

Module Interface Specification for SyncMaster

Team 15, SyncMaster

Kyle D'Souza

Mitchell Hynes

Richard Fan

Akshit Gulia

Rafeed Iqbal

March 10, 2025

1 Revision History

Date	Version	Notes
1/17/2025	1.0	Initial Draft of MIS for Rev0
3/10/2025	1.1	Remove Job Management Module

2 Symbols, Abbreviations and Acronyms

See SRS Documentation at <https://github.com/Spitgranger/SyncMaster/blob/main/docs/SRS-Volere/SRS.pdf>.

Contents

1	Revision History	i
2	Symbols, Abbreviations and Acronyms	ii
3	Introduction	1
4	Notation	1
5	Module Decomposition	2
6	MIS of Audit and Compliance Module	4
6.1	Module	4
6.2	Uses	4
6.3	Syntax	4
6.3.1	Exported Constants	4
6.3.2	Exported Access Programs	4
6.4	Semantics	4
6.4.1	State Variables	4
6.4.2	Environment Variables	5
6.4.3	Assumptions	5
6.4.4	Access Routine Semantics	5
6.4.5	Local Functions	6
7	MIS of Database Interaction Module	6
7.1	Module	6
7.2	Uses	6
7.3	Syntax	6
7.3.1	Exported Constants	6
7.3.2	Exported Access Programs	6
7.4	Semantics	7
7.4.1	State Variables	7
7.4.2	Environment Variables	8
7.4.3	Assumptions	8
7.4.4	Access Routine Semantics	8
7.4.5	Local Functions	9
8	MIS of Logging Module	9
8.1	Module	9
8.2	Uses	9
8.3	Syntax	9
8.3.1	Exported Constants	9
8.3.2	Exported Access Programs	9

8.4	Semantics	10
8.4.1	State Variables	10
8.4.2	Environment Variables	10
8.4.3	Assumptions	10
8.4.4	Access Routine Semantics	10
8.4.5	Local Functions	11
9	MIS of Analytics and Reporting Module	11
9.1	Module	11
9.2	Uses	11
9.3	Syntax	11
9.3.1	Exported Constants	11
9.3.2	Exported Access Programs	11
9.4	Semantics	12
9.4.1	State Variables	12
9.4.2	Environment Variables	12
9.4.3	Assumptions	12
9.4.4	Access Routine Semantics	12
9.4.5	Local Functions	13
10	MIS of File Storage Interaction Module	13
10.1	Module	13
10.2	Uses	13
10.3	Syntax	13
10.3.1	Exported Constants	13
10.3.2	Exported Access Programs	13
10.4	Semantics	14
10.4.1	State Variables	14
10.4.2	Environment Variables	14
10.4.3	Assumptions	15
10.4.4	Access Routine Semantics	15
10.4.5	Local Functions	15
11	MIS of Function Compute Module	15
11.1	Module	15
11.2	Uses	15
11.3	Syntax	16
11.3.1	Exported Constants	16
11.3.2	Exported Access Programs	16
11.4	Semantics	16
11.4.1	State Variables	16
11.4.2	Environment Variables	16
11.4.3	Assumptions	16

11.4.4	Access Routine Semantics	16
11.4.5	Local Functions	16
12	MIS of Routing Module	17
12.1	Module	17
12.2	Uses	17
12.3	Syntax	17
12.3.1	Exported Constants	17
12.3.2	Exported Access Programs	17
12.4	Semantics	17
12.4.1	State Variables	17
12.4.2	Environment Variables	17
12.4.3	Assumptions	17
12.4.4	Access Routine Semantics	18
12.4.5	Local Functions	18
13	MIS of Job Management Module	18
13.1	Module	18
13.2	Uses	18
13.3	Syntax	18
13.3.1	Exported Constants	18
13.3.2	Exported Access Programs	19
13.4	Semantics	19
13.4.1	State Variables	19
13.4.2	Environment Variables	19
13.4.3	Assumptions	20
13.4.4	Access Routine Semantics	20
13.4.5	Local Functions	20
14	MIS of Location Verification Module	20
14.1	Module	20
14.2	Uses	21
14.3	Syntax	21
14.3.1	Exported Constants	21
14.3.2	Exported Access Programs	21
14.4	Semantics	21
14.4.1	State Variables	21
14.4.2	Environment Variables	21
14.4.3	Assumptions	21
14.4.4	Access Routine Semantics	21
14.4.5	Local Functions	22

15 MIS of User Management Module	22
15.1 Module	22
15.2 Uses	22
15.3 Syntax	22
15.3.1 Exported Constants	22
15.3.2 Exported Access Programs	22
15.4 Semantics	23
15.4.1 State Variables	23
15.4.2 Environment Variables	23
15.4.3 Assumptions	23
15.4.4 Access Routine Semantics	23
15.4.5 Local Functions	24
16 MIS of User Authentication Module	24
16.1 Module	24
16.2 Uses	24
16.3 Syntax	24
16.3.1 Exported Constants	24
16.3.2 Exported Access Programs	25
16.4 Semantics	25
16.4.1 State Variables	25
16.4.2 Environment Variables	25
16.4.3 Assumptions	25
16.4.4 Access Routine Semantics	25
16.4.5 Local Functions	26
17 MIS of API Integration Module	26
17.1 Module	26
17.2 Uses	26
17.3 Syntax	26
17.3.1 Exported Constants	26
17.3.2 Exported Access Programs	26
17.4 Semantics	26
17.4.1 State Variables	26
17.4.2 Environment Variables	27
17.4.3 Assumptions	27
17.4.4 Access Routine Semantics	27
17.4.5 Local Functions	27
18 MIS of Document Management Module	28
18.1 Module	28
18.2 Uses	28
18.3 Syntax	28

18.3.1	Exported Constants	28
18.3.2	Exported Access Programs	28
18.4	Semantics	29
18.4.1	State Variables	29
18.4.2	Environment Variables	29
18.4.3	Assumptions	29
18.4.4	Access Routine Semantics	29
18.4.5	Local Functions	30
19	Appendix	32
19.1	Symbolic Parameters	32

3 Introduction

The following document details the Module Interface Specifications for SyncMaster, a facilities management application for the Technical Services team at the Water Division of the City of Hamilton. This application enables the Technical Services team to effectively distribute water station documentation to stakeholders as a single source of truth. It also acts as an authentication tool for external contractors to verify their presence at stations and confirm work being performed.

Complementary documents include the System Requirement Specifications and Module Guide. The full documentation and implementation can be found at <https://github.com/Spitgranger/SyncMaster>.

4 Notation

The structure of the MIS for modules comes from Hoffman and Strooper (1995), with the addition that template modules have been adapted from Ghezzi et al. (2003). The mathematical notation comes from Chapter 3 of Hoffman and Strooper (1995). For instance, the symbol $:=$ is used for a multiple assignment statement and conditional rules follow the form $(c_1 \Rightarrow r_1 | c_2 \Rightarrow r_2 | \dots | c_n \Rightarrow r_n)$.

The following table summarizes the primitive data types used by SyncMaster.

Data Type	Notation	Description
character	char	a single symbol or digit
integer	\mathbb{Z}	a number without a fractional component in $(-\infty, \infty)$
natural number	\mathbb{N}	a number without a fractional component in $[1, \infty)$
real	\mathbb{R}	any number in $(-\infty, \infty)$
boolean	boolean	either true or false
any	any	any data type

The specification of SyncMaster uses some derived data types: sequences, strings, tuples, map, enum, KeyCondition, AttributeCondition, S3File, HTTPRequest, HTTPResponse, and Optional. Sequences are lists filled with elements of the same data type. Strings are sequences of characters. Tuples contain a list of values, potentially of different types. In addition, SyncMaster uses functions, which are defined by the data types of their inputs and outputs. Local functions are described by giving their type signature followed by their specification. Maps are a collection of key-value pairs, where there does not necessarily need to be a restriction on the data types of the keys and values, but one can be placed. An enum is a set of values of which the data can be. AttributeConditions are conditionals placed on

an attribute of a database entry. KeyConditions are a subset of AttributeConditions, and are conditionals placed on the key attributes of a database entry. An S3File, is a unique file that exists in AWS S3. HTTPRequest and HTTPResponse are a subset of maps which conform to the HTTP standard. An Optional is not a datatype by itself, but specifies that there may be the absence of a value of a specific type. LogEntry is an object consisting of a UserID, SiteID, an ISO formatted date and time string, and a type of log (entry or exit). A Document is an object consisting of the following fields [siteId: string, expiryDate: string, name: string, createdDatetime: string, lastEditedDateTime: string, s3Link: string, parentDocumentId: map[string → any], requireAck: boolean, userId: string]

5 Module Decomposition

The following table is taken directly from the Module Guide document for this project.

Level 1	Level 2
Hardware-Hiding Module	N/A
Software Decision Modules	Audit and Compliance Module
	User Authentication Module
	Location Verification Module
	Logging Module
	Analytics and Reporting Module
	User Management Module
	Document Management Module
Behaviour-Hiding Modules	Job Management Module
	API Integration Module
	Database Interaction Module
	Blob Storage Interaction Module
	Request Routing Module
	Function Compute Module

Table 1: Module Hierarchy

6 MIS of Audit and Compliance Module

6.1 Module

Analytics and Reporting Module

6.2 Uses

Database Interaction Module

6.3 Syntax

6.3.1 Exported Constants

N/A

6.3.2 Exported Access Programs

Name	In	Out	Exceptions
GetSignedDocuments	FromDatetime: string ToDatetime: string	sequence[map[string → any]]	ExternalServiceFailure: An internal error from AWS
GetUnsignedDocuments	FromDatetime: string ToDatetime: string	sequence[map[string → any]]	ExternalServiceFailure: An internal error from AWS
GetTrainingData	FromDatetime: string ToDatetime: string	sequence[map[string → any]]	ExternalServiceFailure: An internal error from AWS
GetIncidentsData	FromDatetime: string ToDatetime: string	sequence[map[string → any]]	ExternalServiceFailure: An internal error from AWS
GetWorkOrders	FromDatetime: string ToDatetime: string	sequence[map[string → any]]	ExternalServiceFailure: An internal error from AWS
GetFlaggedContractors	FromDatetime: string ToDatetime: string	sequence[map[string → any]]	ExternalServiceFailure: An internal error from AWS

6.4 Semantics

6.4.1 State Variables

N/A

6.4.2 Environment Variables

Name	Description
LAMBDA_EXECUTION_ROLE	When an AWS Lambda Function (The chosen AWS compute service for this project), it has an AWS IAM role attached to it, that it uses when running. This role needs permission for database access for this module to work

6.4.3 Assumptions

LAMBDA_EXECUTION_ROLE has the required permissions in AWS to access the database.

6.4.4 Access Routine Semantics

GetSignedDocuments(FromDatetime: string, ToDatetime: string):

- output: sequence[map[string → any]]
- exception: ExternalServiceFailure

GetUnsignedDocuments(FromDatetime: string, ToDatetime: string):

- output: sequence[map[string → any]]
- exception: ExternalServiceFailure

GetTrainingData(FromDatetime: string, ToDatetime: string):

- output: sequence[map[string → any]]
- exception: ExternalServiceFailure

GetIncidentsData(FromDatetime: string, ToDatetime: string):

- output: sequence[map[string → any]]
- exception: ExternalServiceFailure

GetWorkOrders(FromDatetime: string, ToDatetime: string):

- output: sequence[map[string → any]]
- exception: ExternalServiceFailure

GetFlaggedContractors(FromDatetime: string, ToDatetime: string):

- output: sequence[map[string → any]]
- exception: ExternalServiceFailure

6.4.5 Local Functions

N/A

7 MIS of Database Interaction Module

7.1 Module

Database Interaction Module

7.2 Uses

boto3 (AWS SDK for Python), AWS DynamoDB (AWS Cloud Service for NoSQL Databases)

7.3 Syntax

7.3.1 Exported Constants

Name	Description
DBTable.Name	The name of the underlying DynamoDB table resource.
DBTable.Access	The level of access the DBTable object has on the DynamoDB table. Either “read” or “write”.

7.3.2 Exported Access Programs

Name	In	Out	Exceptions
DBTable	TableName: string Access: enum[“read”, “write”]	DBTable	-
DBTable.get	Key: map[string → any]	map[string → any]	ItemNotFound: Item with requested key does not exist in the database ExternalServiceFailure: An internal error from AWS
DBTable.put	Item: map[string → any] Condition: AttributeCondition	map[string → any]	ConditionCheckFailed: The given condition is not met ExternalServiceFailure: An internal error from AWS PermissionException: If the current access level is read-only
DBTable.delete	Key: map[string → any] Condition: AttributeCondition	map[string → any]	ExternalServiceFailure: An internal error from AWS ConditionCheckFailed: The given condition is not met PermissionException: If the current access level is read-only
DBTable.query	KeyConditions: sequence[KeyCondition] AttributeConditions: sequence[AttributeCondition]	sequence[map[string → any]]	ExternalServiceFailure: An internal error from AWS

7.4 Semantics

7.4.1 State Variables

Name	Description
Database	The underlying AWS DynamoDB table, can be represented as a set of items: $\{i_0, i_1, \dots, i_n\}$, where $i_k : \text{map}[\text{string} \rightarrow \text{any}], k \in [0, n]$

7.4.2 Environment Variables

Name	Description
LAMBDA_EXECUTION_ROLE	When an AWS Lambda Function (The chosen AWS compute service for this project), it has an AWS IAM role attached to it, that it uses when running. This role needs permission for database access for this module to work

7.4.3 Assumptions

LAMBDA_EXECUTION_ROLE has the required permissions in AWS to access the database.

7.4.4 Access Routine Semantics

DBTable.get(Key: k):

- output: i_k , where $i_k \in Database \wedge dbKey(i_k) == k$
- exception: ExternalServiceFailure

DBTable.put(Item: i_{new} , Condition: c):

- transition: $Database \rightarrow Database \cup \{i_{new}\}$, if $c == true \wedge DBTable.Access == "write"$
- output: i_{new}
- exception: ExternalServiceFailure, ConditionCheckFailed, PermissionException

DBTable.delete(Key: k, Condition: c):

- transition: $Database \rightarrow Database - \{i_{old}\}$, where $k == dbKey(i_{old}, DBTable.Name)$, if $c == true \wedge DBTable.Access == "write"$
- output: i_{old}
- exception: ExternalServiceFailure, ConditionCheckFailed, PermissionException

DBTable.query(KeyConditions: kc, AttributeConditions: ac):

- output: $[i_k | i_k \in Database \wedge \forall c \in kc (c == true) \wedge \forall c \in ac (c == true)]$
- exception: ExternalServiceFailure

7.4.5 Local Functions

Name	In	Out	Description
dbKey	Item: map[string → any] TableName: string	map[string → any]	Returns the keys of the given db item, assuming it is from the given table

8 MIS of Logging Module

8.1 Module

Logging Module

8.2 Uses

Database Interaction Module

8.3 Syntax

8.3.1 Exported Constants

N/A

8.3.2 Exported Access Programs

Name	In	Out	Exceptions
AddLog	UserID: string SiteID: string Datetime: string Type: enum[“entry”, “exit”]	LogEntry	ExternalServiceFailure: An internal error from AWS
ListLogs	UserID: string SiteID: string FromDatetime: string ToDatetime: string	sequence[LogEntry]	ExternalServiceFailure: An internal error from AWS
PurgeLogs	UserID: string	-	ExternalServiceFailure: An internal error from AWS

8.4 Semantics

8.4.1 State Variables

Name	Description
LogEntryDatabase	The underlying AWS DynamoDB table, can be represented as a set of items: $\{l_0, l_1, \dots, l_n\}$, where $l_k : LogEntry, k \in [0, n]$

8.4.2 Environment Variables

Name	Description
LAMBDA_EXECUTION_ROLE	When an AWS Lambda Function (The chosen AWS compute service for this project), it has an AWS IAM role attached to it, that it uses when running. This role needs permission for logging database access for this module to work

8.4.3 Assumptions

LAMBDA_EXECUTION_ROLE has the required permissions in AWS to access the database.

8.4.4 Access Routine Semantics

AddLog(UserID: u, SiteID: s, Datetime: d, Type: t):

- transition: $LogEntryDatabase \rightarrow LogEntryDatabase \cup \{l_{new}\}$, where $l_{new}.UserID = u, l_{new}.SiteID = s, l_{new}.Datetime = d, l_{new}.Type = t$
- output: l_{new}
- exception: ExternalServiceFailure

ListLogs(UserID: u, SiteID: s, FromDatetime: fd, ToDatetime: td):

- output: $[l_k | l_k \in LogEntryDatabase \wedge l_k.UserID == u \wedge l_k.SiteID == s \wedge fd \leq l_k.Datetime \leq td]$
- exception: ExternalServiceFailure

PurgeLogs(UserID: u):

- transition: $LogEntryDatabase \rightarrow LogEntryDatabase - \{l_k | l_k \in LogEntryDatabase \wedge l_k.UserID == u\}$
- exception: ExternalServiceFailure

8.4.5 Local Functions

N/A

9 MIS of Analytics and Reporting Module

9.1 Module

Analytics and Reporting Module

9.2 Uses

Database Interaction Module

9.3 Syntax

9.3.1 Exported Constants

N/A

9.3.2 Exported Access Programs

Name	In	Out	Exceptions
GetErrorsReport	FromDatetime: string ToDatetime: string	sequence[map[string → any]]	ExternalServiceFailure: An internal error from AWS
GetResourceUsageReport	FromDatetime: string ToDatetime: string Resources: sequence[resource]	sequence[map[string → any]]	ExternalServiceFailure: An internal error from AWS
GetUserActivityReport	FromDatetime: string ToDatetime: string UserID: string	sequence[map[string → any]]	ExternalServiceFailure: An internal error from AWS
GetSystemHealthReport	FromDatetime: string ToDatetime: string	sequence[map[string → any]]	ExternalServiceFailure: An internal error from AWS
GetLoginAttemptsReport	FromDatetime: string ToDatetime: string	sequence[map[string → any]]	ExternalServiceFailure: An internal error from AWS

9.4 Semantics

9.4.1 State Variables

N/A

9.4.2 Environment Variables

Name	Description
LAMBDA_EXECUTION_ROLE	When an AWS Lambda Function (The chosen AWS compute service for this project), it has an AWS IAM role attached to it, that it uses when running. This role needs permission for database access for this module to work

9.4.3 Assumptions

LAMBDA_EXECUTION_ROLE has the required permissions in AWS to access the database.

9.4.4 Access Routine Semantics

GetErrorsReport(FromDatetime: string, ToDatetime: string):

- output: sequence[map[string → any]]
- exception: ExternalServiceFailure

GetResourceUsageReport(FromDatetime: string, ToDatetime: string):

- output: sequence[map[string → any]]
- exception: ExternalServiceFailure

GetUserActivityReport(FromDatetime: string, ToDatetime: string):

- output: sequence[map[string → any]]
- exception: ExternalServiceFailure

GetSystemHealthReport(FromDatetime: string, ToDatetime: string):

- output: sequence[map[string → any]]
- exception: ExternalServiceFailure

GetLoginAttemptsReport(FromDatetime: string, ToDatetime: string):

- output: sequence[map[string → any]]
- exception: ExternalServiceFailure

9.4.5 Local Functions

N/A

10 MIS of File Storage Interaction Module

10.1 Module

File Storage Interaction Module

10.2 Uses

boto3 (AWS SDK for Python), AWS S3 (AWS Cloud Service for storing files)

10.3 Syntax

10.3.1 Exported Constants

Name	Description
S3Bucket.Name	The name of the underlying S3 Bucket resource.
S3Bucket.Access	The level of access the S3Bucket object has on the S3 Bucket resource in AWS. Either “read” or “write”.

10.3.2 Exported Access Programs

Name	In	Out	Exceptions
S3Bucket	BucketName: string Access: enum[“read”, “write”]	S3Bucket	-
S3Bucket.createPresignedUrl	Key: string VersionID: string ETag: string Method: enum[“get”, “upload”] ExpiresIn: integer	string	PermissionException: If attempting to get an upload url, while only having read permissions ExternalServiceFailure: An internal error from AWS
S3Bucket.delete	Key: string VersionID: string ETag: string	-	FileNotFound: The given key and version do not match any file in the bucket ETagMismatch: The given ETag does not match the ETag of the file with the given key and version ExternalServiceFailure: An internal error from AWS PermissionException: If the current access level is read-only

10.4 Semantics

10.4.1 State Variables

Name	Description
Bucket	The underlying AWS S3 Bucket, can be represented as a set of files: $\{f_0, f_1, \dots, f_n\}$, where $f_k : S3File \wedge k \in [0, n]$

10.4.2 Environment Variables

Name	Description
LAMBDA_EXECUTION_ROLE	When an AWS Lambda Function (The chosen AWS compute service for this project), it has an AWS IAM role attached to it, that it uses when running. This role needs permission for S3 bucket access for this module to work

10.4.3 Assumptions

LAMBDA_EXECUTION_ROLE has the required permissions in AWS to access the S3 bucket.

10.4.4 Access Routine Semantics

S3Bucket.createPresignedUrl(Key: k, VersionID: v, ETag: e, Method: m, ExpiresIn: x):

- output: string
- exception: ExternalServiceFailure, PermissionException

S3Bucket.delete(Key: k, VersionID: v, ETag: e):

- transition: $Bucket \rightarrow Bucket - \{fold\}$, where $map\{Key : k, VersionID : v, ETag : e\} == S3Bucket.metadata(fold)$, if $S3Bucket.Access == "write"$
- exception: ExternalServiceFailure, ConditionCheckFailed, PermissionException

10.4.5 Local Functions

Name	In	Out	Description
S3Bucket.metadata	File: S3File	map[string \rightarrow any]	Returns S3 key, versionID, and Etag of the given file, assuming it is from the given S3Bucket

11 MIS of Function Compute Module

11.1 Module

Function Compute Module

11.2 Uses

AWS Lambda (AWS Service that executes code in response to events and manages the compute resources needed to run the code)

All other modules run on function compute. Therefore, this module uses all the others, except for the Routing Module, and API Integration Module.

11.3 Syntax

11.3.1 Exported Constants

N/A

11.3.2 Exported Access Programs

Name	In	Out	Exceptions
Invoke	FunctionName: string Event: map[any → any]	map[any → any]	ExternalServiceFailure: An internal error from AWS ExecutionError: Any error that occurs while the function is running

11.4 Semantics

11.4.1 State Variables

N/A

11.4.2 Environment Variables

Name	Description
LAMBDA.EXECUTION_ROLE	When an AWS Lambda Function runs, it has an AWS IAM role attached to it, which it uses when running. This role gives the function the necessary permissions to execute without issue.

11.4.3 Assumptions

LAMBDA.EXECUTION_ROLE has the required permissions in AWS to execute the lambda's required tasks.

11.4.4 Access Routine Semantics

N/A

11.4.5 Local Functions

N/A

12 MIS of Routing Module

12.1 Module

Routing Module

12.2 Uses

AWS APIGateway (AWS Cloud Service for handling routing of API to an underlying serverless function), User Authentication Module

12.3 Syntax

12.3.1 Exported Constants

Name	Description
BaseUrl	The base url of the REST API

12.3.2 Exported Access Programs

Name	In	Out	Exceptions
SubmitRequest	Request: $\text{map}[\text{any} \rightarrow \text{any}]$ Path: String	$\text{map}[\text{any} \rightarrow \text{any}]$	ExternalServiceFailure: An internal error from AWS ExecutionError: If the underlying compute resource that the request gets routed to encounters an error during execution

12.4 Semantics

12.4.1 State Variables

N/A

12.4.2 Environment Variables

N/A

12.4.3 Assumptions

N/A

12.4.4 Access Routine Semantics

N/A

12.4.5 Local Functions

N/A

13 ~~MIS of Job Management Module~~

13.1 Module

~~Module for contractors to enter their job information.~~

13.2 Uses

~~Database Interaction Module~~

13.3 Syntax

13.3.1 Exported Constants

N/A

13.3.2 Exported Access Programs

Name	In	Out	Exceptions
createJob	jobType: string, workOrder: int, userID: string, description: string	jobID: string	ExternalServiceFailure: An internal error from AWS DuplicateWorkOrder: Job with same work order already in database
editJob	jobID: string, workOrder: int, description: string, jobType: string	None	ExternalServiceFailure: An internal error from AWS JobAlreadyCompleted: This job is already closed
completeJob	jobID: string	None	ExternalServiceFailure: An internal error from AWS
deleteJob	jobID: string	None	ExternalServiceFailure: An internal error from AWS JobAlreadyCompleted: This job is already closed

13.4 Semantics

13.4.1 State Variables

Name	Description
jobTypes	Set of available jobs $\{j_0, j_1, \dots, j_n\}$, where $j_k : \text{string}, k \in [0, n]$
jobType	$\text{string} \in \text{jobTypes}$
jobID	string identifying a created job
workOrder	number of type int
description	work description of type string
status	status of job, value in $\{\text{open}, \text{closed}\}$

13.4.2 Environment Variables

Name	Description
LAMBDA_EXECUTION_ROLE	When an AWS Lambda Function (The chosen AWS compute service for this project), it has an AWS IAM role attached to it, that it uses when running. This role needs permission for job database access for this module to work

13.4.3 Assumptions

LAMBDA_EXECUTION_ROLE has the required permissions in AWS to execute the lambda's required tasks.

13.4.4 Access Routine Semantics

`createJob(jobType:string, workOrder:int, userID:string, description:string):`

- transition: creates job associated with userID
- output: the jobID
- exception: ExternalServiceFailure, DuplicateWorkOrder

`editJob(jobID:string, workOrderint, descriptionstring, jobTypestring):`

- transition: edit job attributes if not closed
- exception: ExternalServiceFailure, JobAlreadyCompleted

`completeJob(jobID:string):`

- transition: status == closed
- exception: ExternalServiceFailure

`deleteJob(jobID:string):`

- transition: remove job if status is open
- exception: ExternalServiceFailure, JobAlreadyCompleted

13.4.5 Local Functions

N/A

14 MIS of Location Verification Module

14.1 Module

Location Verification Module

14.2 Uses

Browser location/GPS API, Database Interaction Module

14.3 Syntax

14.3.1 Exported Constants

N/A

14.3.2 Exported Access Programs

Name	In	Out	Exceptions
verifyLocation	siteID: string latitude: float longitude: float accuracy: float	locationState: boolean	Invalidlocation: co-ordinates are invalid

14.4 Semantics

14.4.1 State Variables

N/A

14.4.2 Environment Variables

Name	Description
LAMBDA_EXECUTION_ROLE	When an AWS Lambda Function (The chosen AWS compute service for this project), it has an AWS IAM role attached to it, that it uses when running. This role needs permission for site database access for this module to work

14.4.3 Assumptions

LAMBDA_EXECUTION_ROLE has the required permissions in AWS to execute the lambda's required tasks.

14.4.4 Access Routine Semantics

verifyLocation(siteID: siteID, latitude: lat, longitude: long, accuracy: acc):

- output: true if the distance calculated by calculateDistance is within the range of the intended site, adjusted for accuracy; false otherwise.

- exception: Invalidlocation

14.4.5 Local Functions

Name	In	Out	Description
calculateDistance	siteID: string latitude: float longitude: float	distance: float	Uses siteID to get a second set of coordinates from Database. Using two sets of coordinates the haversine distance between the two points is returned.

15 MIS of User Management Module

15.1 Module

User Management Module

15.2 Uses

boto3 (AWS SDK for Python), AWS Cognito (AWS Cloud Service for user authentication), Database Interaction Module

15.3 Syntax

15.3.1 Exported Constants

N/A

15.3.2 Exported Access Programs

Name	In	Out	Exceptions
createUser	email: string details: map[string → any]	userID: string	DuplicateUser: Existing email address used
editUser	userID: string edit: map[string → any]	boolean	UserNotFound: userID not in database
deleteUser	userID: string	boolean	UserNotFound: userID not in database
getUser	userID: string	map[string → any]	UserNotFound: userID not in database

15.4 Semantics

15.4.1 State Variables

Name	Description
Database	Set of registered users, can be represented as set of items $\{i_0, i_1, \dots, i_n\}$, where $i_k : \text{map}[\text{string} \rightarrow \text{any}], k \in [0, n]$

15.4.2 Environment Variables

Name	Description
LAMBDA_EXECUTION_ROLE	When an AWS Lambda Function runs, it has an AWS IAM role attached to it, which it uses when running. This role gives the function the necessary permissions to execute without issue.

15.4.3 Assumptions

LAMBDA_EXECUTION_ROLE has the required permissions in AWS to execute the lambda's required tasks.

15.4.4 Access Routine Semantics

createUser(email: email, details: details):

- transition: $\text{Database} \rightarrow \text{Database} \cup \{User_{new}\}$, if $\text{email} \notin \text{Database}$, where $User_{new} = \{userID, \text{email}, \text{password}, \dots\}$.
- output: $\{userID, \text{password}\}$, where $userID$ is uniquely generated for the email and password is a temporary password generated during account creation
- exception: DuplicateUser

editUser(userID: userID, edit: changes):

- transition: $\text{Database} \rightarrow \text{Database} \cup \{User_{edited}\}$, if $(userID \in \text{Database}) \wedge (\text{changes.email} \notin \text{Database})$, where $User_{edited} = \{userID, \text{changes}\}$
- output: true if transition successful, false otherwise.
- exception: UserNotFound

deleteUser(userID: userID):

- transition: $\text{Database} \rightarrow \text{Database} - \{User\}$, if $\exists (\{User\} \in \text{Database} \wedge User.userID == userID)$.

- output: true if transition successful, false otherwise.
- exception: UserNotFound

getUser(userID: userID):

- output: $\{User\}$, if $\exists(\{User\} \in Database \wedge User.userID == userID)$.
- exception: UserNotFound

15.4.5 Local Functions

Name	In	Out	Description
generateUserID	email: string	authToken: string	Generates unique userID for each email.
generatePassword	password: string	authToken: string	Creates one-time password for new users.

16 MIS of User Authentication Module

16.1 Module

User Authentication Module

16.2 Uses

boto3 (AWS SDK for Python), AWS Cognito (AWS Cloud Service for user authentication), Location Verification Module, Database Interaction Module

16.3 Syntax

16.3.1 Exported Constants

N/A

16.3.2 Exported Access Programs

Name	In	Out	Exceptions
authenticateUser	email: string password: string	authToken: string	InvalidCredentials: Credentials not in database
authenticateContractor	email: string name: string siteID: String userLocation: {float, float}	authToken: string	InvalidCredentials: Credentials not in database InvalidSiteID: siteID not in database LocationVerificationFailed: Verification of users location failed

16.4 Semantics

16.4.1 State Variables

N/A

16.4.2 Environment Variables

Name	Description
LAMBDA.EXECUTION_ROLE	When an AWS Lambda Function runs, it has an AWS IAM role attached to it, which it uses when running. This role gives the function the necessary permissions to execute without issue.

16.4.3 Assumptions

LAMBDA.EXECUTION_ROLE has the required permissions in AWS to execute the lambda's required tasks.

16.4.4 Access Routine Semantics

authenticateUser(email: email, password: password):

- output: *authToken*, where *authToken* is a unique token generated for the user, tracked, and validated by AWS Cognito.
- exception: InvalidCredentials

authenticateContractor(email: email, name: name, siteID: siteID, userLocation: {latitude, longitude}):

- output: *authToken*, where *authToken* is a unique token generated for the user, tracked, and validated by AWS Cognito.
- exception: InvalidCredentials, InvalidSiteID, LocationVerificationFailed

16.4.5 Local Functions

N/A

17 MIS of API Integration Module

17.1 Module

API Integration Module

17.2 Uses

Routing Module

17.3 Syntax

17.3.1 Exported Constants

N/A

17.3.2 Exported Access Programs

Name	In	Out	Exceptions
submitApiRequest	request: HTTPRequest url: String apiToken (optional): String	HTTPResponse	NetworkException: If a valid network connection is not detected. TimeoutException: If a response is not received within <i>TIMEOUT</i> seconds.

17.4 Semantics

17.4.1 State Variables

N/A

17.4.2 Environment Variables

17.4.3 Assumptions

- API endpoints are up and functional
- The system has a internet network connection

17.4.4 Access Routine Semantics

submitApiRequest(url: String, request: HTTPRequest, apiToken: Optional<String>):

- output: HTTPResponse
- exception: NetworkException, TimeoutException

17.4.5 Local Functions

Name	In	Out	Description
timeElapsed	since: \mathbb{Z}	\mathbb{Z}	Returns the number of seconds elapsed since the provided time given in seconds since January 1, 1970

18 MIS of Document Management Module

18.1 Module

Document Management Module

18.2 Uses

Database Interaction Module

18.3 Syntax

18.3.1 Exported Constants

N/A

18.3.2 Exported Access Programs

Name	In	Out	Exceptions
RetrieveDocs	siteID: string	sequence[Document]	ExternalServiceFailure: An internal error from AWS
CreateDoc	s3Link: string userID: string siteID: string parentDocumentID: Optional <map[string → any]> expiryDate: Optional <string> requiresAck: boolean	map[string → any]	ExternalServiceFailure: An internal error from AWS ValidationError: Non-existent IDs provided
EditDoc	documentID: map[string → any userID: string s3Link: string	-	ExternalServiceFailure: An internal error from AWS ValidationError: Non-existent IDs provided
DeleteDoc	documentID: map[string → any]	-	ExternalServiceFailure: An internal error from AWS ValidationError: Non-existent IDs provided

18.4 Semantics

18.4.1 State Variables

Name	Description
DocumentDatabase	The underlying AWS DynamoDB table, can be represented as a set of items: $\{D_0, D_1, \dots, D_n\}$, where $D_k \in Documents, k \in [0, n]$

18.4.2 Environment Variables

Name	Description
LAMBDA_EXECUTION_ROLE	When an AWS Lambda Function (The chosen AWS compute service for this project), it has an AWS IAM role attached to it, that it uses when running. This role needs permission for document database access for this module to work

18.4.3 Assumptions

LAMBDA_EXECUTION_ROLE has the required permissions in AWS to execute the lambda's required tasks.

18.4.4 Access Routine Semantics

RetrieveDocs(siteId: sID):

- output: $[D_k \in DocumentDatabase \wedge D_k.siteId == sID], \forall k \in \mathbb{Z}$
- exception: ExternalServiceFailure

CreateDoc(s3Link: sL, userID: uID, siteID: sID, parentDocumentID: pID, expiryDate: eD, requiresAck: rA):

- transition: $DocumentDatabase \rightarrow DocumentDatabase \cup \{D_{new}\}$, where $D_{new}.userId = uID \wedge D_{new}.siteId = sID \wedge D_{new}.createdDateTime = getDateTime() \wedge D_{new}.expiryDate = eD \wedge D_{new}.requiresAck = rA \wedge D_{new}.s3Link = sL$
- output: $map[string \rightarrow any]$: The documentId of the created document
- exception: ExternalServiceFailure, ValidationError

EditDoc(documentId: dID, userId: uID, s3Link: sL):

- transition: $DocumentDatabase \rightarrow DocumentDatabase - \{D_{old} | D_{old} \in DocumentDatabase \wedge D_{old} == dID\}$
 $DocumentDatabase \rightarrow DocumentDatabase \cup \{D_{new}\}$ where $D_{new} = copy(D_{old}) \wedge D_{new}.s3Link = sL \wedge D_{new}.userId = uID \wedge D_{old}.lastEditedDateTime = getDateTime()$
and $copy(D)$ is a predicate indicating the deep copy of document D .
- exception: ExternalServiceFailure, ValidationError

DeleteDoc(documentId: dID):

- transition: $DocumentDatabase \rightarrow DocumentDatabase - \{D_{old} | D_{old} \in DocumentDatabase \wedge D_{old}.documentId == dID\}$
- exception: ExternalServiceFailure, ValidationError

18.4.5 Local Functions

Name	In	Out	Description
getTime	-	string	Returns the current date and time in ISO8601 string format

References

- Carlo Ghezzi, Mehdi Jazayeri, and Dino Mandrioli. *Fundamentals of Software Engineering*. Prentice Hall, Upper Saddle River, NJ, USA, 2nd edition, 2003.
- Daniel M. Hoffman and Paul A. Strooper. *Software Design, Automated Testing, and Maintenance: A Practical Approach*. International Thomson Computer Press, New York, NY, USA, 1995. URL <http://citeseer.ist.psu.edu/428727.html>.

19 Appendix

19.1 Symbolic Parameters

TIMEOUT = 5

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Problem Analysis and Design.

The purpose of reflection questions is to give you a chance to assess your own learning and that of your group as a whole, and to find ways to improve in the future. Reflection is an important part of the learning process. Reflection is also an essential component of a successful software development process.

Reflections are most interesting and useful when they're honest, even if the stories they tell are imperfect. You will be marked based on your depth of thought and analysis, and not based on the content of the reflections themselves. Thus, for full marks we encourage you to answer openly and honestly and to avoid simply writing "what you think the evaluator wants to hear."

Please answer the following questions. Some questions can be answered on the team level, but where appropriate, each team member should write their own response:

The information in this section will be used to evaluate the team members on the graduate attribute of Problem Analysis and Design.

1. What went well while writing this deliverable?

In this deliverable, our team was able to delegate tasks and manage our time more effectively. The stakeholder was engaged to show progress on the prototype and obtain feedback that influenced the design.

2. What pain points did you experience during this deliverable, and how did you resolve them?

One pain point the team experienced was determining the specific technologies which would be the most appropriate for the clients problem. One example was deciding the AWS modules to use, such as choosing an EC2 instance or using AWS lambda. This was resolved through discussion of what the advantages of one tool would be over another. For this particular problem, it was decided to use AWS lambda because it makes it easy to scale to 0 instances when not in use, and the application is only expected to be used sporadically during working hours, so costs can be saved by being able to scale to zero when not in use.

3. Which of your design decisions stemmed from speaking to your client(s) or a proxy (e.g. your peers, stakeholders, potential users)? For those that were not, why, and where did they come from?

Through meeting with the stakeholders, the design decisions about the location verification module, creating accounts, file system module, and the function compute module were decided. These arose by discussing what these modules would be capable of doing, and how they would satisfy specific requirements identified in the SRS.

4. While creating the design doc, what parts of your other documents (e.g. requirements, hazard analysis, etc), if any, needed to be changed, and why?

One part of the SRS document which was required to be changed was the original SharePoint integration requirements. This was deemed to be too difficult to do and would open up vulnerabilities for the stakeholder, and as such was removed from the initial requirements identified in the SRS.

5. What are the limitations of your solution? Put another way, given unlimited resources, what could you do to make the project better? (LO_ProbSolutions)

The main limitations of the current solution is that it doesn't stream the location of users in real time. It currently takes two points in time, (entering and exiting time) into account. This limitation provides reduced visibility on the actions of the user at each site, which limits its sophistication but increases its simplicity. Given more time, the project could be further improved to increase the robustness of the verification functionality to automatically collect more information from the user and consider more complicated edge cases that can arise.

6. Give a brief overview of other design solutions you considered. What are the benefits and tradeoffs of those other designs compared with the chosen design? From all the potential options, why did you select the documented design? (LO_Explores)

As discussed above, one design solution that was considered was an Amazon EC2 instance due to its widespread use and support which would be very maintainable after the completion date of the project. However, the benefit of an AWS Lambda instance with scaling to 0 was determined to be the best choice due to the cost saving which it is able to provide,

Another design decision which was explored was the ability to use presigned URLs that would permit larger uploads than uploading through API Gateway.