Digital Assessment Project

SITS Marks Import API v2

Specification

Produced for University College London

Author: David Wraith

Date: 14 January 2022

Version: 0.9

**Table of Contents**

[Introduction](#_heading=h.3dy6vkm) **3**

[Core Data](#_heading=h.1t3h5sf) **4**

[API MARKSIMPORT2](#_heading=h.4d34og8) **6**

[API Parameters](#_heading=h.2s8eyo1) 6

[Data Updates](#_heading=h.17dp8vu) 6

[Mark Scheme Validation](#_heading=h.3rdcrjn) 7

[Error Handling](#_heading=h.26in1rg) 7

[Response](#_heading=h.j881c2ywujge) 8

[Example API request with JSON](#_heading=h.lnxbz9) 8

[API MARKSLOGIMPORT](#_heading=h.35nkun2) **10**

[Data Updates](#_heading=h.44sinio) 11

[Submission Status Calculation](#_heading=h.2jxsxqh) 12

[Error Handling](#_heading=h.z337ya) 13

[Response Body](#_heading=h.1q5v04qcpu3p) 13

[Example API request with JSON](#_heading=h.3j2qqm3) 13

[SITS Configuration](#_heading=h.1y810tw) **15**

# 

# 

# Introduction

UCL have embarked on a project to update the SITS Marks Import API originally written by Stu3. This will firstly involve cloning the original API and changing how the format of the data is received, but all other logic will remain the same. Secondly a new API will be created to handle the transfer of Submission Deadline Date/Time and Hand-in Date/Time as well as calculating a submission status.

SITS will continue to be the source of truth for student, academic model and assessment data and will be loaded into WiseFlow – this is not in scope.

# Core Data

Student assessment data is held in SITS in the CAM\_SAS (Assessment) and CAM\_SRA (Re-assessment) tables and are the core tables the API will update. For the API to function, it is assumed these records will have already been created along with any underlying academic model data (i.e. MOD, MAV, MAP, MAB).

Each record in the CAM\_SAS and CAM\_SRA table is a student assessment / reassessment record.

The primary key of the CAM\_SAS table is:

* SPR\_CODE
* MOD\_CODE
* MAV\_OCCUR
* AYR\_CODE
* PSL\_CODE
* MAP\_CODE
* MAB\_SEQ

The primary key of the CAM\_SRA table is:

* SPR\_CODE
* MOD\_CODE
* MAV\_OCCUR
* AYR\_CODE
* PSL\_CODE
* SRA\_RSEQ

A Student Assessment Log (SAL) record and Student Re-Assessment Log (SRR) record should also have already been created relating back to the parent CAM\_SAS or CAM\_SRA record respectively. This will be used to store Submission Deadline Date/Time, Hand-in Date/Time and Submission Status.

The primary key of the CAM\_SAL table is:

* SPR\_CODE
* MOD\_CODE
* MAV\_OCCUR
* AYR\_CODE
* PSL\_CODE
* MAP\_CODE
* MAB\_SEQ

The primary key of the CAM\_SRR table is:

* SPR\_CODE
* MOD\_CODE
* MAV\_OCCUR
* AYR\_CODE
* PSL\_CODE
* SRA\_RSEQ
* SRA\_CURA

# API MARKSIMPORT2

This API will be a clone of the existing MARKSIMPORT API. Core logic in the Workflow (WFH) will remain the same, with only the way data is received changing.

## API Parameters

The API will take the following parameters to identify the CAM\_SAS or CAM\_SRA and related records to update via the URL. The PUT verb should be used.

| URL parameter | Description | Example |
| --- | --- | --- |
| assessment-component | MAP\_CODE-MAB\_SEQN | BASC0023A6UD-001 |
| student | SPR\_CODE-AYR-CODE-PSL-CODE-SRA\_RSEQ | 12345678\_2-2021-T1-001 |

\*The SPR\_CODE will have the slash (/) replaced with an underscore (\_)

\*The SRA\_RSEQ will be 0 for updates to CAM\_SAS, otherwise assume update to CAM\_SRA

An example of the URL:

<https://evision-dev.ucl.ac.uk/urd/sits.urd/run/SIW_RWS/MARKSIMPORT2/assessment-component/BASC0023A6UD-001/student/12345678_2-2021-T1-001>

The following parameters would also be needed in the body of the request in JSON format, to provide the data from WiseFlow to update SITS:

| API Field Code | Description |
| --- | --- |
| actual\_mark\_or\_grade | SITS will need to look at the mark scheme to determine whether to update the Actual Mark or Actual Grade fields |
| actual\_mark | Actual Mark (if actual\_mark\_or\_grade is blank) |
| actual\_grade | Actual Grade (if actual\_mark\_or\_grade is blank) |

## Data Updates

The update to CAM\_SAS or CAM\_SRA should be simple once a candidate key is passed into the API.

The fields on CAM\_SAS (SRA\_RSEQ = 0) that will need to be updated are:

* SAS\_ACTM = actual\_mark
* SAS\_ACTG = actual\_grade
* SAS\_PRCS = I

The fields on CAM\_SRA (SRA\_RSEQ > 0) that will need to be updated are:

* SRA\_ACTM = actual\_mark
* SRA\_ACTG = actual\_grade
* SRA\_PRCS = I

In the case the actual\_mark\_or\_grade parameter is instead populated (and the actual\_mark / actual\_grade parameters are blank) then the mark and grade will need to be derived based on the information in the [next section](#_heading=h.3rdcrjn).

## Mark Scheme Validation

UCL has three different types of Mark schemes with regards to entering a Mark and Grade. The CAM\_MKS record hold this information in the MKS\_MARKS field; valid values are:

* N = Only a Grade needs to be populated on the CAM\_SAS or CAM\_SRA record
* P = This should work in the same way as the value N
* Y = A mark needs to be populated on the CAM\_SAS or CAM\_SRA record and the Grade (if passed into the API as blank) can be derived from the highest ranked MKC (MKC\_RANK) matching:
* Process (MKC\_PROC) matches RECORDTYPE (i.e. SAS or RAS)
* MARK is within Mark Range (MKC\_MIND and MKC\_MAXD)
* Attempt (MKC\_ATTP) matches SAS\_CURA / SRA\_CURA

The mark scheme also denotes how many decimal places a MARK should be rounded to (field MKS\_NDDM) before it is saved on the CAM\_SAS or CAM\_SRA record

For example, if MKS\_NDDM is set to 2. The entered mark is 60.55. Then this is stored in the database fields as 6055. If MKS\_NDDM is set to 0. The entered mark is 60.00. Then this is stored in the database fields as 60.

To find the mark scheme to use (as it can be both on CAM\_MAP and CAM\_MAB):

* For Assessment (SAS) then first look at MKS\_CODE (from MAB), if blank use MAP\_MKS2 (from MAP).
* For Reassessment (RAS) then first look at the MAP\_MKS3 (from MAP), if blank use MKS\_CODE (from MAB)

If no Mark scheme can be found, then return an error.

## Error Handling

If the update is successful, the API should report a HTTP 200 status.

In the event of an error the API should return a HTTP 400 status, along with an error message.

The following validation / error messages should be considered:

* No CAM\_SAS record exists for candidate key provided
* No CAM\_SRA record exists for candidate key provided
* CAM\_SAS record not in valid state for mark input – no updates completed
* SAS\_PRCS is not blank or I and SAS\_PROC != SAS
* CAM\_SRA record not in valid state for mark input – no updates completed
* SRA\_PRCS is not blank or I and SRA\_PROC != RAS
* If CAM\_SAS already has an actual Mark/Grade populated that doesn’t match the API parameters - update should fail
* If CAM\_SRA already has an actual Mark/Grade populated that doesn’t match the API parameters - update should fail
* Mark (actuall\_mark parameter) is invalid for this assessment – no updates completed
* Grade (actual\_grade parameter) is invalid for this assessment – no updates completed
* Mark and Grade combination (actual\_mark and actual\_grade parameters) is invalid for this assessment – no updates completed
* Mark or Grade (actual\_mark\_or\_grade parameter) is invalid for this assessment – no updates completed

## Response

The format of the response should be:

{

"identifier": "f5840831-8974-4dd6-9325-96cdc52d0ae4",

"request": "/MARKSIMPORT2/assessment-component/BASC0023A6UD-001/student/12345678\_2-2021-T1-0",

"code": "20100",

"message": "No SAS records exists for primary key",

"timestamp": "2021-05-04T01:24:54"

}

Where:

* identifier - unique GUID for the transaction (lowercase)
* request - return the RESTful based query string
* code - any error code (use 0 for success)
* message - any error message (use “Success” for success)
* timestamp - Current date/time

## Example API request with JSON

<https://evision-dev.ucl.ac.uk/urd/sits.urd/run/SIW_RWS/MARKSIMPORT2/assessment-component/BASC0023A6UD-001/student/12345678_2-2021-T1-0>

{

"actual\_mark\_or\_grade":"80"

}

<https://evision-dev.ucl.ac.uk/urd/sits.urd/run/SIW_RWS/MARKSIMPORT2/assessment-component/BASC0023A6UD-001/student/12345678_2-2021-T1-001>

{

"actual\_mark":"0",

"actual\_grade":"IR"

}

<https://evision-dev.ucl.ac.uk/urd/sits.urd/run/SIW_RWS/MARKSIMPORT2/assessment-component/BASC0023A6UD-001/student/12345678_2-2021-T1-001>

{

"actual\_mark":"60",

"actual\_grade":""

}

# API MARKSLOGIMPORT

This new API will update the CAM\_SAL and CAM\_SRR tables with submission due date and hand-in dates as well as a submission status.

## API Parameters

The API will take the following parameters to identify the CAM\_SAL or CAM\_SRR and related records to update via the URL. The PUT verb should be used.

| URL parameter | Description | Example |
| --- | --- | --- |
| assessment-component | MAP\_CODE-MAB\_SEQN | BASC0023A6UD-001 |
| student | SPR\_CODE-AYR-CODE-PSL-CODE-SRA\_RSEQ | 12345678\_2-2021-T1-001 |

\*The SPR\_CODE will have the slash (/) replaced with an underscore (\_)

\*The SRA\_RSEQ will be 0 for updates to CAM\_SAL otherwise assume update to CAM\_SRR

An example of the URL:

<https://evision-dev.ucl.ac.uk/urd/sits.urd/run/SIW_RWS/MARKSLOGIMPORT/assessment-component/BASC0023A6UD-001/student/12345678_2-2021-T1-001>

The following parameters would also be needed in the body of the request in JSON format, to provide the data from WiseFlow to update SITS:

| API Field Code | Description |
| --- | --- |
| original\_due\_datetime | Original Due Date / Time |
| current\_due\_datetime | Current Due Date / Time |
| handin\_datetime | Received Date / Time |
| handin\_status | Hand-in Status Override (validated by CAM\_SUS table) |
| handed\_in | Has submission been made (true or false) |
| handed\_in\_blank | Was submission blank (true or false) |
| permitted\_submission\_period | Permitted Submission Extension Period  This will be in the ISO8601 ‘newer’, not abridged format (PxxDxxHxxM).  A few examples:  P15D7H32M – which means Period of 15 days, 7 hours, 32 minutes.  P15D – which means Period of 15 days.  P15D32M – which means Period of 15 days, 0 hours, 32 minutes.  P7H32M – which means Period of 7 hours, 32 minutes.  P7H – which means Period of 7 hours  P0D or NULL – which means 0 days, 0 hours, 0 minutes  The following code has already been created in UCLs system to break down a period.  <<#PRD=P15D7H32M>>  <<@MARKSLOGIMPORT\_SPLIT\_PRD&L0>>  PRD = <<#PRD>>  PRD DAYS = <<#PRD\_DAYS>>  PRD HOURS = <<#PRD\_HOURS>>  PRD MINS = <<#PRD\_MINS>> |
| export\_staff | Wiseflow Export Staff Member (UPI / PRS\_CODE – but leave unvalidated) |
| export\_timestamp | Wiseflow Export Timestamp |
| export\_flow\_id | Wiseflow Export Flow ID (Treat as a string) |
| no\_of\_items | Number of Items in Submission - This will always be BLANK, 0 or >0 so can be validated as an integer |

\*All Date / Time and timestamp values will be in the format YYYY-MM-DDTHH:NN:SS

## Data Updates

The update to CAM\_SAL or CAM\_SRR should be simple once a candidate key is passed into the API.

The CAM\_SAL and CAM\_SRR record should only be updated if the SAL\_SUSC field is currently blank (indicating no manual intervention is taking place).

The update to CAM\_SAL would use the same PK as for CAM\_SAS and the fields to update are:

* SAL\_UDFK = export\_timestamp
* SAL\_RECD = Date part of handin\_datetime
* SAL\_RECT = Time part of handin\_datetime (HH:NN)
* SAL\_STOD = Current date
* SAL\_STOT = Current time
* SAL\_PRSC = export\_staff
* SAL\_SUSC = handin\_status unless blank then calculate as per [next section](#_heading=h.2jxsxqh)
* ~~SAL\_DUEO = Date part of original\_due\_datetime~~
* ~~SAL\_DUET = Time part of original\_due\_datetime (HH:NN)~~
* ~~SAL\_UDFG = Time part of original\_due\_datetime (HH:NN:SS)~~
* ~~SAL\_DUEC = Date part of current\_due\_datetime (only populate if date/time differs from original\_due\_datetime)~~
* ~~SAL\_DUTC = Time part of current\_due\_datetime (only populate if date/time differs from original\_due\_datetime) (HH:NN)~~
* ~~SAL\_UDFH = Time part of current\_due\_datetime (only populate if date/time differs from original\_due\_datetime) (HH:NN:SS)~~
* ~~SAL\_UDFI = Time part of handin\_datetime (HH:NN:SS)~~
* ~~SAL\_UDFA = permitted\_submission\_period~~
* ~~SAL\_TREF = export\_flow\_id~~
* ~~SAL\_NUMI = no\_of\_items~~

The update to CAM\_SRR would use the same PK as for CAM\_SRA (plus additional field lookup SRA\_CURA from CAM\_SRA) and the fields to update are:

* SRR\_RECD = Date part of handin\_datetime
* SRR\_RECT = Time part of handin\_datetime (HH:NN)
* SRR\_UDFK = export\_timestamp
* SRR\_STOD = Current date
* SRR\_STOT = Current time
* SRR\_PRSC = export\_staff
* SRR\_SUSC = handin\_status unless blank then calculate as per [next section](#_heading=h.2jxsxqh)
* ~~SRR\_DUEO = Date part of original\_due\_datetime~~
* ~~SRR\_DUET = Time part of original\_due\_datetime (HH:NN)~~
* ~~SRR\_UDFG = Time part of original\_due\_datetime (HH:NN:SS)~~
* ~~SRR\_DUEC = Date part of current\_due\_datetime (only populate if date/time differs from original\_due\_datetime)~~
* ~~SRR\_DUTC = Time part of current\_due\_datetime (only populate if date/time differs from original\_due\_datetime) (HH:NN)~~
* ~~SRR\_UDFH = Time part of current\_due\_datetime (HH:NN:SS)~~
* ~~SRR\_UDFI = Time part of handin\_datetime (HH:NN:SS)~~
* ~~SRR\_UDFA = permitted\_submission\_period~~
* ~~SRR\_TREF = export\_flow\_id~~
* ~~SRR\_NUMI = no\_of\_items~~

## Submission Status Calculation

The API will be able to calculate the submission status to be one of these values:

* S = Submitted on-time in AUCL
* L = Submitted late in AUCL
* I = Invalid or Blank submission in AUCL
* N = Not Submitted in AUCL
* (BLANK) = Undefined (Default)

For the purposes of the below conditions, hard\_due\_datetime is calculated as current\_due\_datetime (if blank use original\_due\_datetime)

For the purposes of the below conditions, late\_due\_datetime is calculated as current\_due\_datetime (if blank use original\_due\_datetime) plus any permitted\_submission\_period

If permitted\_submission\_period is blank or 0 then hard\_due\_datetime will equal late\_due\_datetime.

Undefined (BLANK)

late\_due\_datetime is blank OR export\_timestamp < late\_due\_datetime

In this case leave the submission status blank and do not process any of the below conditions. This status should also be used if none of the below conditions pass.

S = Submitted on-time in AUCL

handed\_in\_blank = false AND handed\_in = true AND handin\_datetime <= hard\_due\_datetime

L = Submitted late in AUCL

handed\_in\_blank = false AND handed\_in = true AND handin\_datetime > hard\_due\_datetime AND handin\_datetime <= late\_due\_datetime

I = Invalid or Blank submission in AUCL

handed\_in\_blank = true AND handed\_in = true

N = Not Submitted in AUCL

(handed\_in\_blank = false AND handed\_in = false)

OR

(handed\_in\_blank = false AND handed\_in = true AND handin\_datetime > late\_due\_datetime)

## Error Handling

If the update is successful, the API should report a HTTP 200 status.

In the event of an error the API should return a HTTP 400 status, along with an error message.

The following validation / error messages should be considered:

* No CAM\_SAL record exists for candidate key provided
* No CAM\_SRR record exists for candidate key provided
* The CAM\_SAL record already has SAL\_SUSC populated
* The CAM\_SRR record already has SRR\_SUSC populated
* The hand\_in\_status provided is invalid (not in SUS table)

## Response Body

The format of the response should be:

{

"identifier": "f5840831-8974-4dd6-9325-96cdc52d0ae4",

"request": "/MARKSLOGIMPORT/assessment-component/BASC0023A6UD-001/student/12345678\_2-2021-T1-0",

"code": "20100",

"message": "No SAL records exists for primary key",

"timestamp": "2021-05-04T01:24:54"

}

Where:

* identifier - unique GUID for the transaction (lowercase)
* request - return the RESTful based query string
* code - any error code (use 0 for success)
* message - any error message (use “Success” for success)
* timestamp - Current date/time

## Example API request with JSON

https://evision-dev.ucl.ac.uk/urd/sits.urd/run/SIW\_RWS/MARKSLOGIMPORT/assessment-component/BASC0023A6UD-001/student/12345678\_2-2021-T1-0

{

"original\_due\_datetime":"2020-08-12T23:59:59",

"current\_due\_datetime":"2020-08-14T23:59:59",

"handin\_datetime":"2020-08-16T23:59:59",

"handin\_status":"",

"handed\_in\_blank": “false”,

“handed\_in”: “true”,

"permitted\_submission\_period":"P7D",

"export\_staff":"CCEAWRA",

"export\_timestamp":"2020-08-28T23:59:59",

"export\_flow\_id":"27505",

"no\_of\_items":"3"

}

https://evision-dev.ucl.ac.uk/urd/sits.urd/run/SIW\_RWS/MARKSLOGIMPORT/assessment-component/BASC0023A6UD-001/student/12345678\_2-2021-T1-001

{

"original\_due\_datetime":"2020-08-12T23:59:59",

"current\_due\_datetime":"2020-08-12T23:59:59",

"handin\_datetime":"2020-08-10T23:59:59",

"handin\_status":"S",

"handed\_in\_blank": “false”,

“handed\_in”: “true”,

"permitted\_submission\_period":"",

"export\_staff":"CCEAWRA",

"export\_timestamp":"2020-08-27T23:59:59",

"export\_flow\_id":"27506",

"no\_of\_items":""

}

# SITS Configuration

It is suggested that a RESTful based API is created using a JSON data format – cloning the original Marks Import API setup.

This would mean the setup of a new RRS and RWS record in Stu-talk using the PUT verb for each API.

New WSF would be created and attached to the existing MARKSIMPORT role group (RGF).

The WSF should use the Stu-talk workflow service (MEN\_WFH\_SVC.RUN\_WORKFLOW).

The config of the API should then be setup using the WFH (Stu-talk Workflow) functionality.