Software Detailed Design Approval:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Issued by:  DEV |  | < Author > |  |  |  | < Date > |  |
|  | Name printed |  | Signature |  | Date |  |
| Revision No.: |  | | | | | | |
| Approved by  PL: |  | < Name > |  |  |  |  |  |
|  | Name printed |  | Signature |  | Date |  |
| Approved by  TEST: |  | < Name > |  |  |  |  |  |
|  | Name printed |  | Signature |  | Date |  |
| Approved by  QA: |  | < Name > |  |  |  |  |  |
|  | Name printed |  | Signature |  | Date |  |
| Approved by  IT: |  | < Name > |  |  |  |  |  |
|  | Name printed |  | Signature |  | Date |  |

< Project Number, Project Name >

Knee Balancer application

< Version Number >

< Document ID: >

BLUE TEXT is provided for explanation, description or examples for each section, as applicable. All BLUE TEXT is to be deleted prior to the approval of the document.

Table of Contents

1 Introduction 3

1.1 Purpose 3

1.2 Scope 3

1.3 Definitions, acronyms and abbreviations 4

1.4 References 4

2 System / Component Context 4

3 System / Component Decomposition 6

3.1 Module Decomposition 6

3.1.1 Frontend Module 6

3.1.2 Backend Module 8

3.2 Concurrent Process Decomposition 8

3.3 Data Decomposition 8

4 Interfaces 10

4.1 Interface with external hardware components 10

4.2 Interface with external software components 10

4.3 Interface between modules 10

5 Detailed Design 10

5.1 various UML diagram types (class diagrams, collaboration diagrams, state transition diagrams), 10

5.1.1 Home Module 10

5.1.2 Case input module 11

5.1.3 Case Solution Module 13

5.1.4 User Preference module 15

5.1.5 Case Identifier module 17

5.2 pseudo code to outline algorithms, 18

5.3 flowcharts to visualize control flow or data flow, 18

5.3.1 New Preference: 18

5.3.2 Clone Preference: 18

5.3.3 Edit Preference: 19

5.3.4 Solution : 19

5.3.5 Camera capture: 20

5.4 SOUP items 20

5.5 Module 1 – Initial Position 20

5.6 Module2 – User preference 21

5.7 Module3 - Solution 21

5.8 Upload Module 21

5.8.1 Backend -Cloud 21

5.9 Authentication Module 23

6 Detailed Process Design 23

6.1 Process1 23

6.2 Process2 23

7 Detailed Data Design 24

7.1 Internal details of the data design. The design description might include 24

7.1.1 ER diagrams 24

7.1.2 pseudo code to outline SQL queries (especially for medium-high risk products/modules), 24

7.2 Permission policies 24

7.3 DataUnit1 24

7.4 DataUnit2 24

8 Document Revision History: 24

# Introduction

## Purpose

This document describes how the software system/component Knee Balancer application is structured in order to satisfy the requirements identified in the SRS. It describes the software structure, sub-components, interfaces, and data necessary for understanding the functioning of the software system.

The intended audience for this document is development, regulatory affairs and quality assurance.

## Scope

The Knee Balancer application is a clinical decision support software tool which provides on demand, an automated intra operative plan to the Orthopedic surgeon based on pre-operative data, captured soft tissue information and surgeon preferences.

This document focuses on the major design essentials rather than on implementation details. The latter are described in the source code documentation and. the UML documentation of < System Name >.

Refer to the SAD to get an overview about the overall system design.

## Definitions, acronyms and abbreviations

|  |  |
| --- | --- |
| Term | Definition |
| UML | Unified Modeling Language |
| MAKO | Robotic-Arm Assisted Surgery |
| MA | Mechanical Alignment |
| FA | Functional Alignment |
| IA | Individualized Alignment |
| HKA | Hip Knee Ankle Alignment |
| STR | Soft Tissue Release |

## References

|  |  |  |
| --- | --- | --- |
| **ID** | **Title** | **Doc. No.** |
| SDP | SGTC-QFM-DLC-001-01\_SOFTWARE\_DEVELOPMENT\_PLAN\_rev\_05-draft |  |
| URS | KB SGTC-QFM-DLC-001-22\_USER\_REQUIREMENTS\_SPECIFICATION\_rev\_04 | SGTC-QFM-DLC-001-22 |
| SRS | SGTC-QFM-DLC-001-02\_SOFTWARE\_REQUIREMENTS\_SPECIFICATION\_rev\_07 | SGTC-QFM-DLC-001-02 |
| SAD | SGTC-QFM-DLC-001-04\_SOFTWARE\_ARCHITECTURE\_DESIGN\_rev\_07.docx |  |

# System / Component Context

System diagram is as shown below



This chapter defines the borders between this system and its neighboring systems. It identifies all external interfaces between those.



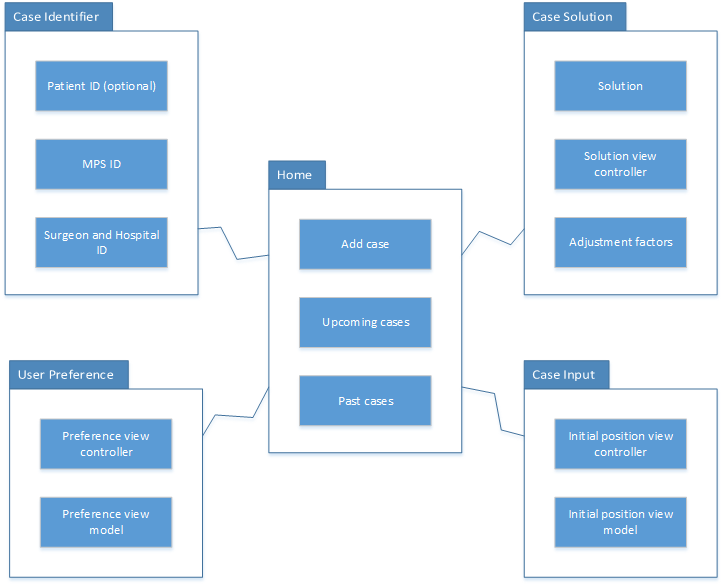
|  |  |
| --- | --- |
| Neighboring System/Component | Description |
| Neighboring System/ Mako system | Provides the plan and the input data to the MPS user Provides the plan and the input data to the MPS user Provides the plan and the input data to the MPS user.  MPS user then keys in or enters the input data into the Knee Balancer system to generate solutions. |
|  |  |

# System / Component Decomposition

System comprises of two main components. Frontend module for the UI and the backend module for the data dump for further analysis

## Module Decomposition

### Frontend Module



The Knee Balancer application consists of 5 main modules

|  |  |  |  |
| --- | --- | --- | --- |
| *SDD ID<Package, Class …>* | Module  *<Purpose, Use, Responsible for…>* | Description | SAD ID |
| SDD100 | Home | Home screen includes upcoming cases to show added case details of patients. | SAD100 |
| SDD101 | Case input | Case input screen allows the user to input femoral pre-operative plan data including medial distal femoral, lateral distal femoral, medial posterior femoral, lateral femoral posterior resection depths, femoral component varus/valgus and internal/ external rotation, and tibial pre-operative plan data including medial proximal and lateral proximal tibial resections and tibial component varus/valgus.  Allows to add intra operative data like medial extension gap, lateral extension gap, medial flexion gap and lateral flexion gap. Or User can input through camera capture All values except mHKA parameters which needs to be entered manually. | SAD101 |
| SDD102 | Case solution | The automated plan will be generated considering the following inputs: pre-operative data, Surgeon Preferences (Ranges and Targets), Intra-operative data (HKA, FFD, initial gaps), within the minimum and maximum gap ranges set in the surgeon preference card.  Adjust solution parameters itself manually either by key entries or camera capture | SAD102 |
| SDD103 | Surgeon/ User preference | Allows user to input the following planning ranges for the femoral component to be used during generation of automated implant plan:  coronal alignment (Varus/Valgus), transverse alignment (I/E Rotation), medial lateral distal and posterior resections.  Allows the user to input planning targets and ranges for the tibial component to be used during generation of automated implant plan:  Coronal alignment (Varus/Valgus)  medial lateral tibial resections.  Allows user to input Planning targets and ranges for the minimum and maximum final gaps (range) and ideal final gaps (target). | SAD103 |
| SDD104 | Case identifier | Application allows to add the following patient details Patient name, patient Id, date of birth, sex Surgeon name, Hospital name, Surgery date, Surgery time and Notes. | SAD104 |

### Backend Module

Diagram

Description automatically generated

The Knee Balancer Application Back end consists of 1 main module

|  |  |  |  |
| --- | --- | --- | --- |
| *SDD ID<Package, Class …>* | Module  *<Purpose, Use, Responsible for…>* | Description | SAD ID |
| SDD105 | Knee Balancer Backend-Cloud | Sync the data from the iOS app over internet whenever requested and store for further analysis and retrieval | SAD106 |

## Concurrent Process Decomposition

Not applicable

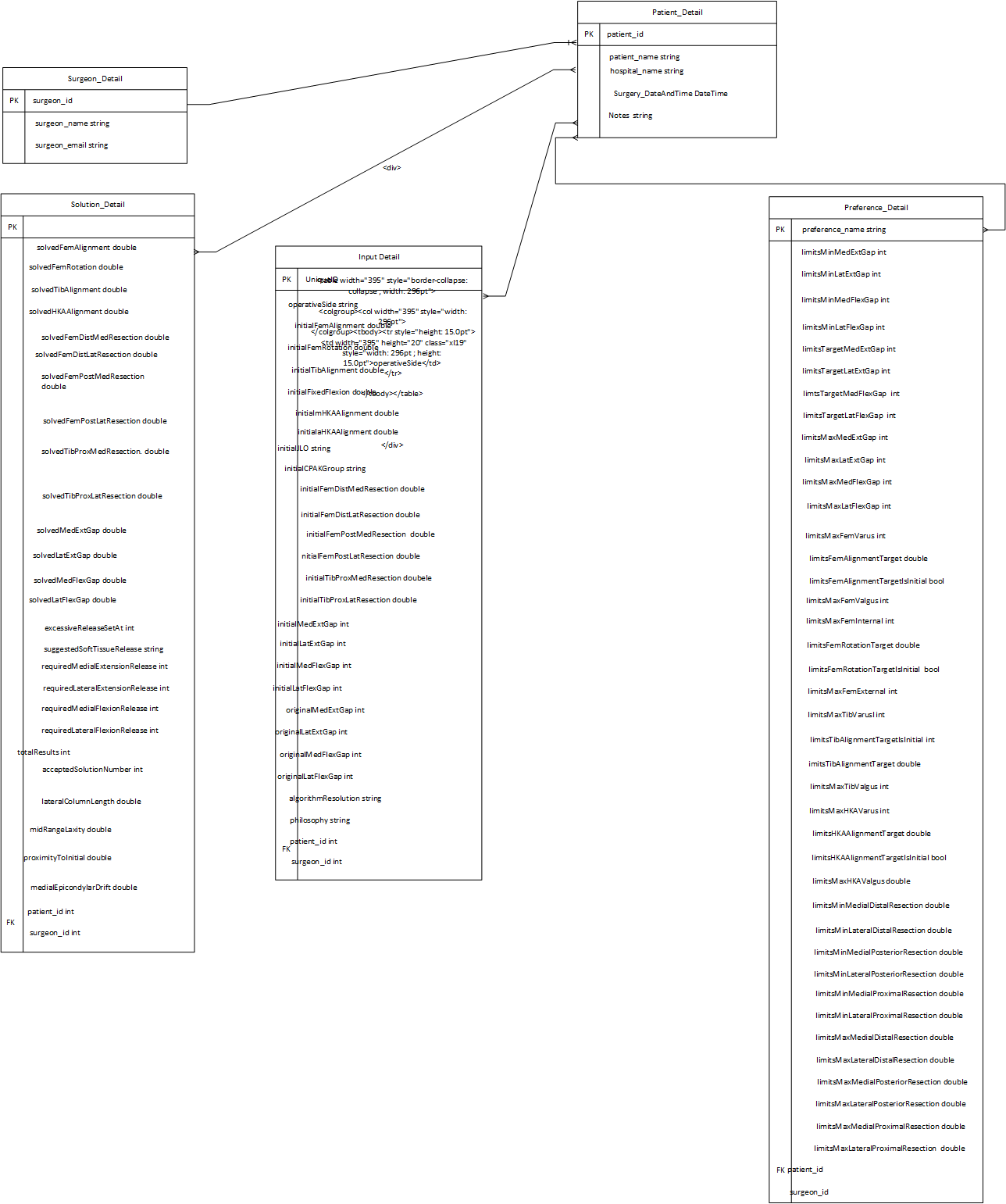
|  |  |
| --- | --- |
| Process <Program, Process, Thread, …> | Description  <Purpose, Use, Responsible for…> |
| Process 1 |  |
| Process 2 |  |

## Data Decomposition

< For data driven software systems / components (e.g. databases, tables, files, file formats, shared memory,...) this section describes how the system is decomposed into data units. In context of concurrent processing data decomposition refers to sharing data between parallel processors.

Especially for databases an appropriate representation of the data decomposition could be an entity relationship diagram. But also UML class or object diagrams might be suitable.>





|  |  |
| --- | --- |
| Data Unit | Description  <Purpose, Use, Responsible for…> |
| user\_Detail | Stores user information’s |
| Preference\_Detail | Stores user preference details |
| Input\_Detail | Stores input details |
| Solution\_Detail | Stores solution details |
| Identifier | Stores Identifier |

Case details can be stores with the combination of identifier data, user data, Prefernce details, CaseInput details, Case Solution details.

**Case Details: **

**Identifier:**

|  |
| --- |
| **patientID** |
| **patientFirstName** |
| **patientLastName** |
| **sex** |
| **legAlinment** |
| **dateOFBirth** |
| **surgeryDate** |
| **notes** |

**User Data**:

|  |
| --- |
| mpsId |
| mpsName |
| emailId |
| surgeonName |
| hospitalName |

**Preference Details:**

|  |  |
| --- | --- |
| preferenceName | limitsFemAlignmentTarget |
| dynamicMedialResectionLimits | limitsFemAlignmentTargetIsInitial |
| dynamicLateralResectionLimits | limitsMaxFemValgus |
| limitsMinMedExtGap | limitsMaxFemInternal |
| limitsMinLatExtGap | limitsFemRotationTarget |
| limitsMinLatExtGap | limitsFemRotationTargetIsInitial |
| limitsMinMedFlexGap | limitsMaxFemExternal |
| limitsMinLatFlexGap | limitsMaxTibVarus |
| limitsTargetMedExtGap | limitsTibAlignmentTargetIsInitial |
| limitsTargetLatExtGap | limitsTibAlignmentTarget |
| limtsTargetMedFlexGap | limitsMaxTibValgus |
| limitsTargetLatFlexGap | limitsMaxHKAVarus |
| limitsMaxMedExtGap | limitsMaxHKAVarus |
| limitsMaxLatExtGap | limitsHKAAlignmentTarget |
| limitsMaxMedFlexGap | limitsHKAAlignmentTargetIsInitial |
| limitsMaxLatFlexGap | limitsMaxHKAValgus |
| limitsMaxFemVarus | limitsFFDCompInitial |

**CaseInput Details:**

|  |  |
| --- | --- |
| operativeSide | initialTibProxMedResection |
| initialFemAlignment | initialTibProxLatResection |
| initialFemRotation | initialMedExtGap |
| initialTibAlignment | initialLatExtGap |
| initialFixedFlexion | initialMedFlexGap |
| initialmHKAAlignment | initialLatFlexGap |
| initialaHKAAlignment | originalMedExtGap |
| initialJLO | originalLatExtGap |
| initialCPAKGroup | originalMedFlexGap |
| initialFemDistMedResection | originalMedFlexGap |
| initialFemDistLatResection | originalLatFlexGap |
| initialFemPostMedResection | originalLatFlexGap |
| initialFemPostLatResection | algorithmResolution |
|  | alignment |

**CaseSolution Details:**

|  |  |
| --- | --- |
| solutionConfirmDate&Time | solvedLatFlexGap |
| solutionNumberAccepted | excessiveReleaseSetAt |
| solvedFemAlignment | suggestedSoftTissueRelease |
| solvedFemRotation | requiredMedialExtensionRelease |
| solvedTibAlignment | requiredLateralExtensionRelease |
| solvedHKAAlignment | requiredMedialFlexionRelease |
| solvedFemDistMedResection | requiredLateralFlexionRelease |
| solvedFemDistLatResection | totalResults |
| solvedFemPostMedResection | resultsWithOutSTR |
| solvedFemPostLatResection | resultsWithSTR |
| solvedTibProxMedResection | resultsOverLimits |
| solvedTibProxLatResection | calculationVersion |
| solvedMedExtGap | lateralColumnLength |
| solvedLatExtGap | midRangeLaxity |
| solvedMedFlexGap | proximityToInitial |
|  | medialEpicondylarDrift |

**Logs**:

1. Error
2. Exception
3. Information

Error handling, we achieving by default swift features those are listed below

Do catch block, throw, try, guard let, if let

do – This keyword starts the block of code that contains the method that can potentially throw an error.

try – You must use this keyword in front of the method that throws. Think of it like this: “You’re *trying* to execute the method.

catch – If the throwing method fails and raises an error, the execution will fall into this catch block. This is where you’ll write code display a graceful error message to the user.

Throw - Use this pattern to handle any potential errors caused by a method that throws

LogStructure:

A picture containing table

Description automatically generated

# Interfaces

<Specify the details for interfaces between external and internal units. If the interface description becomes too extensive for the scope of this document, consider referring to a separate interface specification document.>

## Interface with external hardware components

**NA**

## Interface with external software components

**NA**

## Interface between modules

# Detailed Design

< The detailed design describes the internal details of each unit. There are many representations to describe the details of design units including

## various UML diagram types (class diagrams, collaboration diagrams, state transition diagrams),

### Home Module

|  |  |  |  |
| --- | --- | --- | --- |
| *SDD ID<Package, Class …>* | Module  *<Purpose, Use, Responsible for…>* | Description | SAD ID |
| SDD100 | Home | Home screen includes upcoming cases to show added case details of patients. | SAD100 |

A picture containing text, sky

Description automatically generated

Diagram

Description automatically generated

### Case input module

|  |  |  |  |
| --- | --- | --- | --- |
| *SDD ID<Package, Class …>* | Module  *<Purpose, Use, Responsible for…>* | Description | SAD ID |
| SDD101 | Case input | Case input screen allows the user to input femoral pre-operative plan data including medial distal femoral, lateral distal femoral, medial posterior femoral, lateral femoral posterior resection depths, femoral component varus/valgus and internal/ external rotation, and tibial pre-operative plan data including medial proximal and lateral proximal tibial resections and tibial component varus/valgus.  Allows to add intra operative data like medial extension gap, lateral extension gap, medial flexion gap and lateral flexion gap Or User can input through camera capture All values except mHKA parameters | SAD101 |

Diagram

Description automatically generated

Text

Description automatically generated

InitialPosition Popup :

Diagram

Description automatically generated

### Case Solution Module

|  |  |  |  |
| --- | --- | --- | --- |
| *SDD ID<Package, Class …>* | Module  *<Purpose, Use, Responsible for…>* | Description | SAD ID |
| SDD102 | Case solution | The automated plan will be generated considering the following inputs: pre-operative data, Surgeon Preferences (Ranges and Targets), Intra-operative data (HKA, FFD, initial gaps), within the minimum and maximum gap ranges set in the surgeon preference card.  Adjust solution parameters itself manually either by key entries or camera capture | SAD102 |

Text

Description automatically generatedA picture containing table

Description automatically generated

Text

Description automatically generated with medium confidence

### User Preference module

|  |  |  |  |
| --- | --- | --- | --- |
| *SDD ID<Package, Class …>* | Module  *<Purpose, Use, Responsible for…>* | Description | SAD ID |
| SDD103 | Surgeon/ User preference | Allows user to input the following planning ranges for the femoral component to be used during generation of automated implant plan:  coronal alignment (Varus/Valgus), transverse alignment (I/E Rotation), medial lateral distal and posterior resections.  Allows the user to input planning targets and ranges for the tibial component to be used during generation of automated implant plan:  Coronal alignment (Varus/Valgus)  medial lateral tibial resections.  Allows user to input Planning targets and ranges for the minimum and maximum final gaps (range) and ideal final gaps (target). | SAD103 |

Chart, diagram

Description automatically generated with medium confidence

Text

Description automatically generated

### Case Identifier module

|  |  |  |  |
| --- | --- | --- | --- |
| *SDD ID<Package, Class …>* | Module  *<Purpose, Use, Responsible for…>* | Description | SAD ID |
| SDD104 | Case identifier | Application allows to add the following patient details Patient name, patient Id, date of birth, sex Surgeon name, Hospital name, Surgery date, Surgery time and Notes. | SAD104 |

A picture containing chart

Description automatically generated

Text

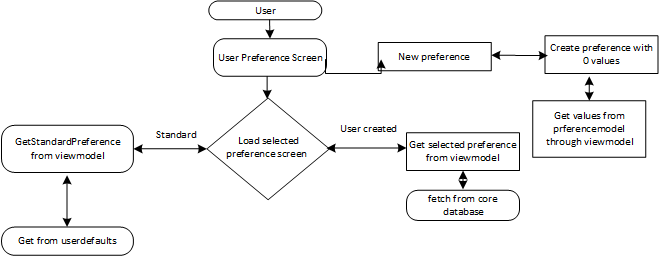
Description automatically generated with low confidence

## pseudo code to outline algorithms,

## flowcharts to visualize control flow or data flow,

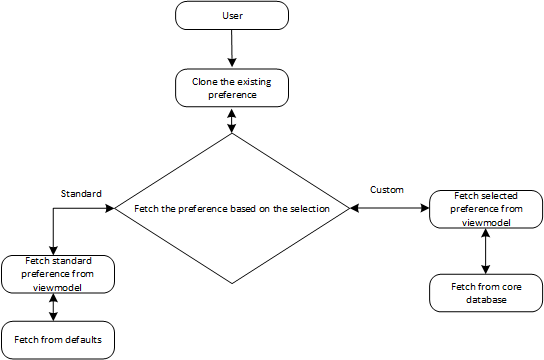
### New Preference:

New preference allows surgeon to create own preference value for Distal Femur, Posterior Femur, Proximal Tibia, Hip Knee Ankle(HKA) with default value as 0.



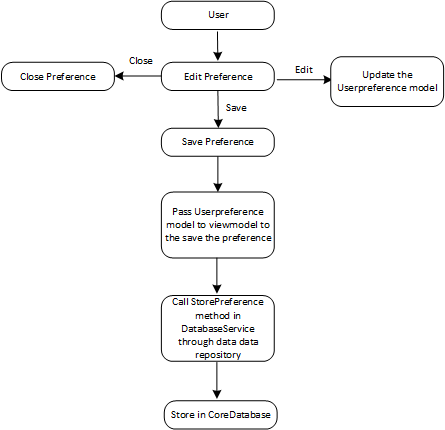
### Clone Preference:

Clone preference allows Surgeon to clone any existing preference.



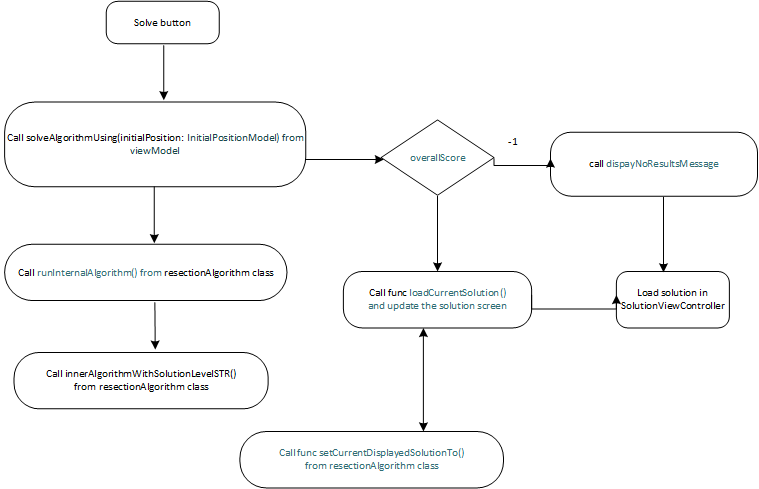
### View Preference:

Edit preference allows surgeon to edit the existing preference other than FDA-Standard and store in DB. Edit preference will be enabled in the preference list only for newly created or cloned preference.



### Solution :

Solution button will be enabled after entering all the input values initialFemAlignment, initialFemRotation, initialTibAlignment, initialFixedFlexion, initialmHKAAlignment, initialaHKAAlignment, initialFemDistMedResection, initialFemDistLatResection, initialFemPostMedResection, initialFemPostLatResection,initialTibProxMedResection, initialTibProxLatResection, initialMedEXTGap,initialLatEXTGap,initialMedFlexGap,initialLatFlexGap, originalMedEXTGap,originalLatEXTGap, originalMedFlexGap, originalLatFlexGap and the selected preference. The input and preference values are passed to solveAlgorithm() method to get solutions.



### Camera capture:

Flowcharts to visualize control flow or data flow of Camera Capture. If user tap on Camera button device camera will open and it will allow the user to take the picture. Once user done with image capture it will send image to Vision Kit.

Diagram

Description automatically generated

## SOUP items

|  |  |  |
| --- | --- | --- |
| **Items** | **License** | **Version** |
| Mix panel | Open source | 3.1.5 |
| Vision Kit(iOS) | Available from iOS 13 | iOS 14.5 |
| Microsoft Authentication | Open source | 1.1.3 |

## Module 1 – Initial Position

| Module  <Package, Class …> | Description  <Purpose, Use, Responsible for…> |
| --- | --- |
| InitialPositionViewController | The class contains the initial position from all of the user controls on the Initial Position landscape view. The initial position object that gets passed into the ResectionAlgorithm object |
| InitialPositionViewModel | This class is used to calculate all the image transform and image rotation required by the InitialPositionViewController. |
| InitialPositionValidator | This class is used to validate the initialPositionModel. |

## Module2 – User preference

| Module  <Package, Class …> | Description  <Purpose, Use, Responsible for…> |
| --- | --- |
| PreferenceViewController | This class is used to display the standard preference, create new, clone, edit, save and delete the preference. Also display UI for preference screen. |
| PreferenceViewModel | This class is used for fetching and save the created, edited preferences. |

## Module3 - Solution

| Module  <Package, Class …> | Description  <Purpose, Use, Responsible for…> |
| --- | --- |
| SolutionViewController | This class is used to display solutions from the ResectionAlgorithm based on the input provided on the initial position and selected preference. |
| Solution | An object that represents a single possible solution from the ResectionAlgorithm. Also contains state information on its scoring elements and its soft tissue release requirements. |

### 5.8 Upload Module

|  |  |  |  |
| --- | --- | --- | --- |
| *SDD ID<Package, Class …>* | Module  *<Purpose, Use, Responsible for…>* | Description | SAD ID |
| SDD105 | Knee Balancer Backend-Cloud | Sync the data from the iOS app over internet whenever requested and store for further analysis and retrieval | SAD106 |

### 5.8.1 Backend -Cloud

**Azure Blob Storage**

Azure Blob storage is Microsoft's object storage solution for the cloud. Blob storage is optimized for storing massive amounts of unstructured data. Unstructured data is data that doesn't adhere to a particular data model.

**Blob storage details**

Storage account - Stryker

Container - KneeBalancer

Blob - \* JSON (case data, logs data)

**Case Details :**

1. Identifier
2. User Data (Mps)
3. Preference
4. CaseInput values
5. CaseSolution values

**Logs Structure:**

1. 2022-04-22 11:39:09.192 ERROR 12852 Description
2. 2022-04-22 11:40:19.153  WARN 18564 Description
3. 2022-04-22 11:41:18.238  INFO 23908 Description

Graphical user interface

Description automatically generated with medium confidence

|  |  |
| --- | --- |
| Units | Description |
| API’s | Azure blob storage providing direct API’s to the save the data on cloud. |
| Azure Storage Account | Azure storage account contains all of your Azure Storage data objects i.e files.The storage account provides a unique namespace for your Azure Storage data that is accessible from anywhere in the world over HTTP or HTTPS. |
| Azure File Service | To store the data as josn file in Azure cloud-azure blob storage for future retrieval |

**Minimum Requirements**

**From Front end**

1. Authenticate the app using the Stryker credentials
2. Configure automatic upload whenever come online
3. Show reminder to user if the data is not uploaded for more than 1 day. Configurable upto 10 days.
4. Small dashboard for upload details in the app to show
   1. Total past case details uploaded
   2. Total past case details pending for upload
5. Data push to blob storage – case data, logs data etc

**From Backend**

1. Azure services and Azure Blob
2. API calls
3. Data dump for analysis

**Sequence diagram – Data push to blob storage:**

A picture containing chart

Description automatically generated

### 5.9 Authentication Module

The Microsoft Authentication Library (MSAL) **enables developers to acquire tokens from the Microsoft identity platform in order to authenticate users and access secured web APIs**. It can be used to provide secure access to Microsoft Graph, other Microsoft APIs, third-party web APIs, or your own web API.

|  |  |  |  |
| --- | --- | --- | --- |
| *SDD ID<Package, Class …>* | Module  *<Purpose, Use, Responsible for…>* | Description | SAD ID |
| SDD106 | Knee Balancer Authentication | Enables developers to acquire tokens from the Microsoft identity platform in order to authenticate users and access secured web APIs. | SAD107 |

**Diagram

Description automatically generated**

# Detailed Process Design

< Internal details about the concurrent behavior of the various threads and processes. The design description might include

* 1. Process activity diagrams
  2. Interaction of processes
  3. Communication protocols

## Process1

## Process2

# Detailed Data Design

## Internal details of the data design. The design description might include

### ER diagrams

### pseudo code to outline SQL queries (especially for medium-high risk products/modules),

## Permission policies

## DataUnit1

## DataUnit2

# Document Revision History:

|  |  |  |  |
| --- | --- | --- | --- |
| Revision  Level | Revision Date | Effective Date | Reason and Description of Revision |
|  |  |  |  |
|  |  |  |  |

**9. Appendix**

| Term/Acronym | Definition |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |