidos de amizade uns aos outros e, com isto, podem então adicioná-los a uma equipa ou agendar um jogo de futebol entre equipas. Cada equipa terão um capitão (administrador) que pode convidar novos utilizadores para fazer parte da equipa, remover utilizadores e agendar jogos de futebol entre equipas.

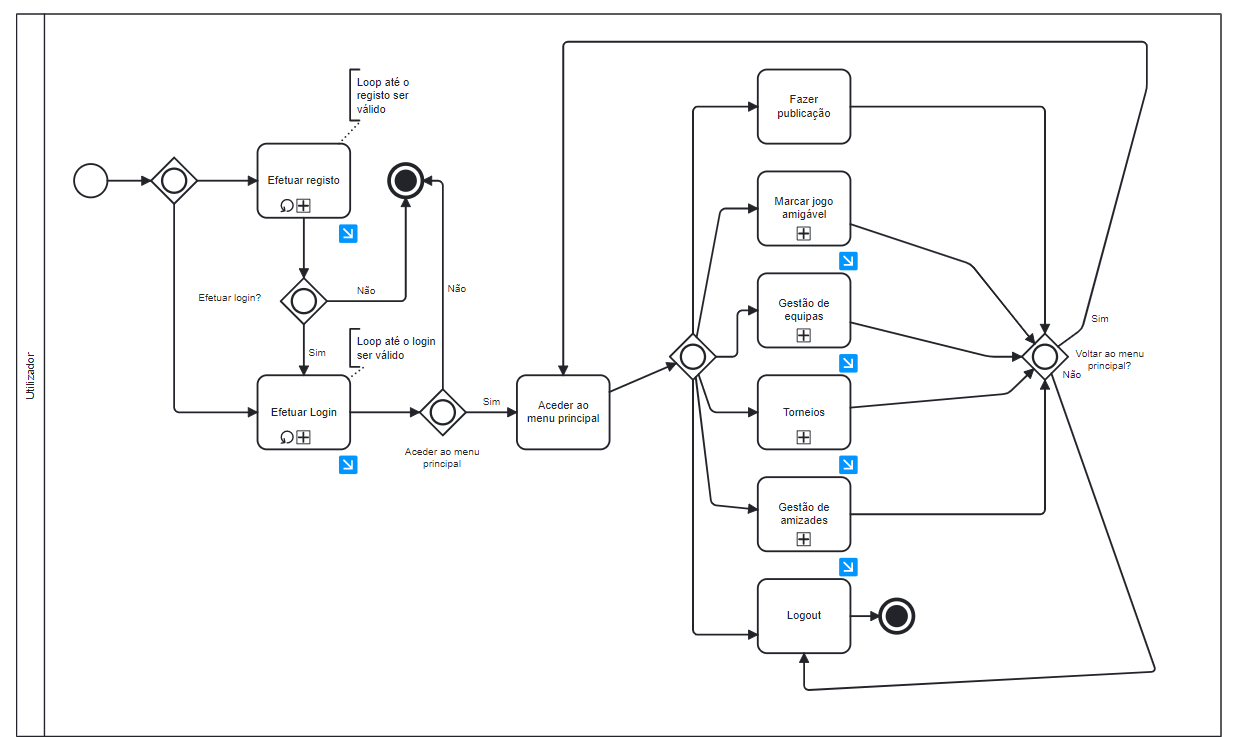
O agendamento de jogos é feito através de um menu onde o capitão propõe um jogo contra uma respetiva equipa e todos os jogadores de ambas as equipas serão notificados e devem aceitar ou recusar o jogo.

## Business Context

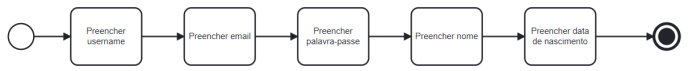
O contexto de negócios é uma consideração crucial ao desenvolver uma aplicação para agendar jogos amigáveis. Ele inclui aspetos como o mercado-alvo, a concorrência, a posição da empresa no mercado e o modelo de negócios. Compreender o contexto de negócios permite-nos desenvolver uma aplicação que atenda às necessidades do mercado e seja competitiva em relação a outras aplicações similares. Além disso, o conhecimento do contexto de negócio permite-nos concentrar nas áreas mais importantes da aplicação e evitar problemas desnecessários.

Para melhor compreensão do negócio foi elaborado um modelo bpmn como mostra na figura \*.

### Visão geral



### Registo

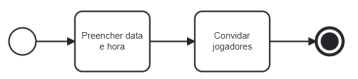


### Login

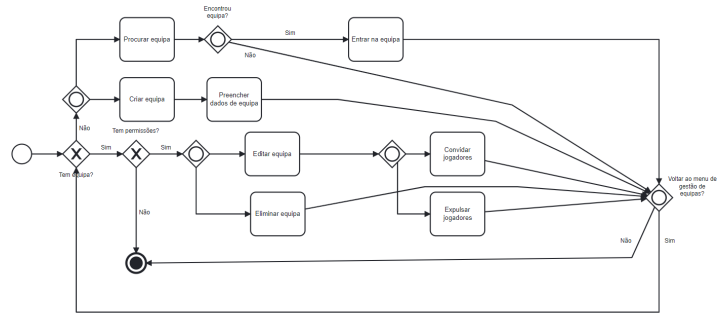
Uma imagem com texto

Descrição gerada automaticamente

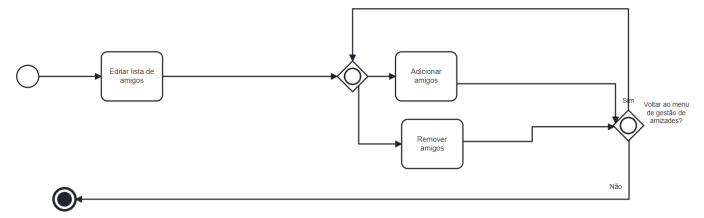
### Marcação de jogos amigáveis



### Gestão de Equipas



### Gestão de Amizades



## Deployment Context

Technical communication of the system with third parties

# Solution strategy

(This section contains a highly compact architecture overview. A contrast of the most important goals and approaches)

## Introduction to the Strategy

The following table contrasts the quality goals of the system ([Section 1.2](https://www.dokchess.de/en/01_introduction/02_qualitygoals/)) with matching architecture approaches and thus provides easy access to the solution.

|  |  |
| --- | --- |
| Quality Goal | Matching approaches in the solution |
| Interoperability |  |
| Efficiency |  |
| …. |  |

Table 6 - Quality goals and Architectures

## Structure of the system

## System Strategy

## The Connection of the System

# Building Block View

(This section describes the decomposition of the system into modules or components. These are also reflected in the package structure of the source code)

## Level 1

(Main Components of the system and their relations)

Logical Architecture (Diagram)

|  |  |
| --- | --- |
| **Component** | **Short description** |
|  |  |
|  |  |
|  |  |

Table 7 - Logical components cescription – Level 1

## Level 2

(Sub-components of and their relations of a particular component)

Logical Architecture (Diagram)

|  |  |
| --- | --- |
| **Component** | **Short description** |
|  |  |
|  |  |
|  |  |

Table 8 - Logical sub-components description – Level 2

# Runtime View

(In contrast to the static building block view, this section visualizes dynamic aspects. How do the pieces play together?)

## “Feature X”

Sequence Diagram of “Feature X”

# Deployment View

(This view describes the operation of the system. Here is an explanation of how to configure the system in conjunction with a graphical frontend)

## Operating System Infrastructure (Windows, IOS, others)

Deployment Diagram

Software requirements

# Technical and crosscutting concepts

(This section describes general structures and system-wide aspects. It also presents various technical solutions.)

## Dependencies Between Modules

Components (ex: Microservices)

Communications (ex: API Gateway)

Security

## Domain Model

(This section contains a brief overview of data structures and their relationships, that sustain the modules interaction. All the classes and enumeration types (enums) must be referred)

ERD Diagram

## User Interface

Mockups

## Exception and Error Handling

## Logging and Tracing

## Testability

# Design Decisions

(This section enables you to understand two fundamental design decisions of the system in detail. Optional Section)

## How Does the Engine Communicate with the Outside World?

### Problem Background

### Influences on the Decision

### Assumptions

### Considered Alternatives

### Decision

## Are Position Objects Changeable or Not?

### Problem Background

### Influences on the Decision

### Assumptions

### Considered Alternatives

### Decision

# Quality Requirements

(The quality scenarios in this section depict the fundamental quality goals (section 1.2) as well as other required quality properties. They allow the evaluation of decision alternatives.)

## Utility Tree

(Overview of the relevant quality attributes and their associated scenarios)

## Quality Scenarios

|  |  |
| --- | --- |
| ID | Scenario |
|  |  |
|  |  |
|  |  |

Table 9 - Quality Scenarios

# Glossary

## Terms

|  |  |
| --- | --- |
| Term | Definition |
|  |  |
|  |  |
|  |  |

Table 10 – Glossary terms

# Source Code

GitHub

# Sample Figures & Tables

CONTENTS OF THIS SECTION: Examples of figures and tables. Delete this page.



Figure 1: Sample Figure

Table 11: Sample Table

| Table Heading | Table Heading | Table Heading | Table Heading |
| --- | --- | --- | --- |
| Table Body | Table Body | Table Body | Table Body |
| Table Body | Table Body | Table Body | Table Body |
| Table Body | Table Body | Table Body | Table Body |
| Table Body | Table Body | Table Body | Table Body |

1. Appendices

|  |
| --- |
| CONTENTS OF THIS SECTION: Appendices may be used to provide information published separately for convenience in document maintenance (e.g., charts, classified data, API specification). As applicable, each appendix is referenced in the main body of the document where the data would normally have been provided. Appendices may be bound as separate documents for ease in handling. If your SAD has no appendices, delete this page. |

* 1. Heading 2 – Appendix title
  2. Heading 2 – Appendix tile