SSID API Phase 1 (Proposal)

Introduction

SSID API allows you to programmatically use a subset of SSID's functionalities. With SSID API, you can create assignments and submit assignment files programmatically, retrieve plagiarism detection results in machine-readable format, and integrate SSID into your applications. This document describes the usage of SSID API.

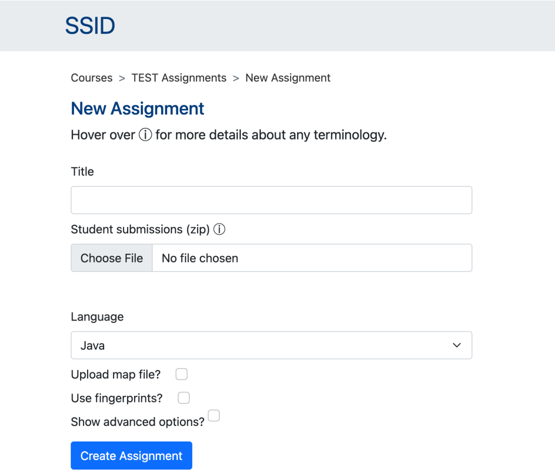
Authentication

- Creating API keys: Please contact SSID team to get your developer API key. [Note: I think there's no need to have refresh tokens as discussed in March because we are only doing persistent developer authentication but not user login. Since Coursemology is the only user of SSID API in this phase, we can manually generate an API key for Coursemology for now. As discussed in March, we can issue keys manually or implement a web interface for key generation from Settings page of SSID].
- Adding API key to your request: You must include your API key with every request in the parameter X-API-KEY in your request header. [Note: Looks like some APIs pass in a parameter api_key in the body; both options seem equally good, so I just went with this].

Create an assignment

POST /api/v1/courses/{course_id}/assignment/

Description: Creates a new assignment for SSID to process. The web equivalence is going to /courses/{course_id}/assignments/new and creating a new assignment.



SSID

Courses > TEST Assignments > New Assignment

New Assignment

Hover over ⓘ for more details about any terminology.

Title

Student submissions (zip) ⓘ

No file chosen

Language: Java

Upload map file? □

Use fingerprints? □

Show advanced options? □

The content type of the request should be "multipart/form-data". Each parameter is a form part, with the parameter name specified in the "name" attribute in "Content-Disposition" header. In the body, any boundary delimiter for different parts to be sent is acceptable.

[Note: The parameters here are the union of existing parameters in SSID and parameters discussed during SSID-Coursemology meetings. We will treat functions not yet available in SSID as feature requests for SSID, and those functions will be made available in either SSID web interface or SSID API or both after they are implemented].

Params:

- title (Required) string: The title of the assignment. [Note: Currently this is already available in SSID's web interface].
- language (Required) string: Programming language of submission files. Currently supported: "java", "python3", "c", "cpp", "javascript", "r", "ocaml", "matlab", "scala". The parameter value must be in lowercase and match exactly one of the options. [Note: Currently this is already available in SSID's web interface].
- useFingerprints (Optional) boolean: If true, enable the optimization of preprocessing batch of submissions using winnowing fingerprinting algorithm before pairwise comparisons. If not specified, defaults to false. [Note: Currently this is already available in SSID's web interface].
- minimumMatchLength (Optional) number: The least number of contiguous identical statements required to flag a match. If not specified, defaults to 2. [Note: Currently this is already available in SSID's web interface].
- sizeOfNGram (Optional) number: Specifies size of n-gram to be used in SSID. An n-gram is a contiguous subsequence of n tokens of a given sequence. If not specified, defaults to 5. [Note: Currently this is already available in SSID's web interface].
- discardAfter (Optional) number: If specified, discard files after such duration in seconds. [Note: In March discussion, Jonathan raised MOSS discards submissions for a limited 2 weeks or so, and Prof Min thinks we can adopt longer per-term expiry. Looks like currently SSID stores all data persistently and data is manually cleared every semester. This parameter accommodates the future auto-discard functionality. This function is not yet available in SSID, so we need to first build it for SSID, then allow it to be used via API].
- studentSubmissions (Required) zip: zip file of student submissions in SSID’s standard format. [Note: The zip file should be in SSID's standard format (potentially includes skeleton codes). For more details, please refer to SSID's User's Guide. Removed file size limit as Prof Min raised there might be users with bigger file sizes.].
- mappingFile (Optional) csv: csv map file that allows you to map between a directory name (in the uploaded zip file) and the student roster that you might be using for your modules. For more details, see SSID's User Guide. [Note: Currently this is already available in SSID's web interface].

- references zip: zip file for past semesters to be used as reference. Maximum of 5 reference zips allowed. [Note: For now let's limit to a maximum of 5 reference zips. Removed file size limit as Prof Min raised there might be users with bigger file sizes. This function is not yet available in SSID, so we need to first build it for SSID, then allow it to be used via API. To be built in Phase 2].

Request example:

Header

```
{
  "Content-Type": "multipart/form-data;
\"boundary=------------------------a41c8131a72f1dad\"",
  "X-Api-Key": "YOUR_API_KEY"
}
```

Body

```
--------------------------a41c8131a72f1dad
Content-Disposition: form-data; name="title"

assignment1
--------------------------a41c8131a72f1dad
Content-Disposition: form-data; name="language"

java
--------------------------a41c8131a72f1dad
Content-Disposition: form-data; name="sizeOfNGram"

5
--------------------------a41c8131a72f1dad
Content-Disposition: form-data; name="studentSubmissions"; filename="dir.zip"
Content-Type: application/octet-stream

<file content>
--------------------------a41c8131a72f1dad–
```

Possible responses:

Code	Status	Return body
200	Successful	{ "assignmentID": 1168 }

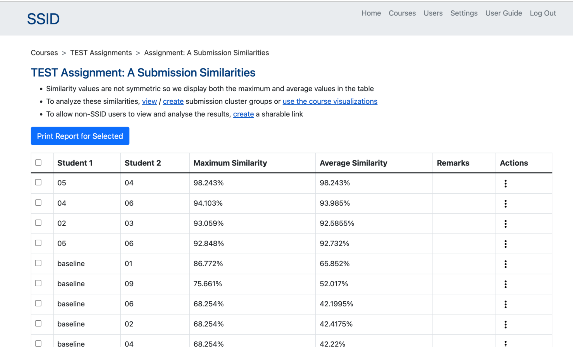
400	Error	{ "error": "Value of {parameterName} is not valid. {state the constraint/ limit}" }
		{ "error": "Missing required parameter {parameterName}" }
		{ "error": "Parameter {parameterName} is invalid or not yet supported." }

401	Unauthorized	{ "error": "Missing or invalid API key." } { "error": "Your API key is not authorized to access this resource." }
503	Service Unavailable	{ "error": "SSID is busy or under maintenance. Please try again later." } [TODO: discuss with team how to set maintenance status]

View pairwise submissions comparison results

GET /api/v1/assignments/{assignment_id}/submission_similarities/

Description: Returns all submission similarities of an assignment. The web equivalence is going to /assignments/{assignment_id}/submission_similarities/



Courses > TEST Assignments > Assignment: A Submission Similarities

TEST Assignment: A Submission Similarities

- Similarity values are not symmetric so we display both the maximum and average values in the table
- To analyze these similarities, view / create submission cluster groups or use the course visualizations
- To allow non-SSID users to view and analyse the results, create a sharable link

□	Student 1	Student 2	Maximum Similarity	Average Similarity	Remarks	Actions
□	05	04	98.243%	98.243%		⋮
□	04	06	94.103%	93.985%		⋮
□	02	03	93.059%	92.5855%		⋮
□	05	06	92.848%	92.732%		⋮
□	baseline	01	86.772%	65.852%		⋮
□	baseline	09	75.661%	52.017%		⋮
□	baseline	06	68.254%	42.1995%		⋮
□	baseline	02	68.254%	42.4175%		⋮
□	baseline	04	68.254%	42.22%		⋮

Params:

- threshold (Optional) number: A number between 0 and 100. If specified, returns only submission similarities whose similarity percentage is between t and 1 inclusive. Otherwise, returns all submission similarities. [Note: This function is not yet available in SSID, so we need to first build it for SSID, then allow it to be used via API].
- page (Optional) number: If specified, returns submission similarities in such page when sorted by highest maximum similarity percentage. Otherwise, returns all submission similarities. [Note: Currently this is already available in SSID's web interface].
- limit (Optional) number: If specified, returns such number of submission similarities with highest maximum similarity percentage. Otherwise, returns all submission similarities. [Note: Currently SSID stores and displays all submission similarities on its web interface. The function to take top N submissions is not yet available in SSID, so we need to first build it for SSID, then allow it to be used via API].

Request example:

Header

{
  "Content-Type": "application/json",
  "X-Api-Key": "YOUR_API_KEY"
}

Body
<empty>

Possible responses:

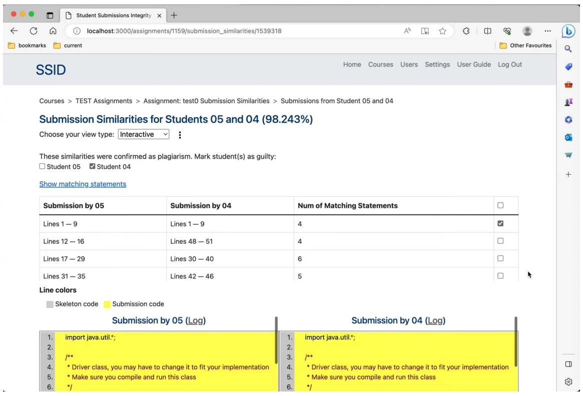
Code	Status	Return body

200	Successful	{ "status": "empty" } [Note: This status, currently in SSID's web interface, is for when the user did not include a zip file when creating an assignment. To rectify this, the user must upload the file via the web interface. When submitting through API, the zip assignment file is compulsory.]
		
		{ "status": "processing" }
		
		{
		  "status": "processed",
		  "submissionSimilarities": []
		}

400	Error	{ “error”: “Submission similarities requested do not exist." }
401	Unauthorized	{ "error": "Missing or invalid API key." } { "error": "Your API key is not authorized to access this resource." }
503	Service Unavailable	{ "error": "SSID is busy or under maintenance. Please try again later." }

GET /api/v1/assignments/{assignment_id}/submission_similarities/{submission_similarities_id}

Description: Returns details of a pair of a flagged submissions. Please use SSID's web interface to view and mark students as suspicious or guilty. The web equivalence is going to /assignments/{assignment_id}/submission_similarities/{submission_similarity_id}.



Student Submissions Integrity
localhost:3000/assignments/1159/submission_similarities/1539318
bookmarks
current
Other Favourites
SSID
Home
Courses
Users
Settings
User Guide
Log Out

Courses > TEST Assignments > Assignment: test0 Submission Similarities > Submissions from Student 05 and 04

Submission Similarities for Students 05 and 04 (98.243%)

Choose your view type: Interactive

These similarities were confirmed as plagiarism. Mark student(s) as guilty:

□ Student 05 ☑ Student 04

Show matching statements

Submission by 05	Submission by 04	Num of Matching Statements	□
Lines 1 — 9	Lines 1 — 9	4	☑
Lines 12 — 16	Lines 48 — 51	4	□
Lines 17 — 29	Lines 30 — 40	6	□
Lines 31 — 35	Lines 42 — 46	5	□

Line colors

▨ Skeleton code 🟨 Submission code

Submission by 05 (Log)

```
import java.util.*;

/**
 * Driver class, you may have to change it to fit your implementation
 * Make sure you compile and run this class
 */
```

Submission by 04 (Log)

```
import java.util.*;

/**
 * Driver class, you may have to change it to fit your implementation
 * Make sure you compile and run this class
 */
```

Params:

- No param. Pass in desired assignment's id in and submission similarity's id in the URL.

Request example:

Header

{
  "Content-Type": "application/json",
  "X-Api-Key": "YOUR_API_KEY"
}

Body
<empty>

Possible responses:

Code	Status	Return body

200	Successful	{
		    "maxSimilaryPercentage": ...,
		    "matches": []
		}
		[Note: max similarity percentage is between 0 and 100]

400	Error	{ “error”: “Submission similarities requested do not exist." }
401	Unauthorized	{ "error": "Missing or invalid API key." } { "error": "Your API key is not authorized to access this resource." }
503	Service Unavailable	{ "error": "SSID is busy or under maintenance. Please try again later." }