AMELIA V3I

Day Three

Presented by
IPsoft Global Training and Development

Recap – Day Two

- Service Prefix - response:
- Putting It Together: Connecting Intents and BPNs
- Additional BPN Functionality and Task Types
- Present Task
- Request Task
- Amelia Trainer: Entities
- Putting It Together: Entities and ask for the slot
- Service Prefixes - slot: and ctx:
- Datum Types
- Service Prefix – bpn:

Agenda – Day Three

- BPN Version Control
- Amelia Trainer: Additional Tools
- Role Entity
- Spanless Entity
- Intelligent Arbitration
- Semantic Memory: Intent FAQs, Semantic Memory FAQs, Manual
- BPN Script Tasks and Libraries
- Additional Scripting and Services
- Formatting Entities with quantityService and qty:
- Clarifying Question Asker (CQA)
- Tabular Data
- Q&A

Learning Objectives

By the end of this course, you will be able to:

Train Amelia to conduct a business process and answer organization-specific questions using:

- End user intent recognition (intents)
- Customer answer capture (entities)
- Process design (BPNs)
- Frequently Asked Questions (intent FAQs)
- Workflow decision-making based on end user responses (prefixes)
- Integration with API (integration framework)
- Custom UI elements such as widgets and chat notes (custom UI/script tasks)

Role Entity

Role Entity

Two or more entities of the **same datum type**

- e.g., checkInDate and checkOutDate -> reserveARoom
- e.g., countryTo and countryFrom -> moneyTransfer

- Particularly if those values could be provided as part of an **intent**/other utterance **prior to the ask for the slot tasks**
  - e.g., I want to reserve a room from 05/18 to 05/22.
  - e.g., Need to transfer $10,000 from Japan to Ireland

Role Entity

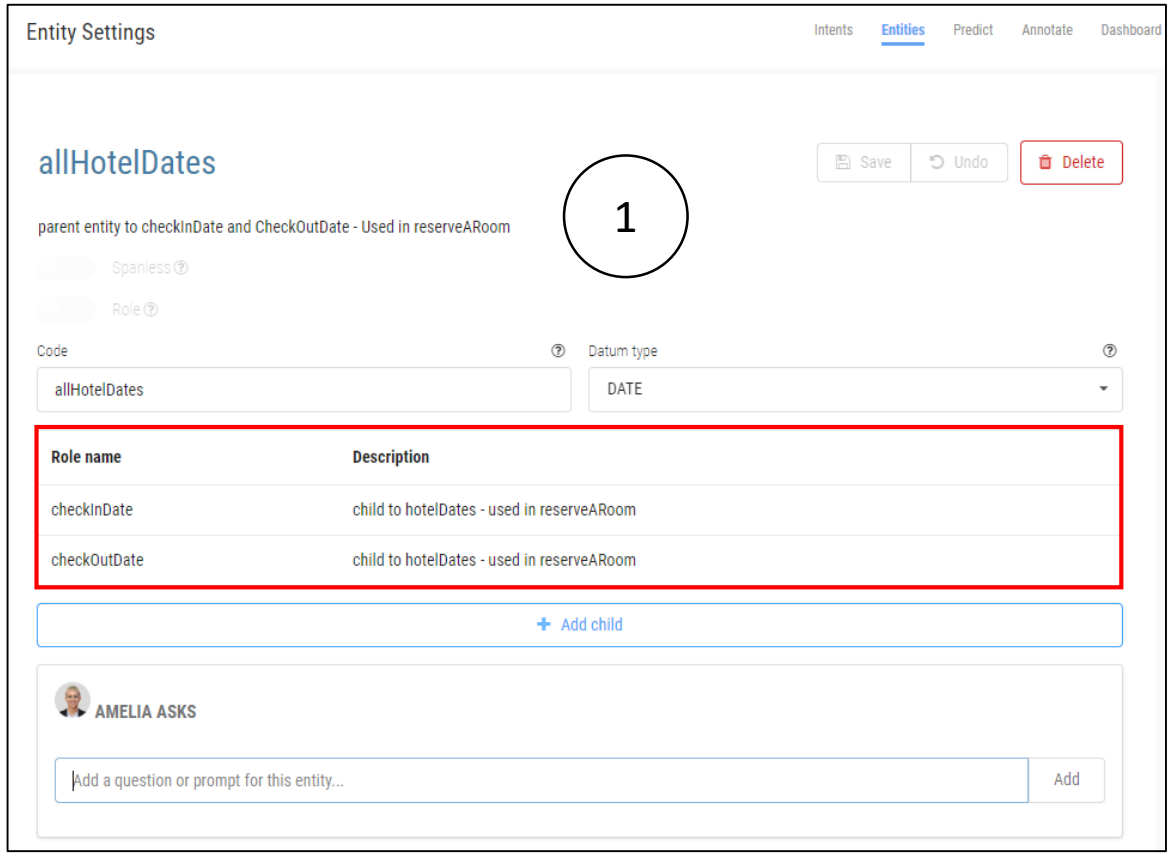
- Defines a relationship between these entities using a parent entity
- Allows Amelia to learn how different language, sentence constructs, and placement of entities within those constructs determine which value provided by the end user fills which entity slot

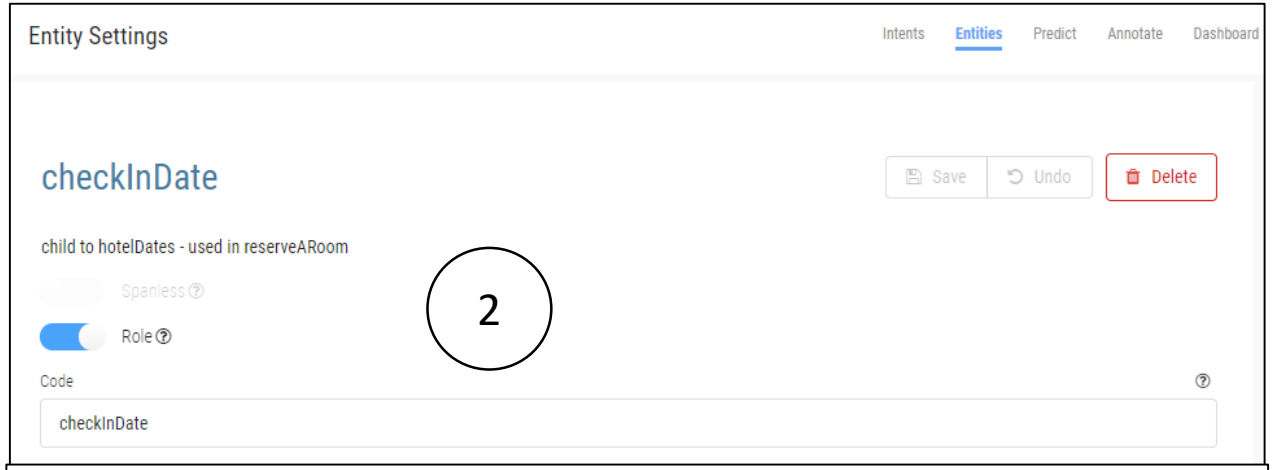
Group Activity: Role Entities

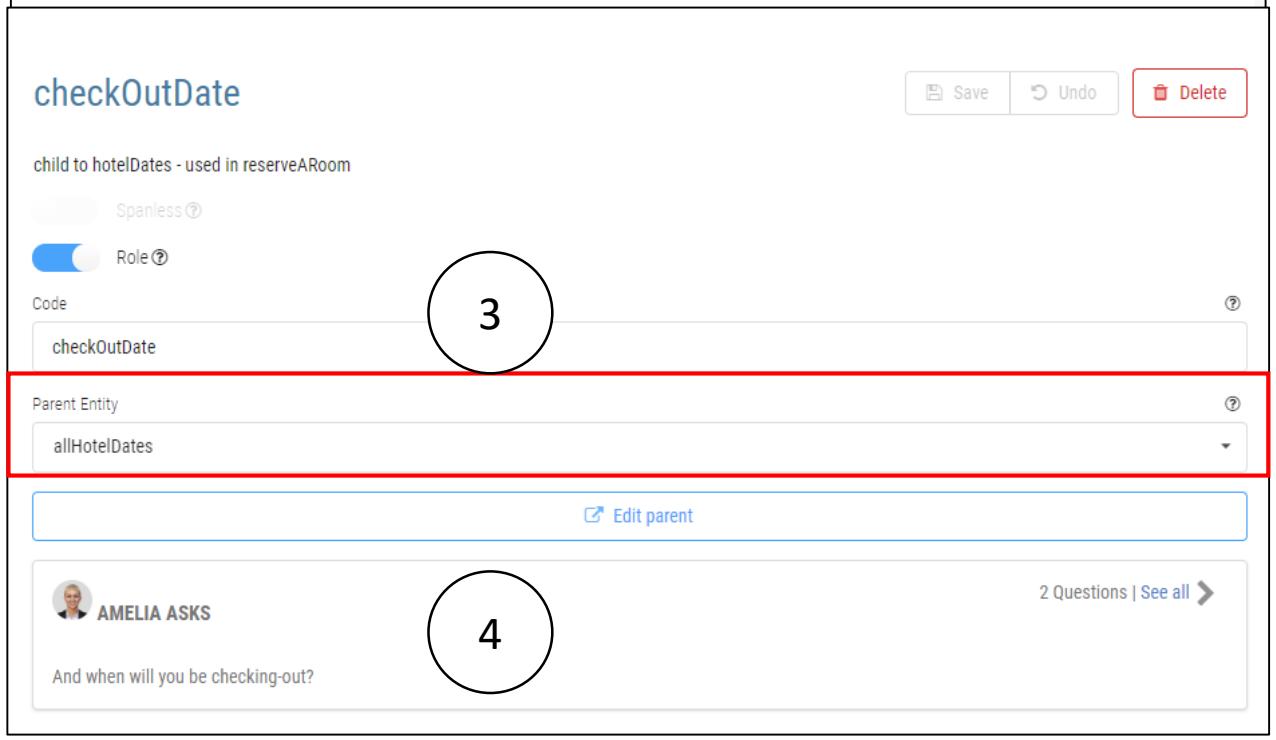
1. Admin view: click on Amelia Trainer->Entities->Add Entities tab.
2. Add “1YourInitialsParent” for both Name/Code. Datum Type: Integer. Description: Parent. Click Add.
3. Admin view: click on Amelia Trainer->Entities->Add Entities tab.
4. Add “yourInitialsBedrooms” for both Name/Code. Datum Type: Integer. Click Add.
5. In this child entity, click Role Entity and select the parent - 1YourInitialsParent.
6. Add a question like “How many bedrooms would you like?”
7. Repeat steps 3-6 for “yourInitialsBathrooms” with the question “How many bathrooms do you want?”
8. In V3DayThreeExerciseFiles, rename exercise file “roleEntityBedroomBathroomDataSet” to include your initials.
9. Import into annotation framework as plain text, and annotate integers as either yourInitialsBedrooms or yourInitialsBathrooms. Save data set as yourInitialsRoleEntity.
10. Click on Train. Select Type: Role Entity Classifier and name your entity model.
11. Select 1YourInitialsParent from the Parent Entity dropdown list and the training corpus (the data set you just annotated). Leave evaluation as default. Select Algorithm and choose Meta Classifier.
12. Review the results in Dashboard and deploy. Test in Predict.

Create Role Classifier Entities

1. Create a parent entity
2. Create at least two child entities with same datum type
3. Child entities assigned a parent entity => allHotelDates
4. Include question for each child entity – Amelia Asks

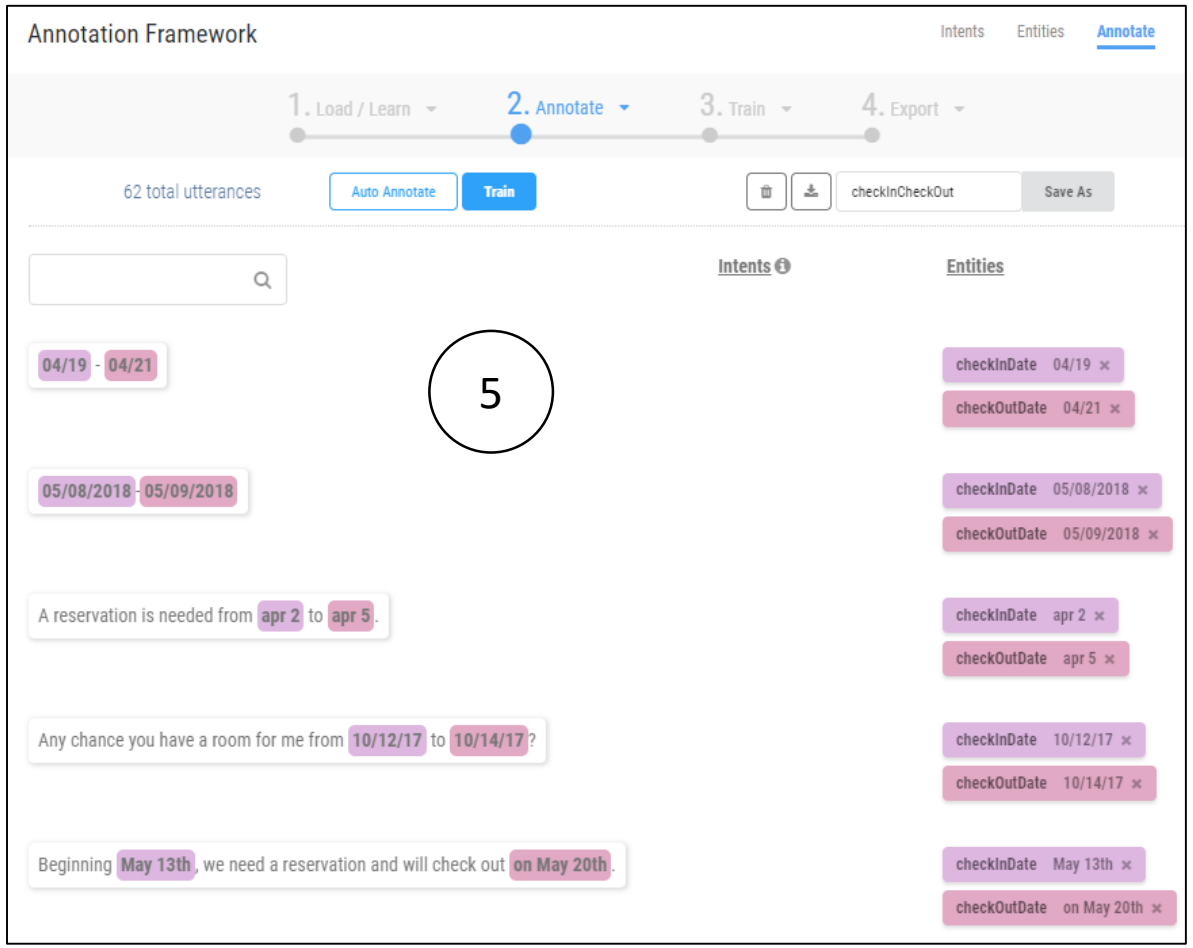




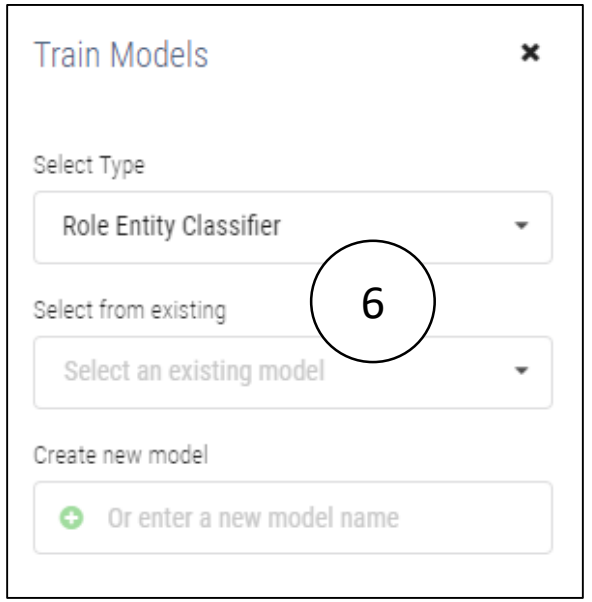


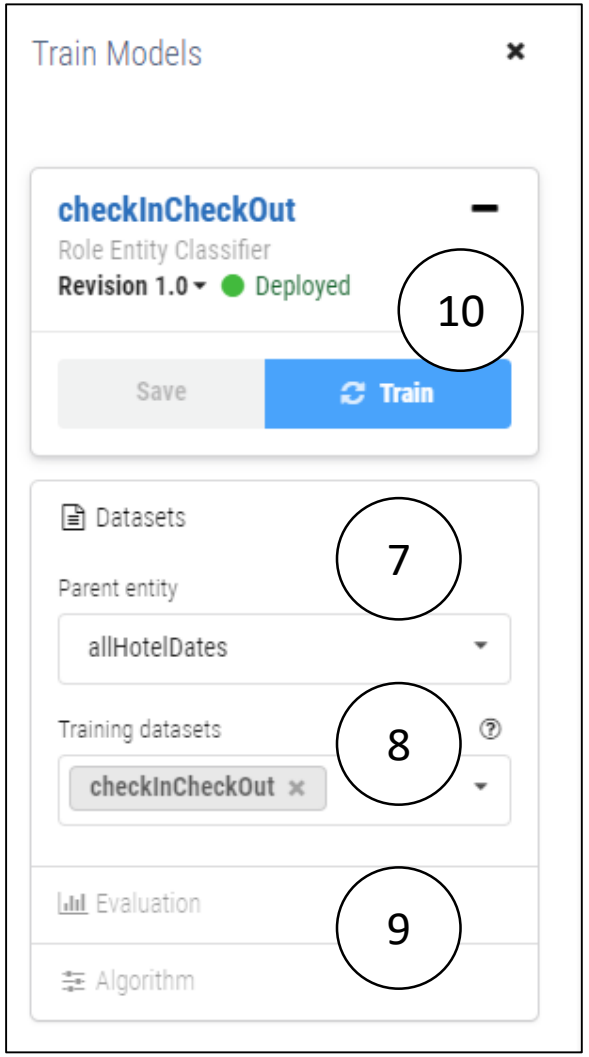
Annotate and Train Role Classifier Entities

5. Annotate utterances with examples of one/both entities



6. In Train, select Entity Tagger Type and name the entity
7. Select the parent entity
8. Select the training corpus
9. Modify other settings as needed
10. Train the model





Role Classifier Model Review

11. Review the Dashboard stats and deploy model

11

Model Name	Revision	Created By	Utterances	Algorithm	Actions	Logs	Stats	Status
countryToCountryFrom	v.1.0	Brian Kuchta	53	Logistic Regression				deployed

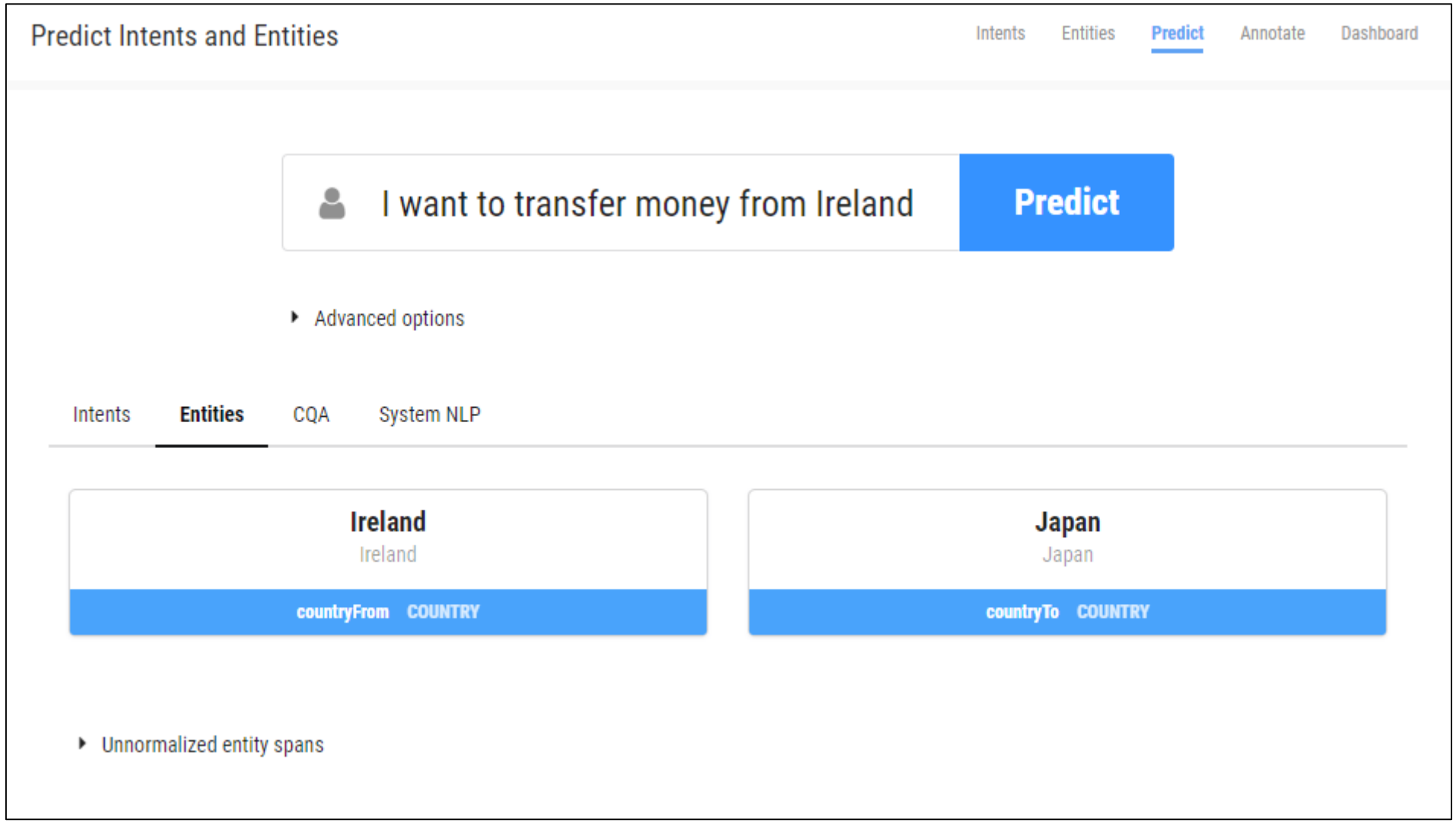
1 Classifier Models, Page 1 of 1

countryToCountryFrom_5-fold_cv: Statistics

	Correct	Gold	System	Precision	Recall	F1
countryFrom	39	40	40	97.50	97.50	97.50
countryTo	35	36	36	97.22	97.22	97.22
Overall	74	76	76	97.37	97.37	97.37

countryToCountryFrom_5-fold_cv_1	7/19/2018, 5:30:33 PM
countryToCountryFrom_5-fold_cv_0	7/19/2018, 5:30:32 PM

Role Entity Predict



Role Classifier Entities

Parent entity span is extracted from end user utterance, entity classifier applied on span to determine which child receives which extracted value

End User: Coming to your city soon for vacation and need to reserve a single king room starting on June 5th and leaving on 06/07.

Amelia: I'm happy to help you make a reservation.
Amelia: Please confirm that your check-in date is 2018-06-05.

End User: yes

Amelia: Please confirm that your check-out date is 2018-06-07.

End User: yes

Example:
Coming to your city soon for vacation and need to reserve a single king room starting on June 5th and leaving on 06/07.

Extracted parent span:
starting on June 5th and leaving on 06/07

Extracted role entities:
checkInDate = June 5th
checkOutDate = 06/07

Normalized date datum entities:
checkInDate = 2018-06-05
checkOutDate = 2018-06-07

Activity

Role Entity Workflow:

1. Determine the parent entity and child entities (Amelia skill)
2. Create the parent entity (Entities)
   - Name/Code
   - Datum Type
   - Description
3. Create the child entities (Entities)
   - Name/Code
   - Datum Type (must be the same as the parent)
   - Description
   - Role entity and parent
   - Include question/questions
4. Annotate utterances that include one or both child entities (Annotation Framework)
5. Train entity model for a role entity (Annotation Framework)
6. Review model statistics (Dashboard)
7. Deploy model (Dashboard)
8. Test in Predict

Timing: 30 minutes

Validation: Mind View Testing

Role Entity

- Design a role entity for your Amelia skill.
- Create the parent entity and the children entities.
- Develop 20+ grammatically varied sentences that include one or both children entities within the context of the sentences.
- Import into the annotation framework and annotate the role entities (start with auto annotation to help spot the entities in the sentences).
- Create a role entity model in the annotation framework.
- Create a new BPN and import bedroomBathroomRoleEntity BPN from the exercise folder.
- Deploy the model and test in mind view.

Spanless Entity

Spanless Entity

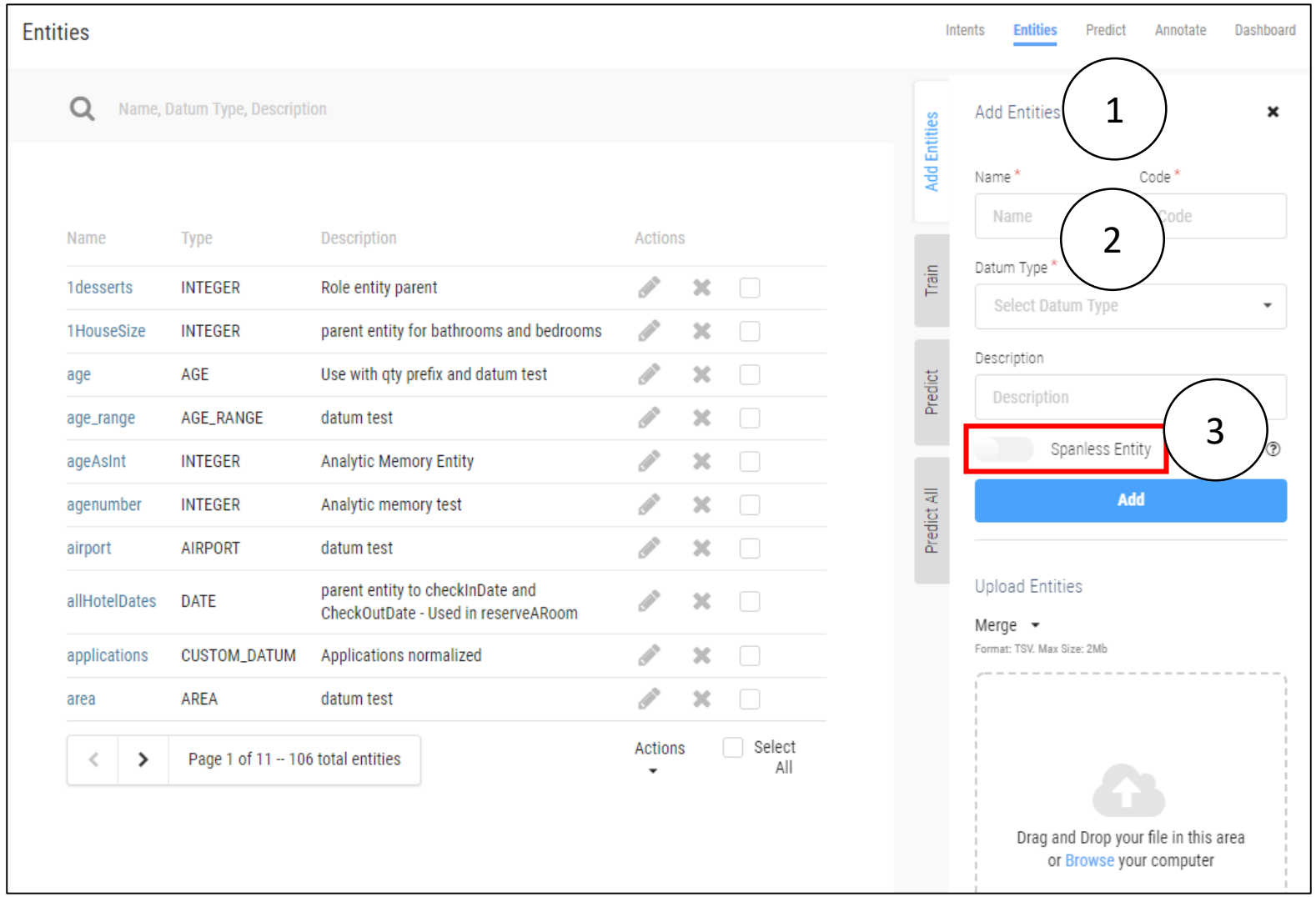
- Needs to be inferred from whole phrase/ sentence
  - Doesn’t have well-defined “span”
- Typically, few possible values
  - These values become labels or “classes”
- Sentence-level classification
- Output value => label
- Trained independently from other entities
- Uses sentence-level algorithms

```
    There's smoke coming from the fireplace.
    There's smoking coming from the house
    think there 's one on the dining room table
    Unless we take a break and you go outside, there's no smoking allowed in the theater
    Very unhealthy to be doing that kind of smoking activity in a closed space
    want you to put the sofa in the living room
    We do not like when smoke comes into the house in the fireplace
    Where there is smoke there is often fire
    You must be 21 to purchase cigarettes
nonsmoking  Allergic to cigarettes so non-smoking
nonsmoking  Always a nonsmoking room for me
nonsmoking  Because I have breathing issues, I must get non-smoking
nonsmoking  Can't stand smoking of any kind
nonsmoking  Cigarettes are disgusting and I can't stand the smell
nonsmoking  Cigars and cigarettes are disgusting.
nonsmoking  Definitely nonsmoking
nonsmoking  Every time I stay with you I get a non-smoking room thanks
nonsmoking  Forget the smoking room as I hate cigar and cigarette smoke smell.
smoking Because I love my cigarettes I have to have a smoking room
smoking Can't live without my cigars
smoking cigars are a favorite so need a smoking room
smoking Cigars are my calling and need to be able to have my smokes
smoking Do let me have a booking where I can smoke as much as I want
smoking Each time I stay at a hotel, I get a room where I can have my smokes if possible.
smoking For me, it's all about the smoking room
smoking Get room where smoking is allowed
smoking give me a smoking allowed room
```

Group Activity: Spanless Entity

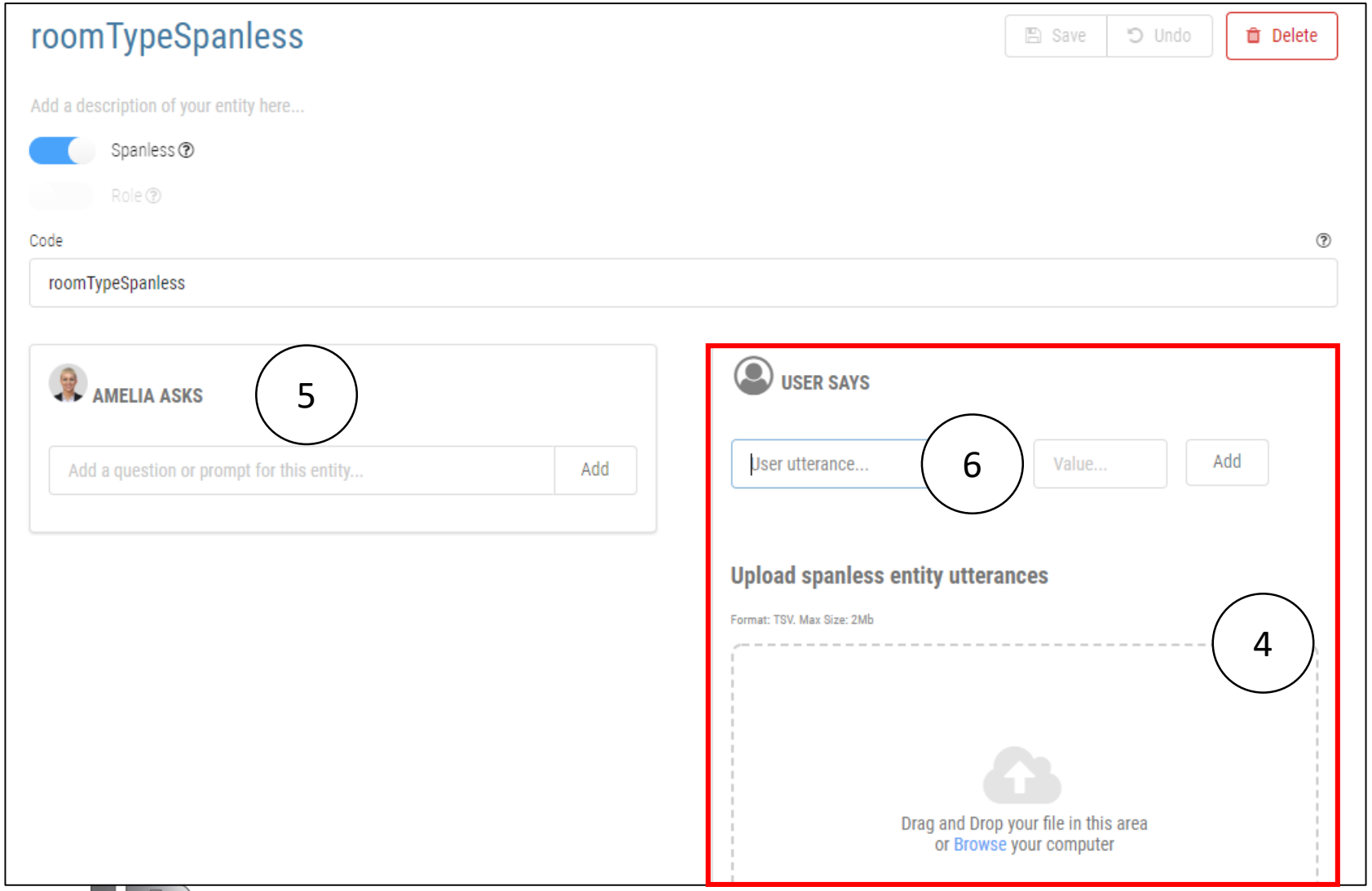
1. Admin view: click on Amelia Trainer->Entities-> Add Entities.
2. Add “yourInitialsHungryNotHungry” for both Name and Code.
3. Select Datum Type: Text. Add a description: My first spanless entity. Click Add.
4. Find your entity in the entity list and click on the name.
5. Under Amelia Asks, type “Describe how you are feeling.” and click Add.
6. Upload spanlessHungryNotHungryDataSet under User Says. Click Train.
7. Select Spanless Entity Classifier Type and name model.
8. Choose the spanless entity from the dropdown list. Turn off Add fallback examples.
9. Leave Evaluation as default. In Algorithm, change to Meta Classifier. Train the model.
10. Click on Amelia Trainer->Dashboard and review the stats and deploy.
11. Test in Amelia Trainer->Predict.

Spanless Entities – New Entity



1. Click Add Entities and provide Name/Code
2. Datum Type: always Text
3. Click the Spanless Entity button and Add

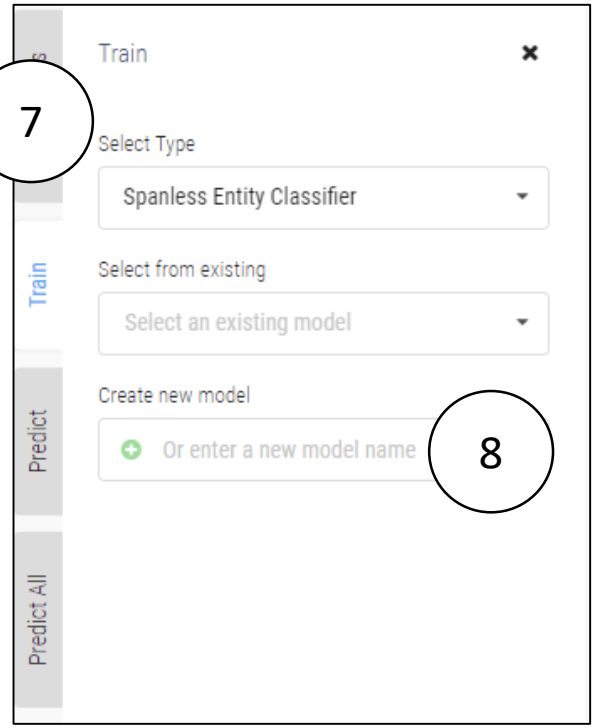
Spanless Entities – Classes and Example Utterances

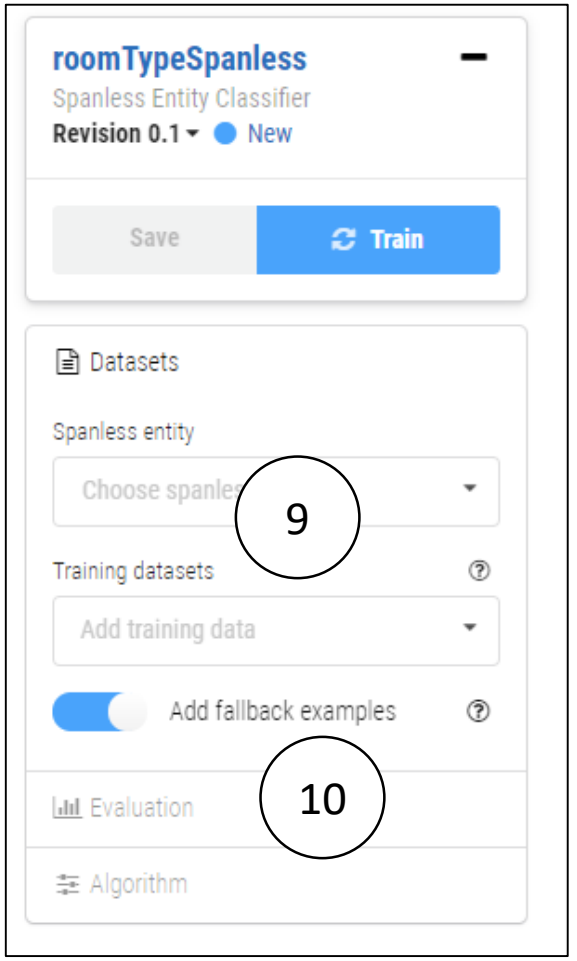


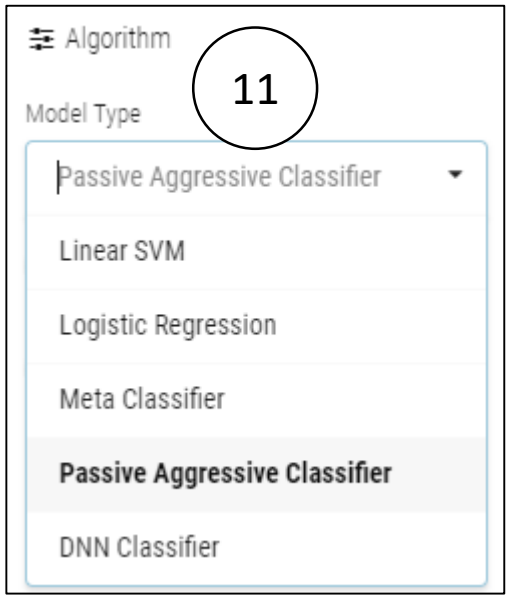
4. Import annotated utterances
   - Label[TAB]Utterance
   - [TAB]NegativeUtterance
5. Add question(s) that should be asked by Amelia
6. Include additional example utterances with label

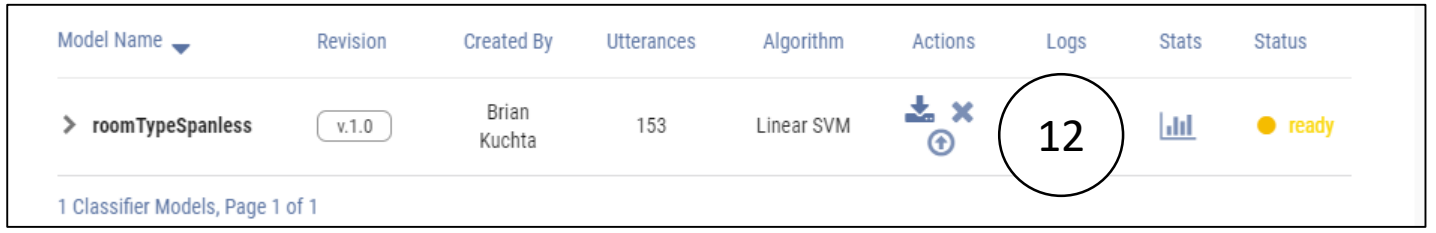
Spanless Entities – Train Entity Model

7. Click Train and set Select Type to Spanless Entity Classifier
8. Name the entity model
9. Under Datasets, select the Spanless entity from the drop down list and turn off add fallback examples
10. Leave Evaluation as default settings
11. Set the appropriate algorithm (Meta Classifier recommended)
12. Review the Dashboard stats and Deploy
13. Test in Amelia Trainer->Predict









Create Spanless Validation Data Set

Using a validation data set for intents/entities = best practice

- Spanless entities uniquely include training data in entity, not in annotation framework

Create validation data set for spanless entity specific annotation:
entityName=spanlessLabel1[TAB]utterance for first class
entityName=spanlessLabel2[TAB]utterance for second class
entityName=[TAB]utterance for negative class

```
roomType=single No need for more than one bed
roomType=double Prefer the queen beds over the kings
roomType=   love the queen of kings
```

Entity Training – Multiple Spanless Entities

- Spanless entity can be recognized from the intent utterance
- Can have multiple spanless entities in BPN

Example:

“What can I do for you today?”
- End user responds “ I want a nonsmoking room with a queen sized bed”
- Fills smokingNonSmoking spanless entity and roomType spanless entity

Activity

Spanless Entity Workflow:

1. Determine entity and the need for analysis of full end user utterance to determine specific value (Amelia skill)
2. Create the entity (Entities):
   - Name/Code
   - Datum Type
   - Description
   - Spanless entity
3. Add question(s) that should be asked to obtain entity in BPN (Entities)
4. Import class-annotated utterances and negatives into the entity itself (Entities)
5. Train entity model for a spanless entity (Annotation Framework)
6. Review model statistics (Dashboard)
7. Deploy model (Dashboard)
8. Test in Amelia Trainer->Predict

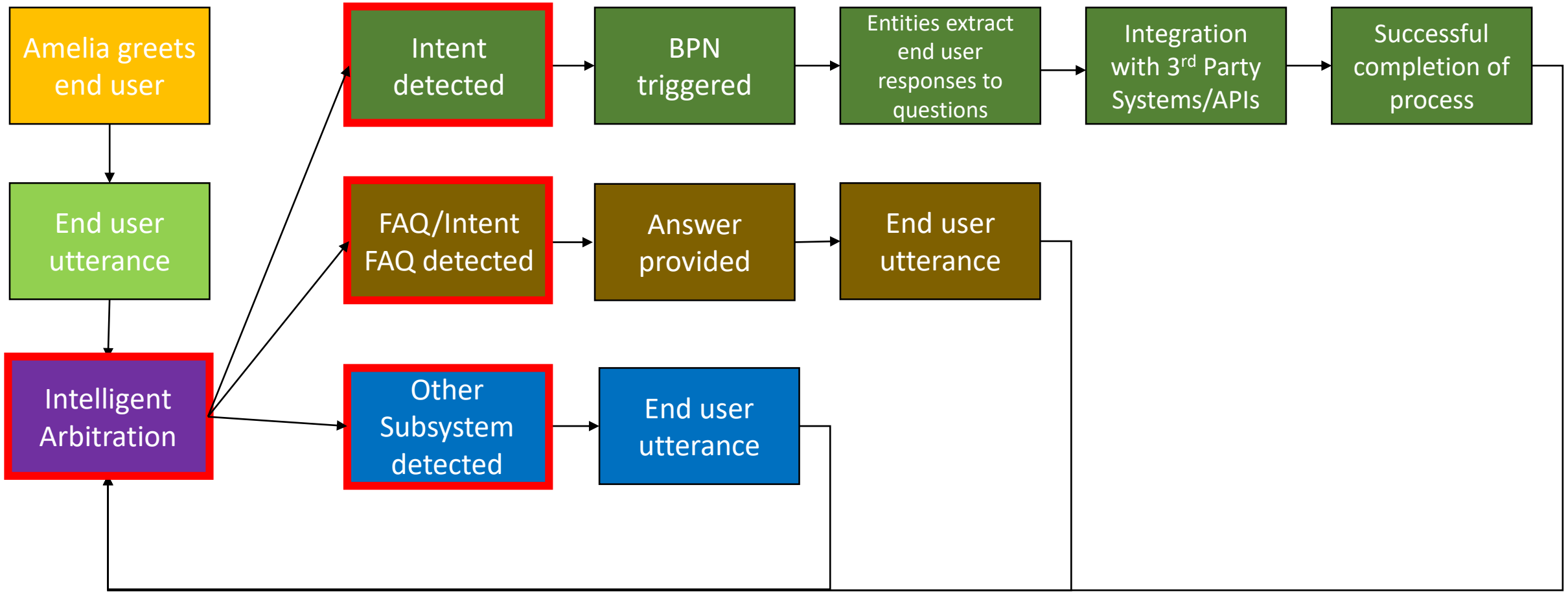
Spanless Entity

- Create a spanless entity for your Amelia skill.
- Develop 20+ grammatically varied sentences within specific classes for your entity.
- Develop 15+ contextually negative utterances.
- Import as annotated examples into the spanless entity.
- Train the spanless entity model.
- Review and deploy in Dashboard.
- Test in Amelia Trainer->Predict.

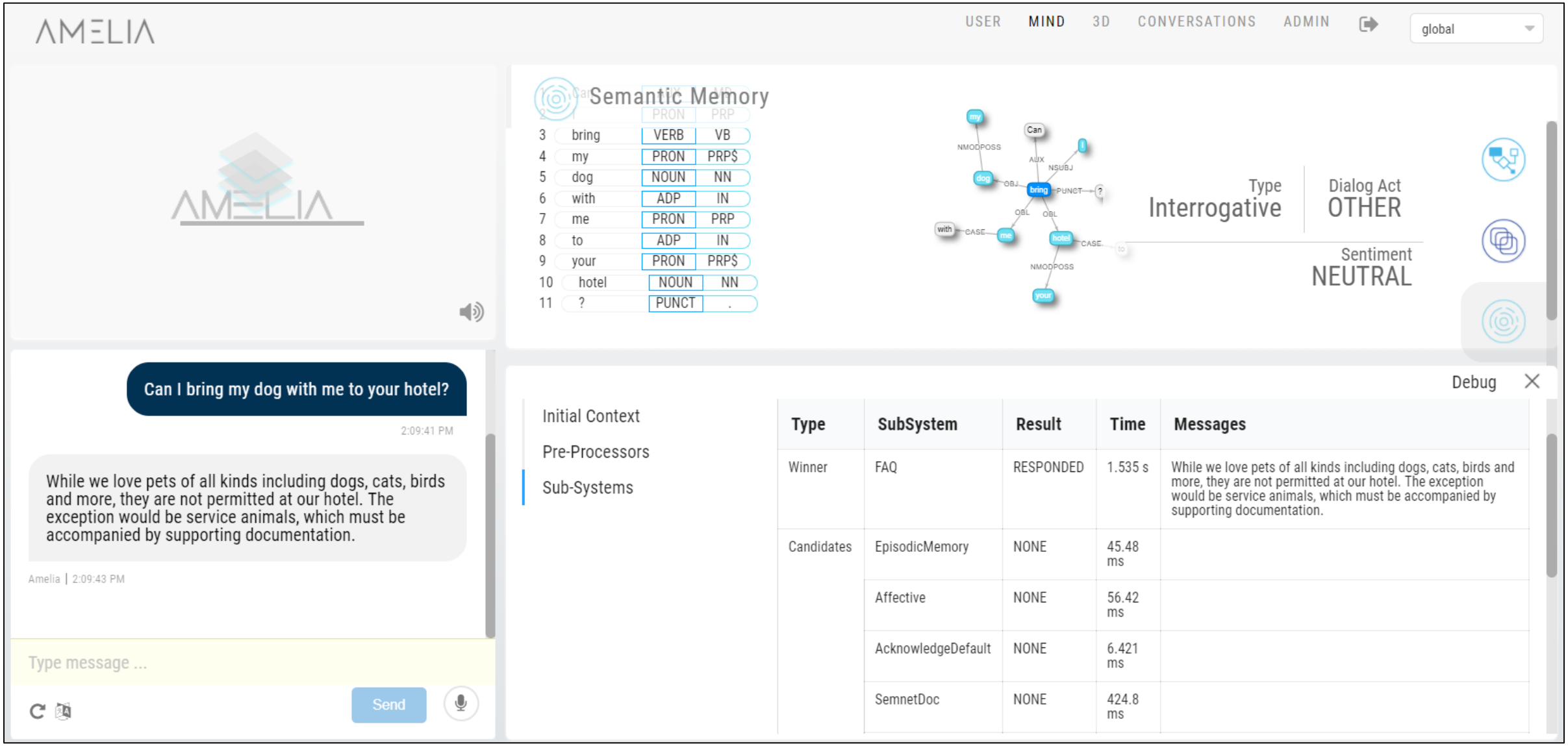
Timing: 20 minutes
Validation: Predict Tool

Intelligent Arbitration

General Amelia Conversation Flow

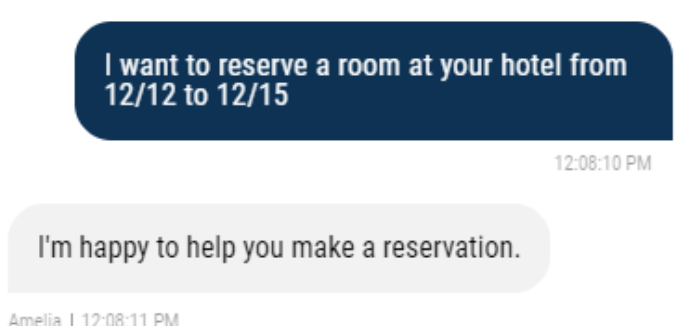


Mind: Intelligent Arbitration and Sub-Systems 3.5



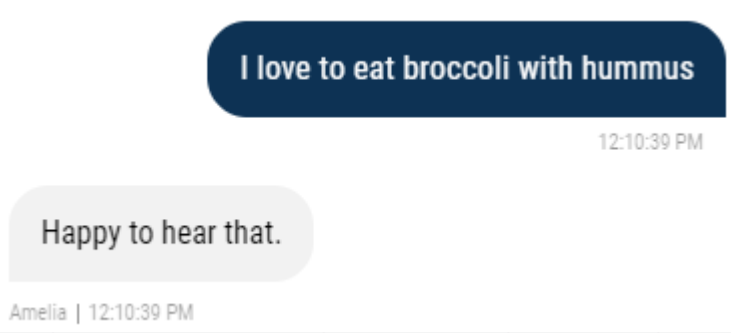
Intelligent Arbitration - Subsystems

- Subsystems work simultaneously to generate best response
  - Decide whether or not to answer
    - Based on functional boundaries
      - Subsystems do NOT interfere with one another



Type	SubSystem	Result	Time	Messages
Winner	Bpn	RESPONDED	338.9 ms	I'm happy to help you ma reservation.What's your
Candidates	SemnetDoc	NONE	5.748 ms	
	DontKnow	RESPONDED	15.95 ms	I don't know.
	FAQ	RESPONDED	424.7 ms	Guests at our hotel can rooms as early as 12:00 require an earlier check-

Type	SubSystem	Result	Time	Messages
Winner	FAQ	RESPONDED	1.430 s	While we love pets of all kinds in cats, birds and more, they are not our hotel. The exception would b animals, which must be accompa supporting documentation.
Candidates	SemnetDoc	NONE	100.9 ms	
	DontKnow	RESPONDED	11.70 ms	I'm not sure about that one.



Type	SubSystem	Result	Time	Messages
Winner	Affective	RESPONDED	67.37 ms	Happy to hear t
Candidates	SemnetDoc	NONE	21.89 ms	
	DontKnow	RESPONDED	16.03 ms	I don't know.
	FAQ	NONE	291.7 ms	
	AcknowledgeDefault	RESPONDED	29.18 ms	Good to know!

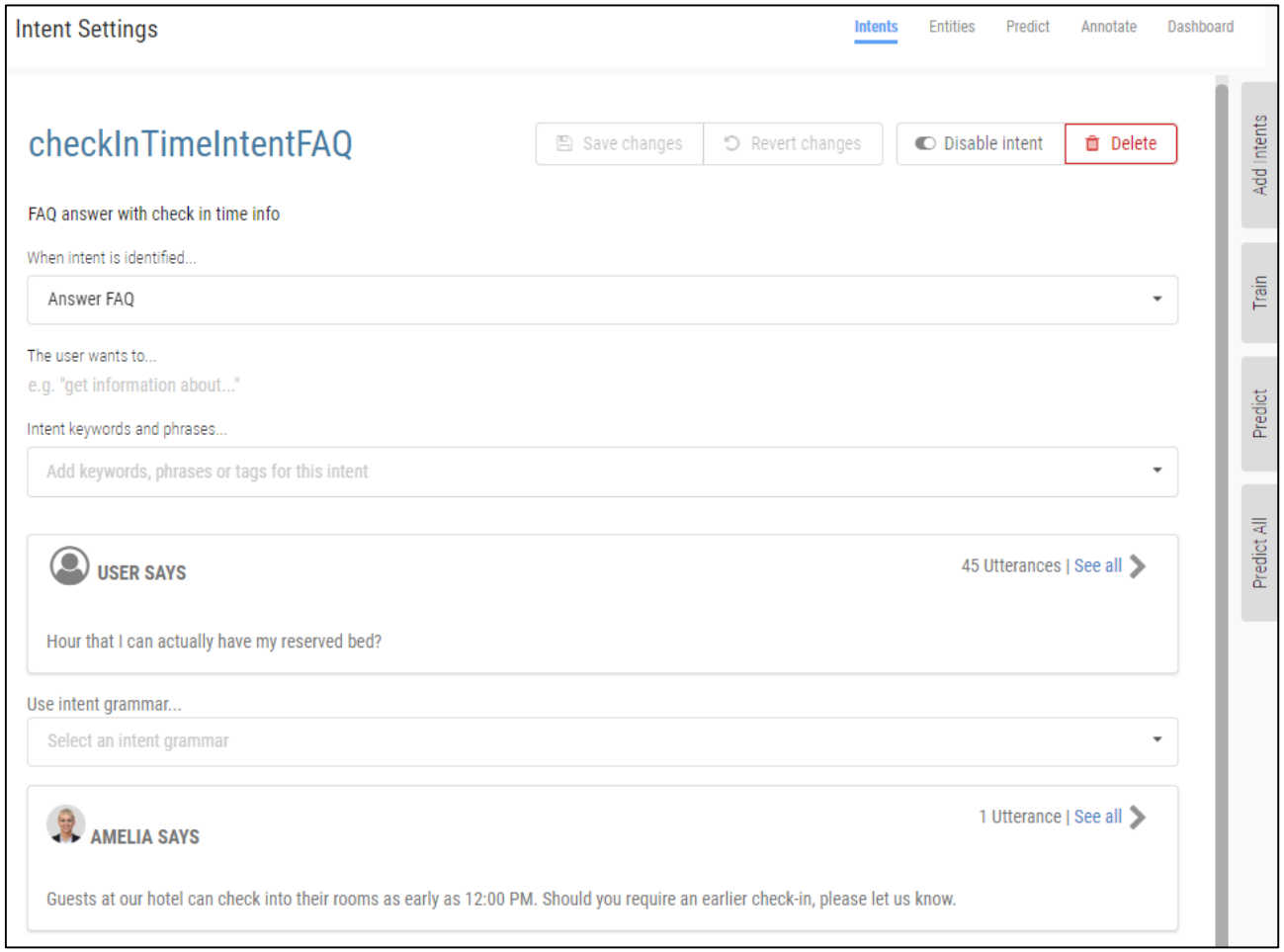
Intelligent Arbitration – Current Invocation Order 3.5

- Escalation
- BPN
- CQA
- Episodic
- Intent FAQ
- SemNet FAQ
- SemNet Responder
- SemNet Manual/Document
- Positive Response/Affective
- AIML (chit chat)
- Social Talk
- Acknowledge Default (e.g., Ok. Got it.)
- Don’t Know Default (e.g., To be honest, I really don’t know.)

Semantic Memory:
Intent FAQ, Semantic FAQ and Manual

Intent FAQ 3.5

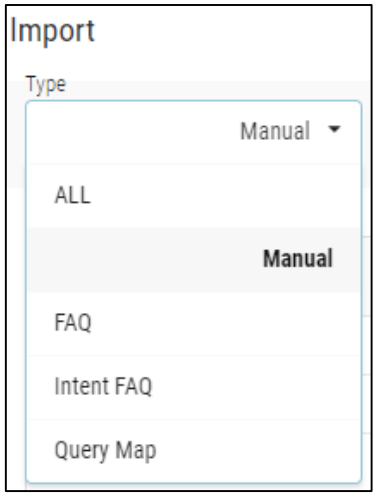
- Separate subsystem from BPN
- Intent created that only responds with answer to FAQ
- Included in intent model so statistical analysis of model training available

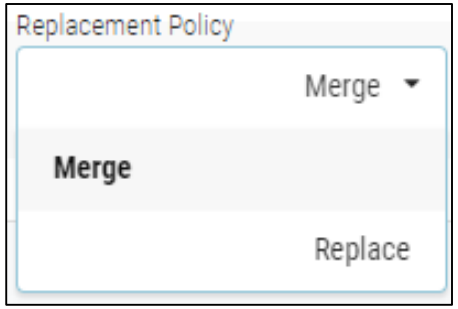


Import Intent FAQ

- Upload.xlsx file into Semantic Memory
  - First column label = Question/Answer
  - Second column = utterance
    - Include multiple, structurally-varied paraphrases of questions
  - Third column (1st row of each question set only) = intent FAQ name
    - Do NOT create Intent FAQ in Intents first
    - Will auto create based on Excel spreadsheet

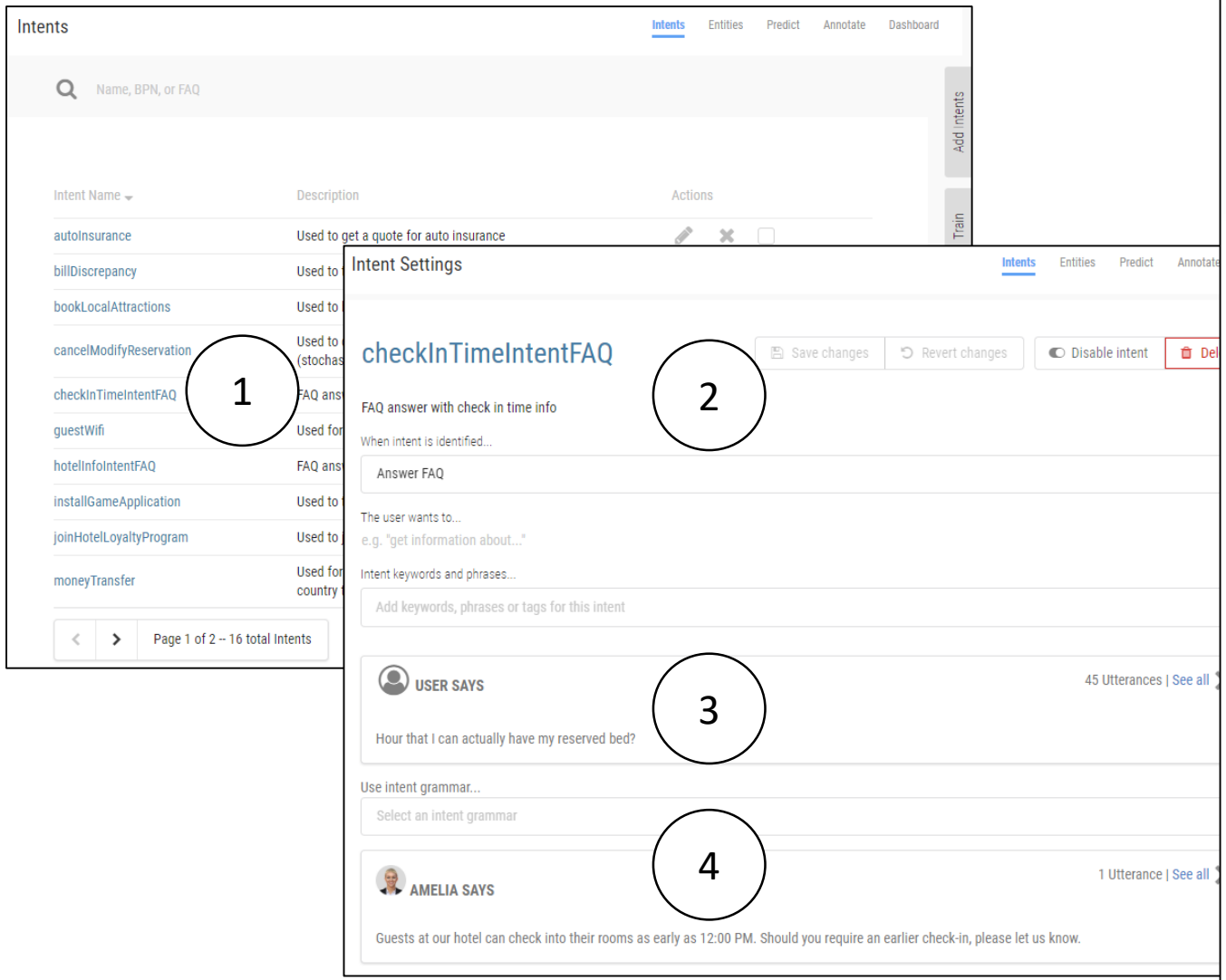
Question	Do you have wifi in your hotel rooms?	get_wifi
Question	Can I access the Internet in my room at your place?	
Question	Is there free wifi in the suites?	
Question	Wifi available in the rooms for all guests?	
Question	You have internet in rooms?	
Answer	Yes, all of our rooms have free Wi-Fi access.	
Answer	Of course all guest rooms have Internet access.	





Revise Intent FAQ and Intent Model

1. In Amelia Trainer-> Intents, click on the intent FAQ.
2. Include a description (optional).
3. Modify/Add example utterances as needed.
4. Modify FAQ answer(s) in the Amelia Says field.
5. Add/include intent FAQs in the intent model for the domain.



Semantic Memory FAQ Process

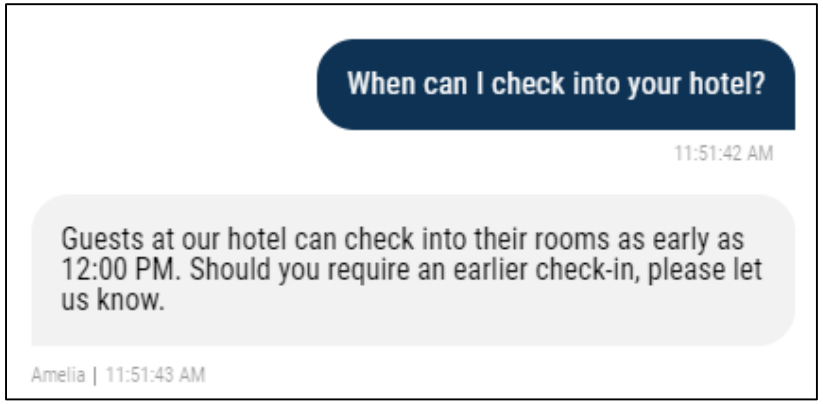
Another type of FAQ in V3
- Below BPN and Intent FAQ in intelligent arbitration hierarchy
- No model needed but also no statistics provided

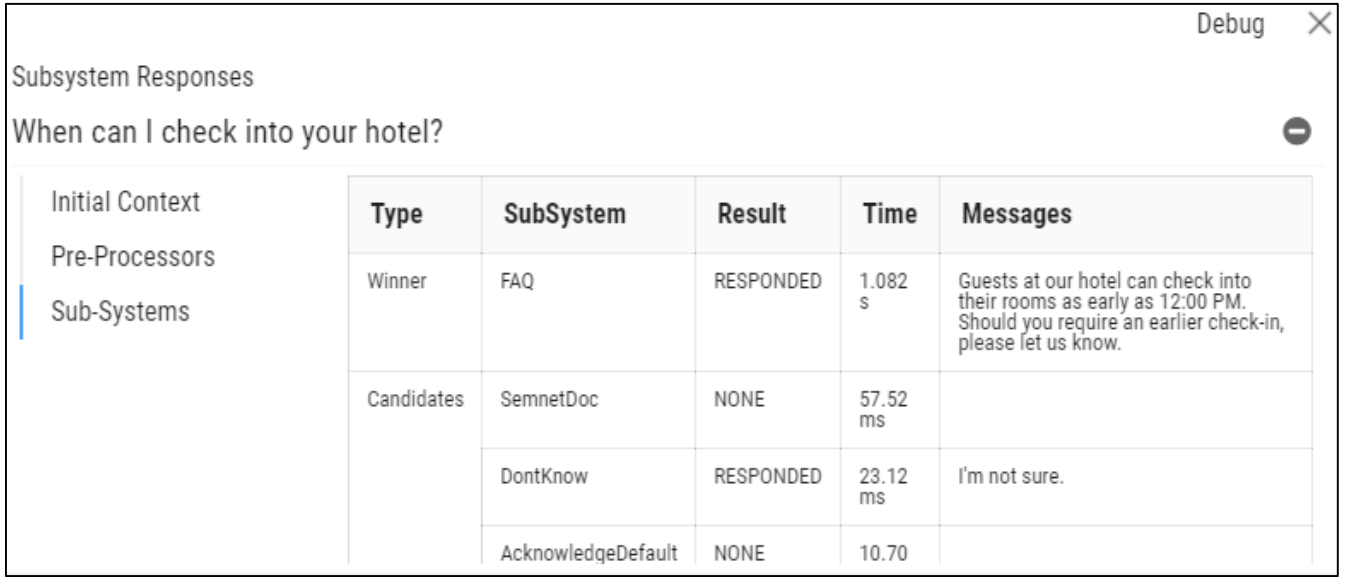
1. Analyzes end user utterance extended with Wordnet, synonyms, etc.
2. Top 10 possible responses obtained by querying Lucene Index of all question variation-answer pairs
3. Entailment narrows potential matches above a certain threshold
4. BiDAF analyzes remaining questions and answers => selecting best answer
   - Answers highly important as they are part of the analysis
   - Should have relevance/significance in relation to the question
5. Paragram vector similarity used to determine if answer is in-context by comparing the end user utterance and question corresponding to BiDAF's answer => Minimizes FAQ impact on other subsystems

Import Semantic Memory FAQs

- Upload.xlsx file
  - Format: First column label = Question/Answer
  - Second column = utterance
    - Include multiple, structurally-varied paraphrases of questions
  - FAQ algorithm takes into account question and answer so answer should be significant and reference the question

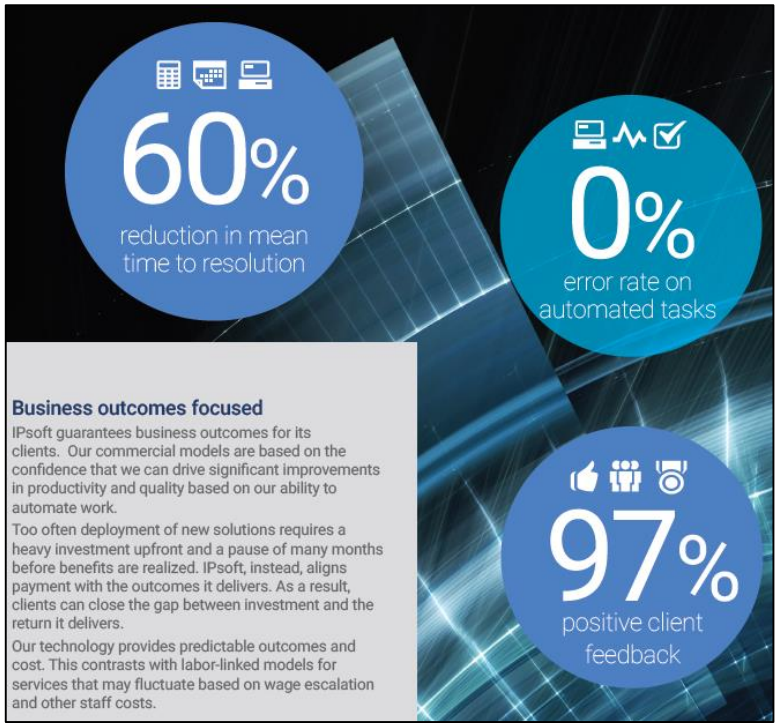
Question	i want more information on your hotel where can i go
Question	to discover more about your place, is there a site i can
Question	any hotel information online for me to read?
Question	if i want to learn more about your hotel, can i go some
Answer	You can find more information on our website.





Import Manual (PDF Document)

- Upload PDF of well-curated information
  - Bulleted lists, numbered lists, and tables useful
  - Amelia finds answers to queries based on document
  - Useful for general information for which an FAQ cannot be created



Why IPsoft?

IPsoft is the proven leader in enterprise AI, serving more than 500 of the world's leading brands directly as well as more than half of the world's largest IT services providers. We have almost 20 years of experience and expertise in the IT operations and automation market with IPcenter, and our award-winning virtual agent, Amelia, delivers cognitive, natural language and self-learning capabilities through an easy-to-use interface.

Manual Process

1. PDF parse understands how paragraphs are segmented
   - Including headings in text, lists and tables
2. User utterance analyzed -> obtains top results using Lucene Index
3. Determines if utterance entails very closely with any heading
4. Produces entire section as answer (i.e., list, paragraph, row, etc.)
5. Checks if there is high similarity between nouns in utterance and row/column headers of any output
6. BiDAF analyzes paragraphs followed by binary classifier to determine if question be answered given context (i.e., paragraphs, sections)

Activity

Question	Do you have wifi in your hotel rooms?	get_wifi
Question	Can I access the Internet in my room at your place?	
Question	Is there free wifi in the suites?	
Question	Wifi available in the rooms for all guests?	
Question	You have internet in rooms?	
Answer	Yes, all of our rooms have free Wi-Fi access.	
Answer	Of course all guest rooms have Internet access.	

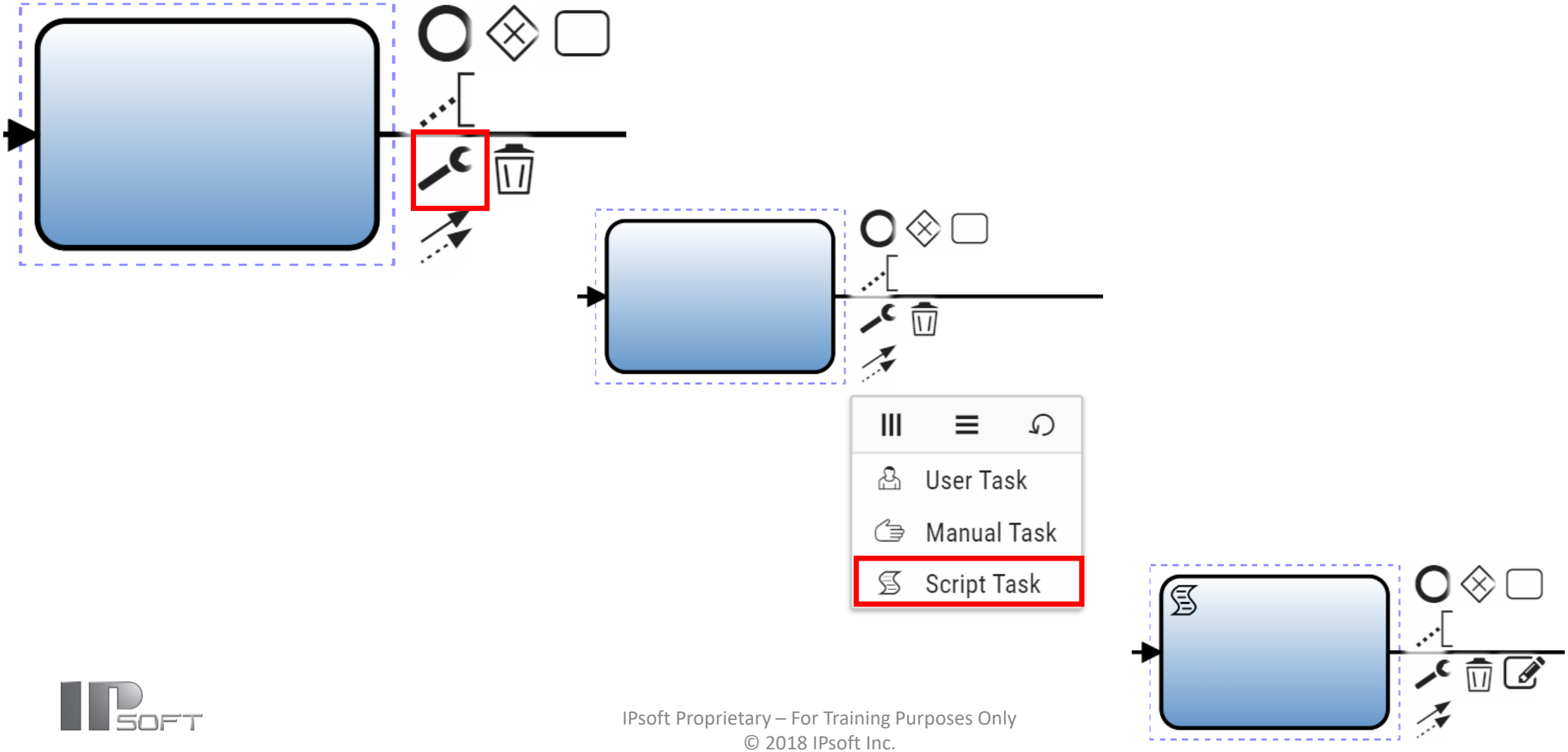
Timing: 20 minutes
Validation: Predict Tool

Intent FAQs

- Create an FAQ with at least 10 paraphrases of the Question and one Answer in Excel.
- Include a name for your intent FAQ in the third column, first row.
- Import the intent FAQ excel file into Semantic Memory.
- Develop 10+ in- and out-of-domain negative utterances.
- Import these into annotation framework and save.
- Create an intent model and include your intent FAQ utterances and negative utterances.
- Review the stats in dashboard.
- Test in Predict in Intents.

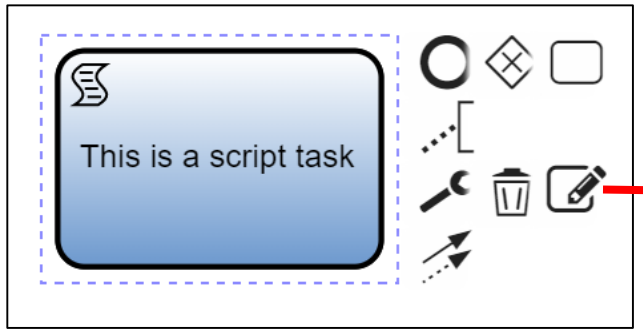
BPN Script Tasks and Libraries

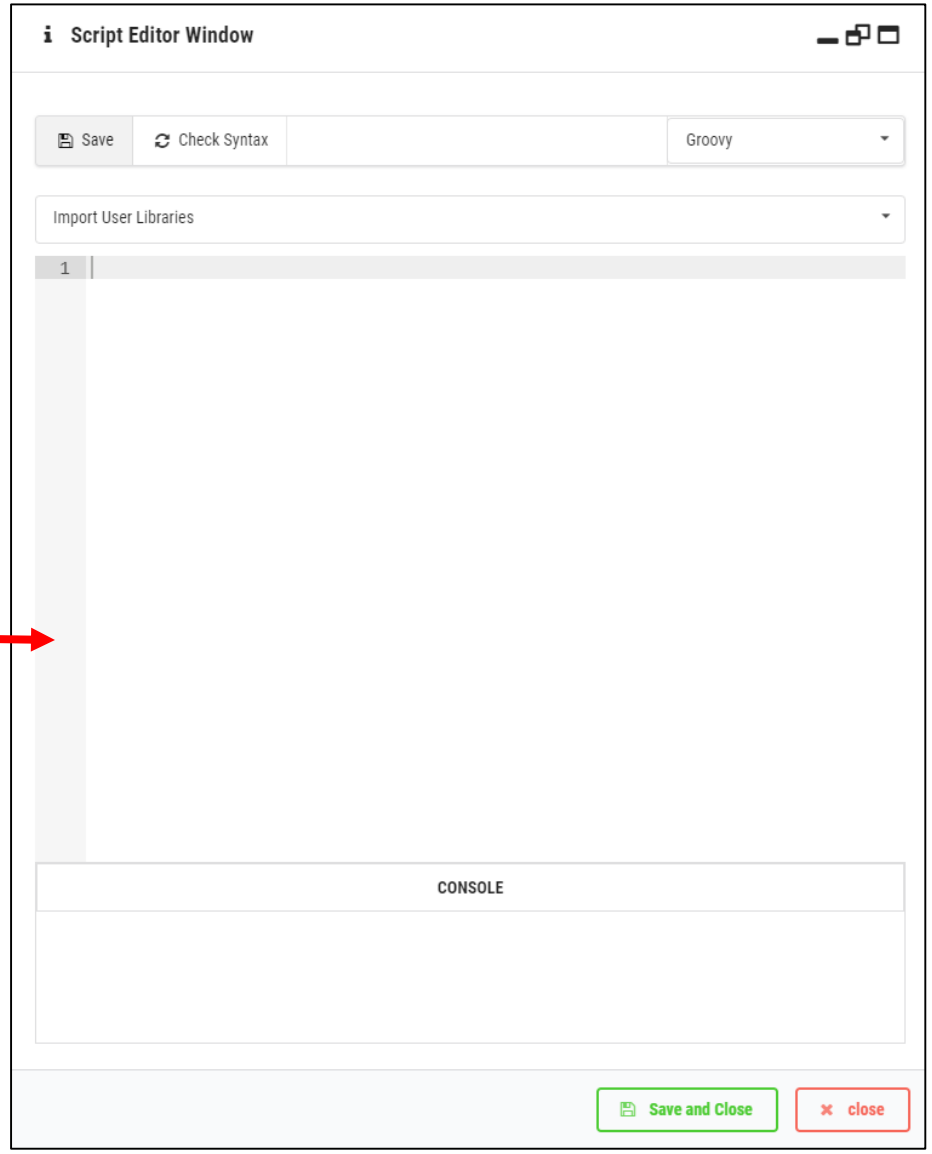
Task Tools



Script Editor

- Script editor opens
  - Select language
    - Groovy
  - Access any defined variables and functions
  - Script editor sandboxed
    - Import of whitelisted libraries and classes allowed





- Ctrl+space/Cmd+space activates autocomplete in script editor

Script Editor

```
movieTitle = execution.getVariable("movie")

switch (movieTitle.toLowerCase()) {

    case "scream":
        commentary = "That's a scary movie."
        break
    case "roger rabbit":
        commentary = "That's a funny movie."
        break
    case "raging bull":
        commentary = "That's a classic movie."
        break
    case "star wars":
        commentary = "That's a sci fi movie."
        break
    default:
        commentary = "I don't know that one."
        break
}

execution.setVariable ("commentary", commentary)
```

```
def breakfast = ["omelette", "pancakes", "breakfast burrito"]
def lunch = ["pizza", "salad", "hamburger"]
def dinner = ["tacos", "lobster", "steak"]


execution.setVariable('breakfast', breakfast)
execution.setVariable('lunch', lunch)
execution.setVariable('dinner', dinner)
```

Execution methods in script tasks for process scope variables:

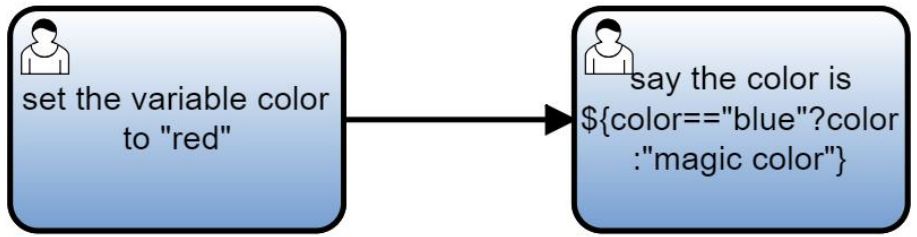
execution.getVariable('varName') -> gets process scope variable values into the script editor

execution.setVariable('varName', varValue) -> sets process scope variable holding value out to conversation

Operators for Process Scope Variables

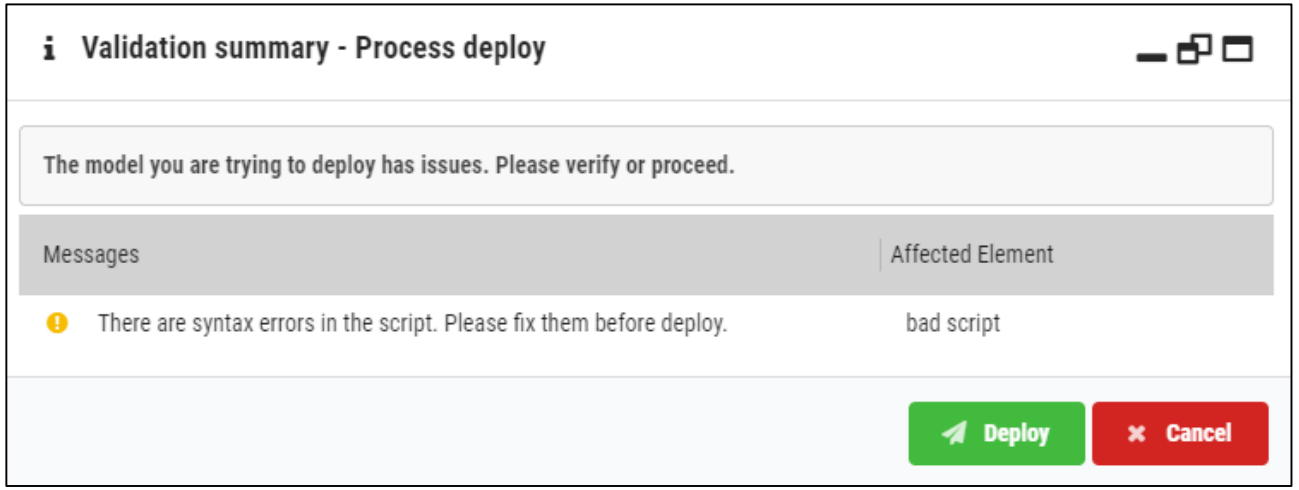
- equal to: == or eq
- greater than: > or gt
- less than: < or lt
- not equal to: != or ne
- less than or equal to: <= or le
- greater than or equal to: >= or ge
- and && or and
- or || or or
- not empty !empty varName
- ternary test? value1:value2

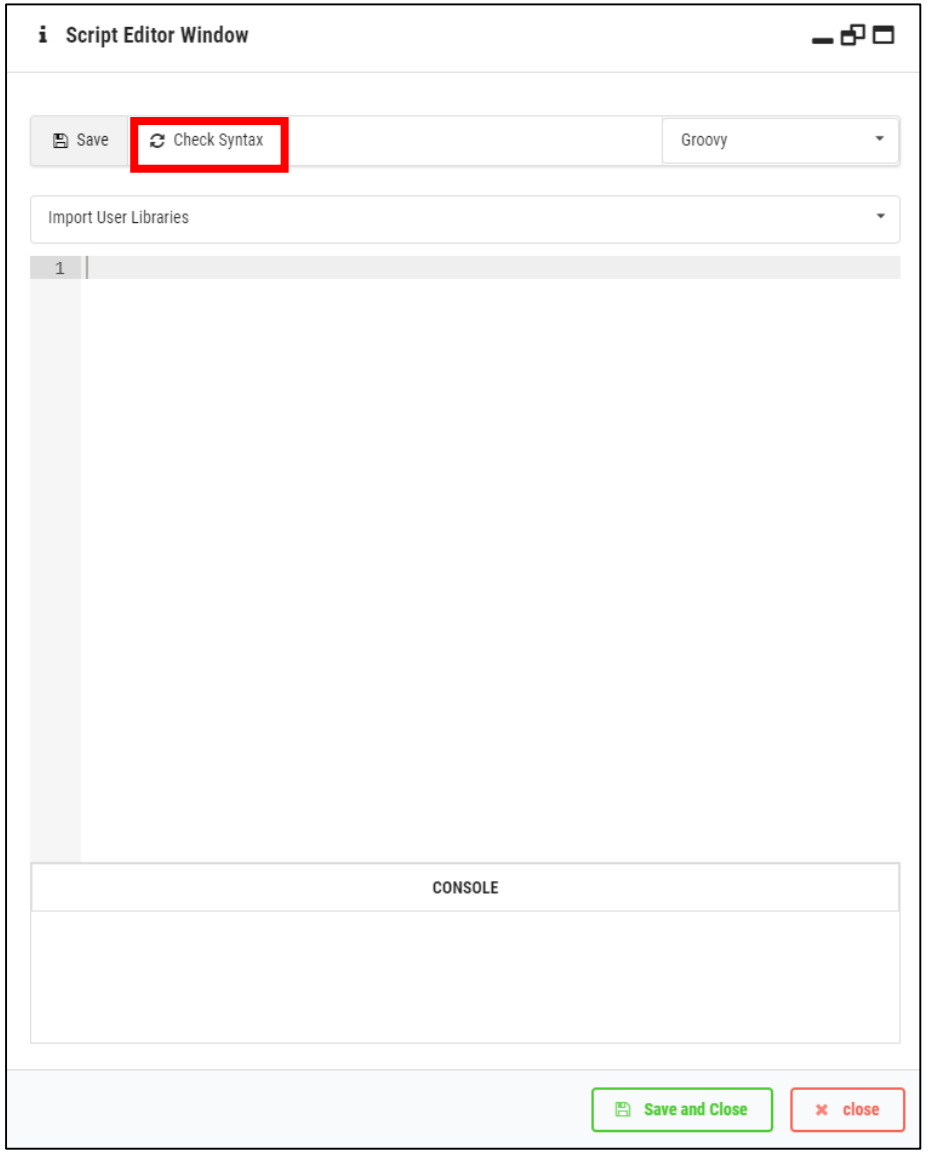
Please note: textual operators require space before/after y gt 2 is the same as y>2



Script Warning

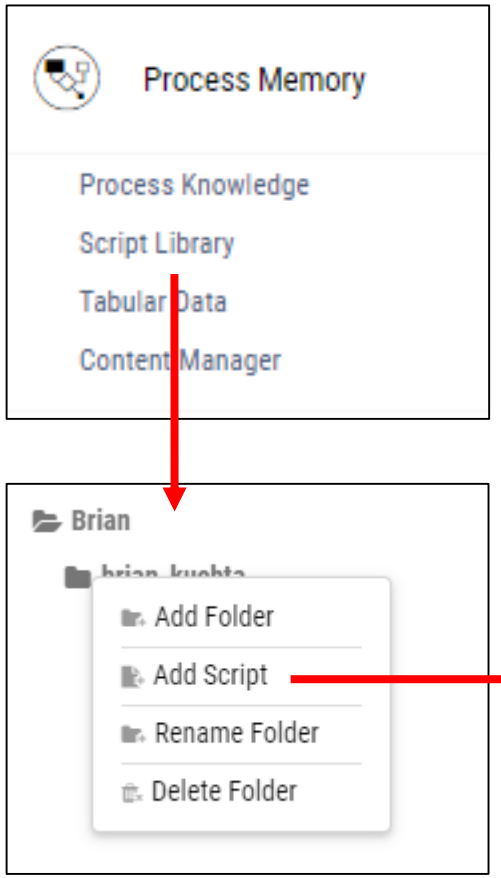
- Script editor includes Check Syntax
  - If errors exist and deploy attempted, warning appears regarding script task

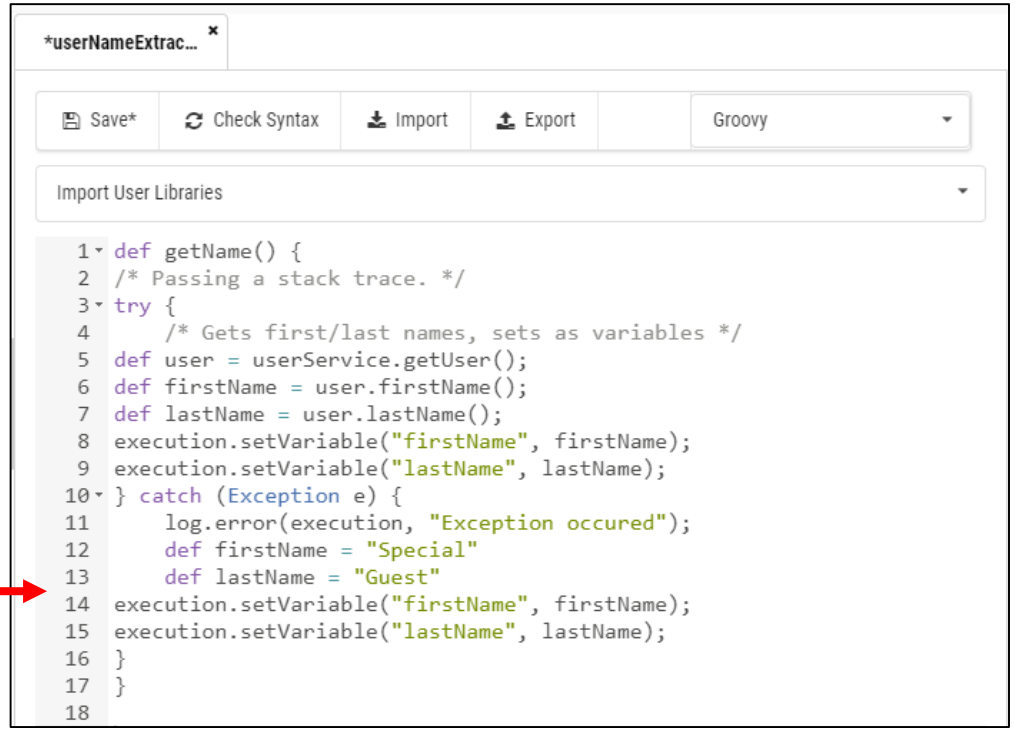




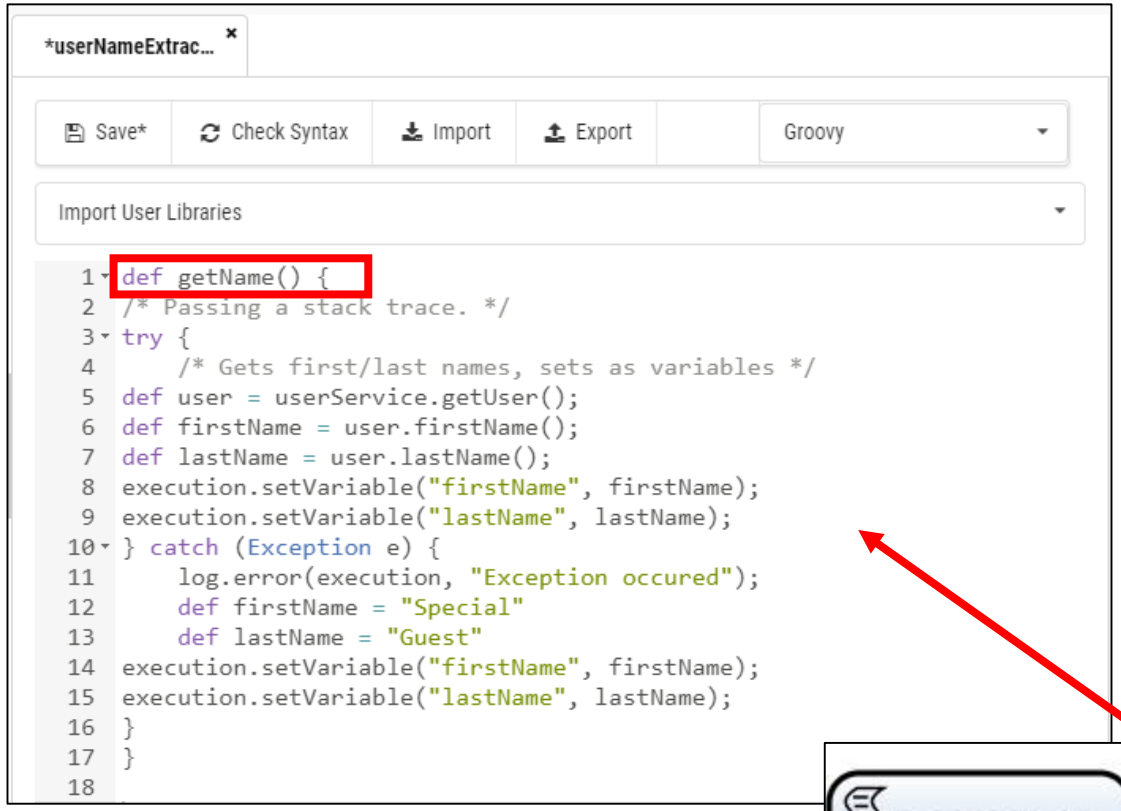
Script Library 3.5

- Create scripts to be used in multiple BPNs
  - Right click on Root -> Add folder
    - Cannot create script in Root
  - Right click on folder
    - Add Folder – Creates subfolder
    - Add Script – Creates script in folder
    - Rename Folder
    - Delete Folder





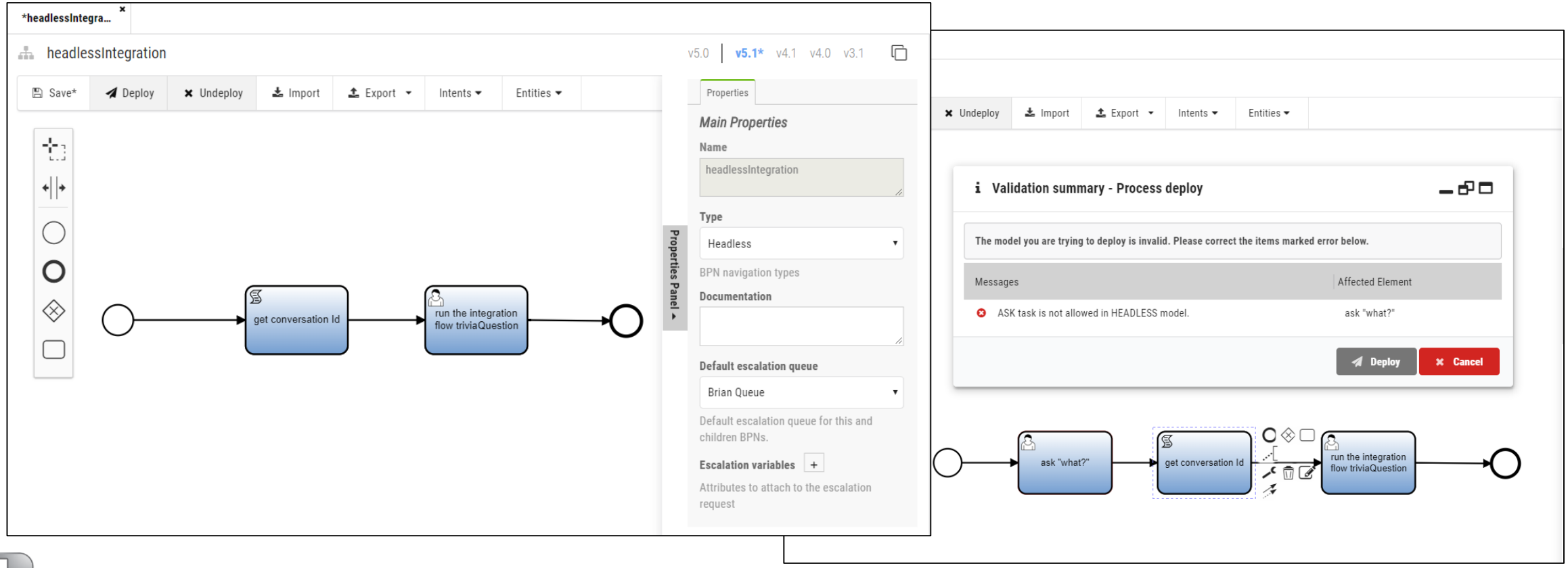
Using Scripts from Script Library in BPNs 3.5



- Create a BPN with script task
  - Click edit script on task action palette
  - Script editor opens
  - Drop down import libraries shows available scripts
  - Select script(s)
- To use any method of script library, call method directly with proper parameters

Headless BPN

- Used for non-conversational workflows
- Cannot include conversation tasks (e.g., ask, say)



Additional Scripting and Services

Content Management Services – Metadata 3.5

Exposed as **cmService**

- Used to manipulate assets in Amelia content manager

List<CmObjectMetadata> **getBucketResources**(String bucketName)

- Obtains metadata of all resources in given bucket
- Returns list containing resource metadata visible from current conversation

CmObjectMetadata **findResourceById**(String cmObjectMetadataId)

- Finds multimedia resource by its CM object metadata ID.

Returns the resource metadata/null if no resource

CmObjectMetadata **getResource**(String bucketName, String resourceName)

- Gets metadata by bucket name and resource name

Stored variable represents array of objects containing a bucket and uploaded file metadata which includes:

- bucketId - bucket ID
- bucketName - bucket Name
- objectId - uploaded resource metadata ID
- objectName - uploaded resource name
- contentType - resource content type
- contentLength - resource length
- contentDisposition - resource content disposition
- httpContentDisposition - content disposition mappable to RFC 2616 ("INLINE" or "ATTACHMENT")
- contentMd5 - resource MD5
- scope - scope to which an object is associated ("CONVERSATION" or "SYSTEM")
- url - URL to get direct access to the resource

Content Management Service

String **addFile**(String bucketName, String fileName, byte[] fileBytes)

- Adds resource/file to content manager

String **addFileBase64**(String bucketName, String fileName, String base64EncodedFile)

- Adds resource/file as Base64-encoded to content manager

String **getResourceUrl**(String cmObjectMetadataId)

- Obtains URL to get direct access to specified resource

String **getResourceUrl**(String cmObjectMetadataId, String contentDisposition)

- Obtains URL to get direct access to specified resource

byte[] **getResourceBytes**(String cmObjectMetadataId)

- Gets bytes of specified multimedia resource

String **getResourceBase64**(String cmObjectMetadataId)

- Gets resource as base64 encoded string

Context Service – Slots

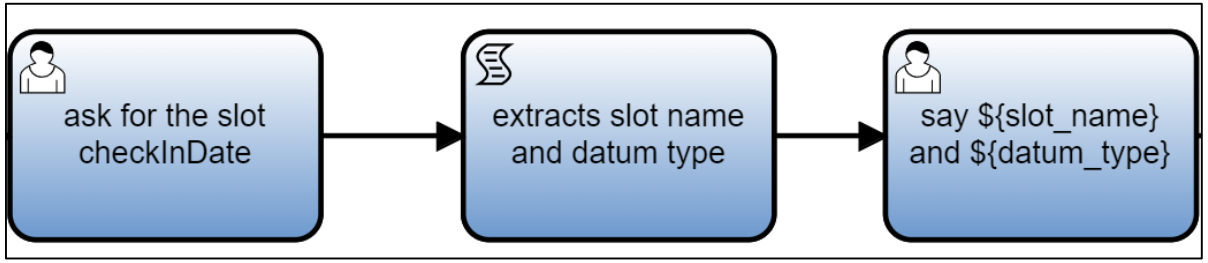
Exposed as **contextService**
- Used to access entity slots

Slot Instance Fields:
- value(): Value held by Datum object
- datumType() : Type of datum
- slotInstanceId() : UUID
- slotId (): UUID
- datum(): Populated DatumDecorator object

Set<SlotInstanceDto> **slots**()
- Gets all elicited slots in current context

List<SlotInstanceDto> **slots**(String slotCode)
- Gets elicited slots associated with current context by code

boolean **hasSlot**(String slotCode)
- Determines whether slot was elicited
- Examples:
  - ctx:hasSlot('fullName')
  - contextService.hasSlot('fullName')




```
def slot_name = contextService.latestSlot('checkInDate').value()
def datum_type = contextService.latestSlot('checkInDate').datumType()


execution.setVariable('slot_name', slot_name)
execution.setVariable('datum_type', datum_type)
```

Context Service – Slot Definitions

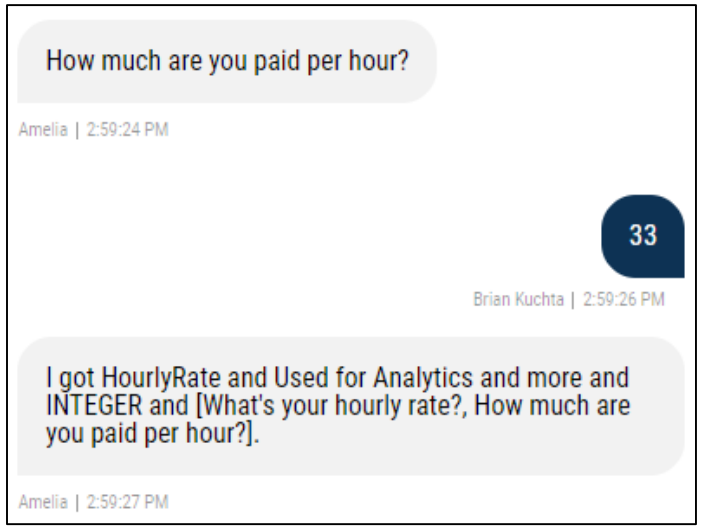
SlotDefinitionDto:
- parent: SlotInstanceDto
- code: String
- name: String
- description: String
- datumType: String
- spanless: Boolean
- questions: List<String>

Example:
```
definition = contextService.slotDefinition('HourlyRate')
execution.setVariable('name', definition.name())
execution.setVariable('description', definition.description())
execution.setVariable('datumType', definition.datumType())
execution.setVariable('question', definition.questions())}
```

List<SlotDefinitionDto> **slotDefinitions**()
- Returns list of SlotDefinitionDto objects

SlotDefinitionDto **slotDefinition**(String slotCode)
- Returns SlotDefinitionDto object or null

SlotDefinitionDto **slotDefinition**(SlotInstanceDto slotInstance)
- Obtains slot definition associated with slot instance.
- Returns SlotDefinitionDto object or null



util: Service Prefix and Slot Definitions 3.5

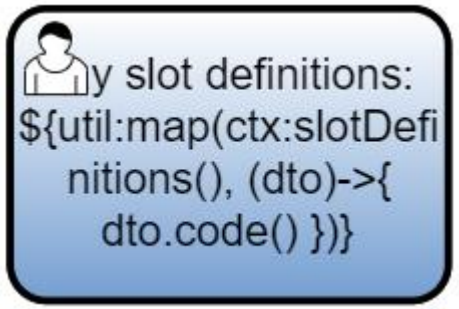
Provides utility operations
- Can be used in task names/edge expressions
- Exposed as util:

Collection<Object> **map**(Collection<Object> collection, Closure mapper)
- collection = collection to be transformed
- mapper = mapping function
- Returns collection of transformed objects

Collection<Object> **filter**(Collection<Object> collection, Closure predicate)
- collection = collection to be transformed
- predictate
- Returns collection of elements where predicate TRUE

Map example:
util:map(ctx:slotDefinitions(), (dto)->{ dto.code() })

Filter example:
util:filter(ctx:slotDefinitions(), (dto)->{ dto.datumType() == 'INTEGER' }).size()



Slot definitions: [1desserts, 1HouseSize, 1installParent, age, age_range, ageAsInt, agenumber, airport, allHotelDates, applications, area, area_range, average_monthly_hours, bathrooms, bedrooms, boolean, BusinessTravel, capital, checkInDate, checkOutDate, children, children2, colors, colorsEntityNormal, colorsNormalized, confirmationNumber, cookies, cost, costRange, countryFrom, countryParent, countryTo, county, cupcakes, Date, dateRange, dateTime, dateTimeRange, decimal, decimalRange, DistanceFromHome, distanceFromWork, dob, double, duration, durationRange, earnings, emailAddress, emailRegexTest, employee_id, foodOrder, fullAddress, Gender, hometown, hotelComplaint, hotelLocation, HourlyRate, installApplication, integerRange, job_type, lengthRange, MonthlyIncome, movieCost, movieTitles, newYorkTouristSpot, numberOfGuests, numberOfRooms, ordinal, ordinalRange, organization, Past_Due_Days, percentage, percentageRange, personName, phoneNumber, Principal, promotion, release_date, releaseDateRange, room_Type_Slot, roomCost, roomNumber, roomServiceDateTime, roomServiceDeliveryTime, roomType, salary, single, singleWordTester, smokingNonsmoking, speed, speedRange, temperature, temperatureRange, Term_in_Years, test, testCode, time, time_spend_company, timeRange, transferAmount, typeOfFoodChoice, uninstallApplication, username, usState, volume, volumeRange, weight, weightOfCar, weightRange, yearsmarried, zipCode].

Context Service – Slots

Last entity slot:

AbstractDatum **latestSlot**(String slotCode)
- Obtains current value of slot by its code

boolean **hasLatestSlotsForCode**(String slotCode)
- Checks if newly elicited slots fills slotCode

Set<SlotInstanceDto> **latestSlotsForCode**(String slotCode)
- Obtains all newly elicited slots based on slotCode

First time entity slot filled:

List<SlotInstanceDto> **newSlots**()
- Slots elicited for first time over last user response

boolean **hasNewSlotsForCode**(String slotCode)
- Checks if at least one of first time elicited slots fills slotCode

Set<SlotInstanceDto> **newSlotsForCode**(String slotCode)
- Obtains all first-time elicited slots based on slotCode

Last changed entity slot:

List<SlotInstanceDto> **modifiedSlots**()
- Obtains current slot values after one+ instances already existed in current context

boolean **hasModifiedSlotsForCode**(String slotCode)
- Checks if currently elicited slot fills slotCode after one instance already existed in current context

Set<SlotInstanceDto> **modifiedSlotsForCode**(String slotCode)
- Obtains all newly elicited slots associated with slot definition and one instances already existed in the current context

Prior entity slots excluding latest:

List<SlotInstanceDto> **oldSlots**()
- Returns values not updated over last user utterance

boolean **hasOldSlotsForCode**(String slotCode)
- Checks if least one slot within slotCode has not been created/updated over the last utterance

Set<SlotInstanceDto> **oldSlotsForCode**(String slotCode)
- Obtains all slots associated with slotCode but not created/ updated over the last utterance

Context Service and Person Datum

Stores name-related information

- Value in Person datum object => object containing details of person's name: prefix, first, middle, last, suffix

- Call value method to get desired parameter(s)
  - **.value().hasFirstName**(): Returns true if first name exists for the datum
  - **.value().firstName**(): Returns the first name of the datum
  - **.value().hasMiddleName**(): Returns true if middle name exists for the datum
  - **.value().middleName**(): Returns the middle name of the datum
  - **.value().hasLastName**(): Returns true if last name exists for the datum
  - **.value().lastName**(): Returns the last name of the datum

Examples:

```
Say ${ctx:latestSlot('username').value().firstName()}

def personDatum = contextService.latestSlot('username');
if (personDatum.value().hasFirstName()) {
firstName = personDatum.value().firstName();
}
```

Context Service – Intents

- Used to access intents

IntentInstanceDto **triggeredIntent**()
- Currently triggered/running intent, if existing

boolean **isCurrentProcessTargetedByIntent**(String intentCode)
- All intents pointing to BPN process in context/are visible from current domain

IntentDto **processIntents**()
- All intents that point to BPN process in context/are visible from current domain
  - intentCode: Code of intent definition whose instance may have been used to trigger current BPN execution

Fields:
- id : UUID
- code : String
- name : String
- actionType : EXECUTE_BPN | ECHO_GOAL
- actionSystemData : String (BPN model name)

Examples:

```
execution.setVariable('triggeredIntent',
    contextService.triggeredIntent())

execution.setVariable('isTriggeredByIntent',contextService.
    isCurrentProcessTargetedByIntent('intentName'))

execution.setVariable('processIntents',
    contextService.processIntents().collect{it.code()})
```

Conversation Service

Get conversation transcript in standard format

```
def transcript = conversationService.getTranscript()
transcript = transcript.replace("\n", "")
execution.setVariable('transcript', transcript)
```

Get the last utterance:

```
def lastUtterance =
conversationService.transcriptUtterances[-1].utterance()
execution.setVariable('lastUtterance', lastUtterance)
```

Amelia (2018-03-28 01:53:23.726): Good to see you Brian! How can I help you today? Brian Kuchta (2018-03-28 01:53:35.052): run the workflow conversationServiceTranscriptAmelia (2018-03-28 01:53:35.677): Hi there. Amelia (2018-03-28 01:53:35.872): Not to be impolite, but i have to ask...what's your age? Brian Kuchta (2018-03-28 01:53:40.726): 55.

Get conversation transcript that is formattable

List<TranscriptUtterance> getTranscriptUtterances()

TranscriptUtterance		
Accessor	**Data Type**	**Description**
sequenceId()	Integer	Utterance sequence number.
utterance()	String	The utterance.
sessionId()	UUID	Associated internal session identifier.
dateTime()	String	Date time in the yyyy-MM-dd HH:mm:ss.SSS format.
sessionMode()	SessionMode	Session mode; may be USER, AGENT, or OBSERVER.
userId()	UUID	User identifier.
username()	String	User's first name + space + last name, e.g., "John Doe".
userFirstName()	String	User's first name.
userLastName()	String	User's last name.

Escalation Queue Service

Exposed as **escalationQueueService**

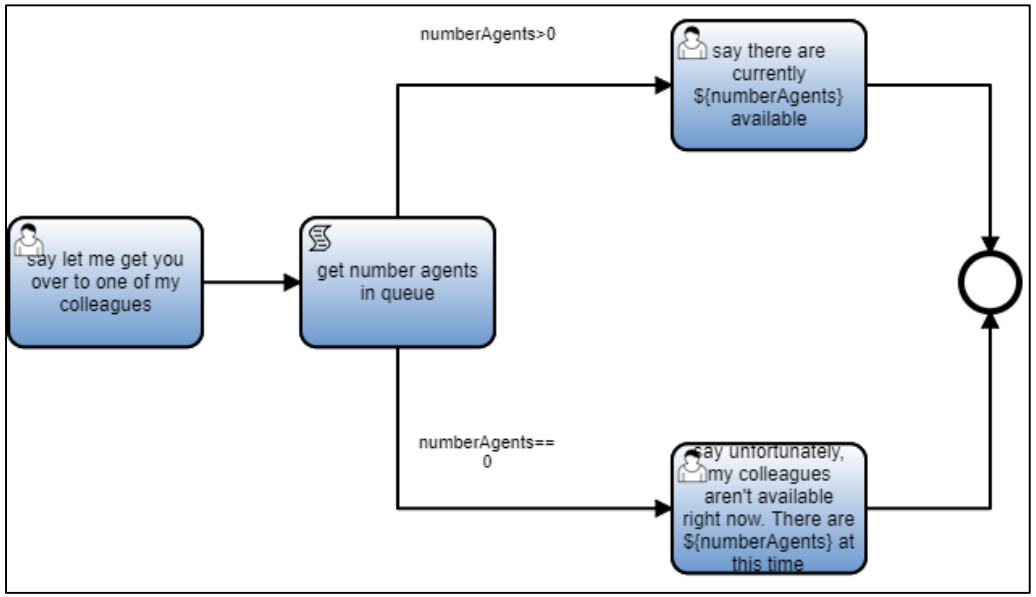
- Access agent information for given queue

int **getNumberOfAvailableAgentsByQueueCode**(String queueCode)
int **getNumberOfAvailableAgentsByQueueName**(String queueName)

int **getNumberOfLoggedInAgentsByQueueCode**(String queueCode)
int **getNumberOfLoggedInAgentsByQueueName**(String queueName)

int **getNumberOfBusyAgentsByQueueCode**(String queueCode)
int **getNumberOfBusyAgentsByQueueName**(String queueName)

List<UserDto> **getAvailableAgentsByQueueCode**(String queueCode)
List<UserDto> **getAvailableAgentsByQueueName**(String queueName)



Execution Service – Process Instance

Exposed as **execution**
- Access elements of current process

Process Instance:

String **getId**();
- Return: Process instance ID

String **processDefinitionId**();
- Return: Process definition ID

String **parentProcessInstanceId**();
- Return: Parent process instance ID

String **currentTaskInstanceId**();
- Return: Current task instance ID

String **conversationId**();
- Return: Conversation ID

String **status**();
- Return: Current process instance status

```
execution.setVariable('instanceId', execution.getId())
execution.setVariable('processId', execution.processDefinitionId())
execution.setVariable('currentId', execution.currentTaskInstanceId())
execution.setVariable('currentConv', execution.conversationId())
execution.setVariable('status', execution.status())
```

```
Process Instance ID: 1e2ffee9-6972-4cc2-90ef-40d3c8405f49
Process Definition ID: 5416a7c6-ff75-4049-bd49-dca419b00fcf
Current Task Instance ID: 88a0cb3d-399d-4683-9c78-82d83149d473
Conversation ID: LAY7ZXC6QAIAA-1
Status: ACTIVE
```

Execution Service – Process Variables

Process Variables

Object **getVariable**(String key);
- Gets variable by name in given process instance
  - key: variable name
  - Returns variable as object

Set<VariableInstance> **getVariables**();
- Get all variables in given process instance
  - Returns set of variable instances associated with current process instance

Map<String, Object> **getVariablesAsMap**();
- Get all variables in given process instance
  - Returns map of all variables associates with keys

Set<String> **getVariableNames**();
- Get all variable names for given process instance
  - Returns set of all variable names

boolean **hasVariables**();
- Test if given process instance has one+ variables
  - Returns Boolean –> TRUE if given process instance has at least one variable

boolean **containsVariable**(String key);
- Test if given process instance has variable associated with given variable name
  - key: name of variable
  - Returns Boolean –> TRUE if it contains variable

Execution Service – Process Variables

Process Variables

void **setVariable**(String key, Object value);
- Add variable to given process instance
  - key: Variable name
  - value: Variable value

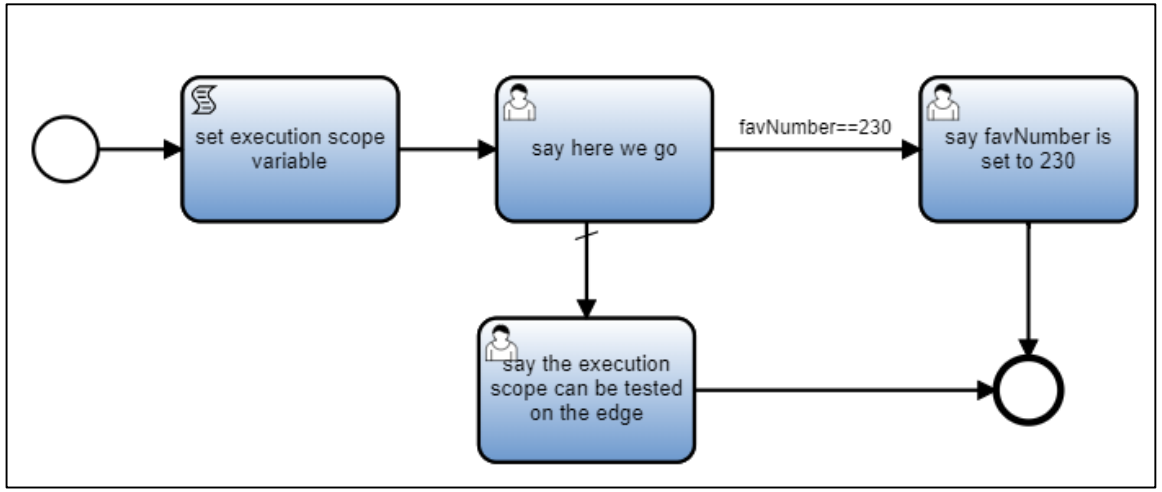
void **setVariables**(Map<String, Object> variables);
- Add variables to a process instance

void **removeVariable**(String key);
- Removes variable based on variable name in given process instance
  - key: variable name

void **removeVariables**(Set<String> keys);
- Remove set of variables based on variable names in given process instance
  - keys: Set of variable names

void **removeVariables**();
- Remove all variables for given process instance

```
execution.setVariable('favNumber', '230')
```



Execution Service – User Information

User Information

String **domainId**();
- Return: Domain ID

String **userId**();
- Return: User ID

String **userDisplayName**();
- Return: User friendly username

boolean **isAnonymous**();
- Determines if current conversation is anonymous
- Return: Boolean

```
execution.setVariable('id', execution.domainId())
execution.setVariable('user', execution.userId())
execution.setVariable('displayName', execution.userDisplayName())
execution.setVariable('anon', execution.isAnonymous())
```

```
Domain ID: 29b9396e-edc2-41c2-a704-c8468e031a7a
User ID: 7063ff2d-bd8f-486c-89d6-151a6e69710f
User Display Name: Brian Kuchta
Anonymous Login: false
```

Logging Service

Adding INFO message to log:

- **log.info**("Hello world!")

void **debug**(Object somethingOrThrowable);

- Logs a message as debug

void **info**(Object somethingOrThrowable);

- Logs a message as information

void **warn**(Object somethingOrThrowable);

- Logs a message as warning

void **error**(Object somethingOrThrowable);

- Logs a message as error
  - somethingOrThrowable String/object

```
log.info("This section shows in Diagnosis or the engine log.")
log.debug("Likely telling you there is a problem...")
log.warn("and use this when there is something going badly.")
log.error("Or when if it falls apart.")
log.info("But that rarely happens, if ever.")
```

Variables | **Diagnosis**

Time Stamp	Severity	Message
2018-04-25 07:48:00	INFO	But that rarely happens, if ever.
2018-04-25 07:48:00	ERROR	Or when if it falls apart.
2018-04-25 07:48:00	WARN	and use this when there is something going badly.
2018-04-25 07:48:00	DEBUG	Likely telling you there is a problem...
2018-04-25 07:48:00	INFO	This section shows in Diagnosis or the engine log.

Text Service

Exposed as **textService**

- Text matching operations not available in Groovy scripting engines

Table<String, String, Double> **searchExact**(Set<String> values, String sentence, boolean caseSensitive)

- Searches for set of values in sentence

Triple<String, String, Double> **bestExactMatch**(Set<String> values, String sentence, boolean caseSensitive)

- Finds closest match in specified search

Table<String, String, Double> **searchFuzzy**(Set<String> values, String sentence, Double tokenMatchThreshold)

- Fuzzily matches set of strings against sentence
    - tokenMatchThreshold: similarity threshold (0-1) for match

Triple<String, String, Double> **bestFuzzyMatch**(Set<String> values, String sentence, Double tokenMatchThreshold)

- Finds closest fuzzy match in specified search

Transient Variable Service

Exposed as **transientVariableService**

- Separate scope for variables longer than 64kB
- Accessible across processes during conversation
- Removed once conversation closed
- Faster than process scope as not storied in memory

Object **addVariable**(String name, Object value)

Map<String, Object> **addVariables**(Map<String, Object> variables)

Boolean **hasVariable**(Sting name)

Object **getVariable**(String name)

- Current process instance/Returns null if no variable

Map<String, Object> **getVariables**()

void **clearVariables**()

- Removes all variables in current conversation

```
transientVariableService.addVariable('varName', 'varValue')

def localVar = transientVariableService.getVariable('bpnVar')
```

bpn: Prefix, Process (Regular) Variables and Transient Variables

Boolean **bpn:hasRvar**(String regularVariableName)
String **bpn:rvar**(String regularVariableName)
Map<String, Object> **bpn:rvars**(String... regularVariableNames)

Boolean **bpn:hasTvar**(String transientVariableName)
String **bpn:tvar**(String transientVariableName)
Map<String, Object> **bpn:tvars**(String... transientVariableNames)

Boolean **bpn:hasVar**(String varName)
String **bpn:var**(boolean regularFirst, String variableName)
Map<String, Object> **bpn:vars**(boolean regularFirst, String... variableNames)

- Searches one scope then the other for varName
- Should only be used if variables with same name found in different scopes or when merging process (regular) and transient
- Resource intensive -> Do NOT use otherwise

? = single character wildcard
% = multiple character wildcard

Boolean **bpn:hasRvarLike**(String sqlLikeVarNamePattern)
Boolean **bpn:hasTvarLike**(String sqlLikeVarNamePattern)
Boolean **bpn:hasVarLike**(String sqlLikeVarNamePattern)

String **bpn:rvarsLike**(String sqlLikeVarNamePattern)
String **bpn:tvarsLike**(String sqlLikeVarNamePattern)
Map<String, Object> **bpn:varsLike**(String sqlLikeVarNamePattern)

User Service

Exposed as **userService**

- Access/modify user account information

BpnUser **getUser**();

- Get user associated with current conversation

BpnUser **registerUser**(BpnUser bpnUser, String userGroupName);

- Register new user

BpnUser **updateUser**(BpnUser existingUser);

- Update user associated with given conversation

boolean **userExistsWithName**(String name);

- Returns true if user already exists with given name

boolean **userExistsWithEmail**(String email);

- Returns true if a user already exists with given email

Name	Accessor	Mutator(Setter)
anonymous	anonymous()	N/A (defaults to false)
attributes	attributes()	attributes(Map<String, String> m)
email	email()	email(String s)
firstName	firstName()	firstName(String s)
lastName	lastName()	lastName(String s)
userId	userId()	N/A (defaults to null)

```
def user = userService.getUser()
def firstName = user.firstName()
def lastName = user.lastName()
def emailAddress = user.email()

execution.setVariable("firstName", firstName)
execution.setVariable("lastName", lastName)
execution.setVariable("email", emailAddress)
```

Library Scripts with Parameters

dateAsStri... | dateVaria... | **stringVari...** | variablePa... | randonCo... | userName...

Save* | Check Syntax | Groovy

Import User Libraries

```groovy
def colorsStatement(String colors){
def statement = "Your favorite color is $colors"
execution.setVariable('statement', statement)
}

```

stringVari...

```groovy
def colors = contextService.latestSlot('colors').value()

colorsStatement(colors);

```

dateAsStri... | dateVaria... | stringVari... | **variableP...** | randonCo... | userName...

Save* | Check Syntax | Groovy

Import User Libraries

```groovy
def costStatement(long cost){
def statement = "The cost is $cost"
execution.setVariable('statement', statement)
}

```

variableP...

```groovy
def cost = contextService.latestSlot('cost').value()

costStatement(cost);

```

Presenting Dynamic Content

```
<h1><strong>We welcome you to Training Domain Hotels, ${firstName}.</strong>
</h1>

<hr />
<p><img alt="" src="/Amelia/admin/api/bucket/resource/file/6d09ef0d-e6c0-495d-
88a5-bd45b38f6398" style="border-style:solid; border-width:2px" /></p>

<p>Your reservation is for a ${roomTypeVar} starting ${checkInDate}.</p>

<p>We have you checking out on ${checkOutDate}.</p>

<p>Here is your confirmation number:</p>

<p>${confirmationNumber}</p>

<p>Should you require more information about our hotel, please visit:</p>

<h2><a href="http://www.ipsoft.com" target="_blank">Training Domain Hotels</a>
</h2>
```

We welcome you to Training Domain Hotels, ${firstName}.

Your reservation is for a ${roomTypeVar} starting ${checkInDate}.

We have you checking out on ${checkOutDate}.

Here is your confirmation number:

We welcome you to Training Domain Hotels, Brian.

Your reservation is for a single starting 12/12/2017.

Activity

```
def transcript = conversationService.getTranscript()

transcript = transcript.replace("\n", "")

execution.setVariable('transcript', transcript)
```

```
log.info("This section shows in Diagnosis or the engine log.")

log.debug("Likely telling you there is a problem...")

log.warn("and use this when there is something going badly.")

log.error("Or when if it falls apart.")

log.info("But that rarely happens, if ever.")
```

Timing: 20 minutes
Validation: Mind View Testing

Script Service

- Create a script task and include a basic script in your BPN
  OR
- Use one of the script services in your Amelia skill BPN (if you are not a developer, keep it simple).
- Some simple examples are provided in the exercise files.

Formatting Entities with quantityService and qty:

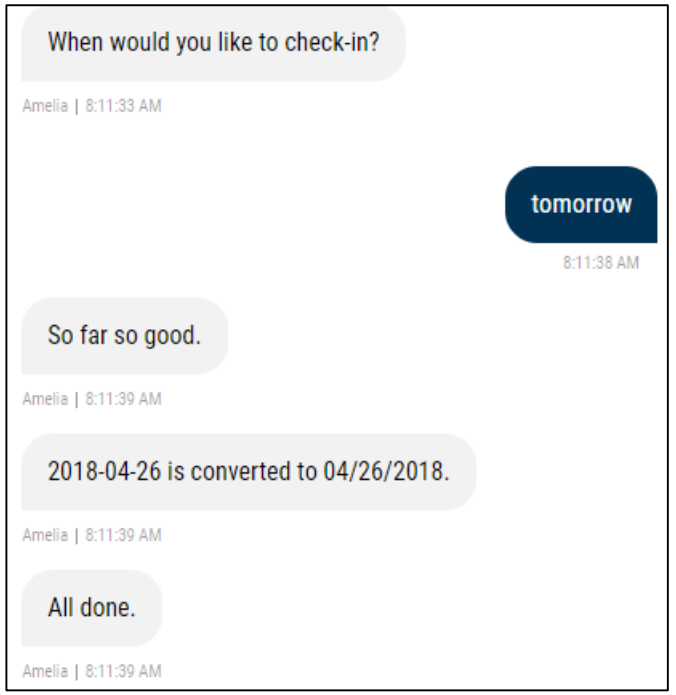
quantityService format and normalizeAndFormat

Exposed as **quantityService**

- Can normalize/format values from script tasks and libraries

format(AbstractDatum datum)
format(AbstractDatum datum, Formatter formatter)
format(AbstractDatum datum, Formatter formatter, >, timeZones<timezone>)

```
def checkInDateCap = contextService.latestSlot('checkInDate')
def checkInDateVal = quantityService.format(checkInDateCap, Formatters.DATE)
execution.setVariable('checkInDate', checkInDateVal)
```



With String, normalize to AbstractDatum object and format with Formatter:
normalize(String text, DatumType.<datum type>)
normalizeAndFormat(String text, DatumType.<datum type>, Formatters.<formatter>)
normalizeAndFormat(String text, DatumType.<datum type>, Formatters.<formatter>, timeZones<timezone>)

```
def formattedValue = quantityService.normalizeAndFormat("12/01/2018", DatumType.DATE, Formatters.DATE_IN_WORDS)
Returns: Saturday, December 01 2018
```

Formatters

Simple Formatters	Location Formatters	Unit Formatters	Simple Range Formatters	Unit Range Formatters
• BOOLEAN • DATE • DATE_IN_WORDS • DATE_LONG • DATE_MEDIUM • DATE_SHORT • DATE_TIME_12_HR • DATE_TIME_24_HR • DATE_TIME_IN_WORDS_12_HR • DATE_TIME_IN_WORDS_24_HR • DATE_TIME_LONG_12_HR • DATE_TIME_LONG_24_HR • DATE_TIME_MEDIUM_12_HR • DATE_TIME_MEDIUM_24_HR • DATE_TIME_SHORT_12_HR • DATE_TIME_SHORT_24_HR • DATE_TIME_WITH_TIME_ZONE_12_HR • DATE_TIME_WITH_TIME_ZONE_24_HR • DECIMAL • DECIMAL_IN_WORDS • EMAIL • INTEGER • INTEGER_IN_WORDS • ORDINAL • ORDINAL_IN_WORDS • ORGANIZATION • PERSON_FULL_NAME • PERSON_FIRST_NAME • PERSON_LAST_NAME • PHONE_NUMBER • TEXT • TIME_12_HR • TIME_24_HR	• AIRPORT_NAME • AIRPORT_CODE • CAPITAL • COUNTRY_NAME • COUNTRY_CODE • CITY • US_COUNTY • US_STATE_NAME • US_STATE_CODE • STREET_ADDRESS • POSTAL_CODE	• AGE • AGE_IN_WORDS • AREA_METRIC • AREA_IMPERIAL • AREA_METRIC_IN_WORDS • AREA_IMPERIAL_IN_WORDS • CURRENCY • CURRENCY_IN_WORDS • DURATION • DURATION_IN_WORDS • LENGTH_METRIC • LENGTH_IMPERIAL • LENGTH_METRIC_IN_WORDS • LENGTH_IMPERIAL_IN_WORDS • PERCENTAGE • PERCENTAGE_IN_WORDS • SPEED_METRIC • SPEED_METRIC_IN_WORDS • SPEED_IMPERIAL • SPEED_IMPERIAL_IN_WORDS • TEMPERATURE_METRIC • TEMPERATURE_METRIC_IN_WORDS • TEMPERATURE_IMPERIAL • TEMPERATURE_IMPERIAL_IN_WORDS • VOLUME_METRIC • VOLUME_METRIC_IN_WORDS • VOLUME_IMPERIAL • VOLUME_IMPERIAL_IN_WORDS • WEIGHT_METRIC • WEIGHT_METRIC_IN_WORDS • WEIGHT_IMPERIAL • WEIGHT_IMPERIAL_IN_WORDS	• TIME_RANGE_12_HR • TIME_RANGE_24_HR • DATE_RANGE • DATE_RANGE_IN_WORDS • DATE_RANGE_LONG • DATE_RANGE_MEDIUM • DATE_RANGE_SHORT • DATE_TIME_RANGE_12_HR • DATE_TIME_RANGE_24_HR • DATE_TIME_RANGE_IN_WORDS_12_HR • DATE_TIME_RANGE_IN_WORDS_24_HR • DATE_TIME_RANGE_LONG_12_HR • DATE_TIME_RANGE_LONG_24_HR • DATE_TIME_RANGE_MEDIUM_12_HR • DATE_TIME_RANGE_MEDIUM_24_HR • DATE_TIME_RANGE_SHORT_12_HR • DATE_TIME_RANGE_SHORT_24_HR • INTEGER_RANGE • INTEGER_RANGE_IN_WORDS • DECIMAL_RANGE • DECIMAL_RANGE_IN_WORDS • ORDINAL_RANGE • ORDINAL_RANGE_IN_WORDS	• AGE_RANGE • AGE_RANGE_IN_WORDS • AREA_RANGE_METRIC • AREA_RANGE_METRIC_IN_WORDS • CURRENCY_RANGE • CURRENCY_RANGE_IN_WORDS • DURATION_RANGE • DURATION_RANGE_IN_WORDS • LENGTH_RANGE_METRIC • LENGTH_RANGE_METRIC_IN_WORDS • PERCENTAGE_RANGE • PERCENTAGE_RANGE_IN_WORDS • SPEED_RANGE_METRIC • SPEED_RANGE_METRIC_IN_WORDS • TEMPERATURE_RANGE_METRIC • TEMPERATURE_RANGE_METRIC_IN_WORDS • VOLUME_RANGE_METRIC • VOLUME_RANGE_METRIC_IN_WORDS • WEIGHT_RANGE_METRIC • WEIGHT_RANGE_METRIC_IN_WORDS • AREA_RANGE_IMPERIAL • AREA_RANGE_IMPERIAL_IN_WORDS • LENGTH_RANGE_IMPERIAL • LENGTH_RANGE_IMPERIAL_IN_WORDS • SPEED_RANGE_IMPERIAL • SPEED_RANGE_IMPERIAL_IN_WORDS • TEMPERATURE_RANGE_IMPERIAL • TEMPERATURE_RANGE_IMPERIAL_IN_WORDS • VOLUME_RANGE_IMPERIAL • VOLUME_RANGE_IMPERIAL_IN_WORDS • WEIGHT_RANGE_IMPERIAL • WEIGHT_RANGE_IMPERIAL_IN_WORDS

Normalize – Datum Types

Simple Datum
- DatumType.TEXT
- DatumType.BOOLEAN
- DatumType.INTEGER
- DatumType.DECIMAL
- DatumType.ORDINAL
- DatumType.AGE
- DatumType.EMAIL
- DatumType.PERSON
- DatumType.ORGANIZATION
- DatumType.DATE
- DatumType.TIME

Location Datum
- DatumType.AIRPORT
- DatumType.PHONE_NUMBER
- DatumType.POSTAL_CODE
- DatumType.CAPITAL
- DatumType.COUNTRY
- DatumType.US_CITY
- DatumType.US_STATE
- DatumType.US_COUNTY
- DatumType.US_STREET_ADDRESS

Range Datum
- DatumType.DATE_TIME
- DatumType.DATE_RANGE
- DatumType.DATE_TIME_RANGE
- DatumType.DECIMAL_RANGE
- DatumType.INTEGER_RANGE
- DatumType.ORDINAL_RANGE
- DatumType.TIME_RANGE
- DatumType.AGE_RANGE
- DatumType.AREA_RANGE
- DatumType.CURRENCY_RANGE
- DatumType.DURATION_RANGE
- DatumType.LENGTH_RANGE
- DatumType.PERCENTAGE_RANGE
- DatumType.SPEED_RANGE
- DatumType.TEMPERATURE_RANGE
- DatumType.VOLUME_RANGE
- DatumType.WEIGHT_RANGE

- **Unit Datum**
- DatumType.CURRENCY
- DatumType.DURATION
- DatumType.AGE
- DatumType.AREA
- DatumType.LENGTH
- DatumType.VOLUME
- DatumType.WEIGHT
- DatumType.SPEED
- DatumType.PERCENTAGE
- DatumType.TEMPERATURE

Time Zone Examples

TimeZones.ETC_GMT_PLUS_12
TimeZones.ETC_GMT_PLUS_11
TimeZones.MIT
TimeZones.PACIFIC_APIA
TimeZones.PACIFIC_MIDWAY
TimeZones.PACIFIC_NIUE
TimeZones.PACIFIC_PAGO_PAGO
TimeZones.PACIFIC_SAMOA
TimeZones.US_SAMOA
TimeZones.AMERICA_ADAK
TimeZones.AMERICA_ATKA
TimeZones.ETC_GMT_PLUS_10
TimeZones.HST
TimeZones.PACIFIC_FAKAOFO
TimeZones.PACIFIC_HONOLULU
TimeZones.PACIFIC_JOHNSTON
TimeZones.PACIFIC_RAROTONGA
TimeZones.PACIFIC_TAHITI
TimeZones.SYSTEMV_HST10

TimeZones.US_ALEUTIAN
TimeZones.US_HAWAII
TimeZones.PACIFIC_MARQUESAS
TimeZones.AST
TimeZones.AMERICA_ANCHORAGE
TimeZones.AMERICA_JUNEAU
TimeZones.AMERICA_NOME
TimeZones.AMERICA_YAKUTAT
TimeZones.ETC_GMT_PLUS_9
TimeZones.PACIFIC_GAMBIER
TimeZones.SYSTEMV_YST9
TimeZones.SYSTEMV_YST9YDT
TimeZones.US_ALASKA
TimeZones.AMERICA_DAWSON
TimeZones.AMERICA_ENSENADA
TimeZones.AMERICA_LOS_ANGELES
TimeZones.AMERICA_TIJUANA
TimeZones.AMERICA_VANCOUVER
TimeZones.AMERICA_WHITEHORSE

TimeZones.CANADA_PACIFIC
TimeZones.CANADA_YUKON
TimeZones.ETC_GMT_PLUS_8
TimeZones.MEXICO_BAJANORTE
TimeZones.PST
TimeZones.PST8PDT
TimeZones.PACIFIC_PITCAIRN
TimeZones.SYSTEMV_PST8
TimeZones.SYSTEMV_PST8PDT
TimeZones.US_PACIFIC
TimeZones.US_PACIFIC_NEW
TimeZones.AMERICA_BOISE
TimeZones.AMERICA_CAMBRIDGE_BAY
TimeZones.AMERICA_CHIHUAHUA
TimeZones.AMERICA_DAWSON_CREEK
TimeZones.AMERICA_DENVER
TimeZones.AMERICA_EDMONTON
TimeZones.AMERICA_HERMOSILLO
TimeZones.AMERICA_INUVIK

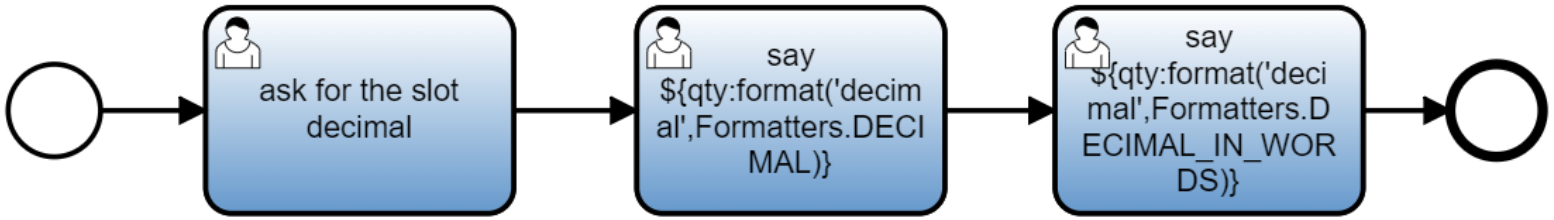
qty: Prefix

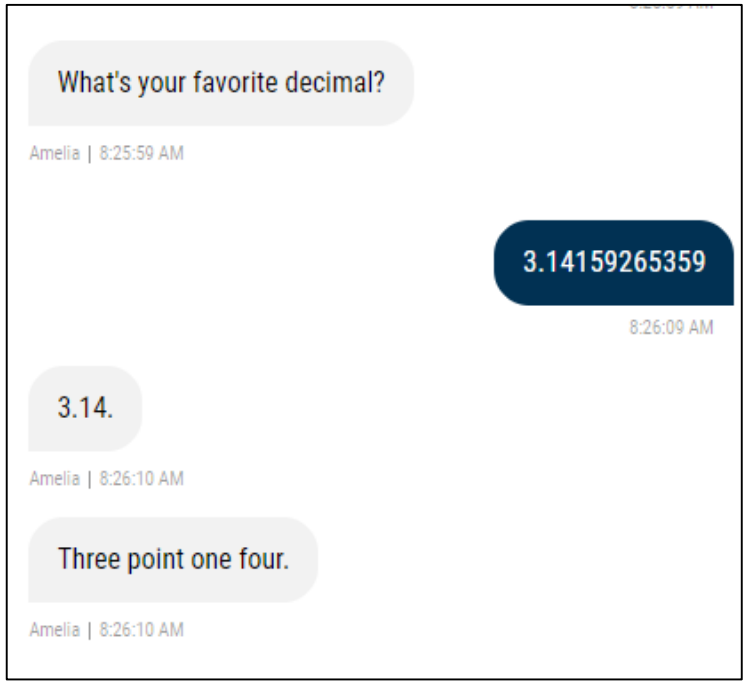
Service prefix for formatting entities in tasks

format(String slotCode)
format(String slotCode, Formatters.<FormatterType>)
format(String slotCode, Formatters.<FormatterType>, TimeZones.<timeZone>)

- Formats latest slot instance with default formatter for datum type of object

Example:
say ${qty:format('decimal',Formatters.DECIMAL_IN_WORDS)}

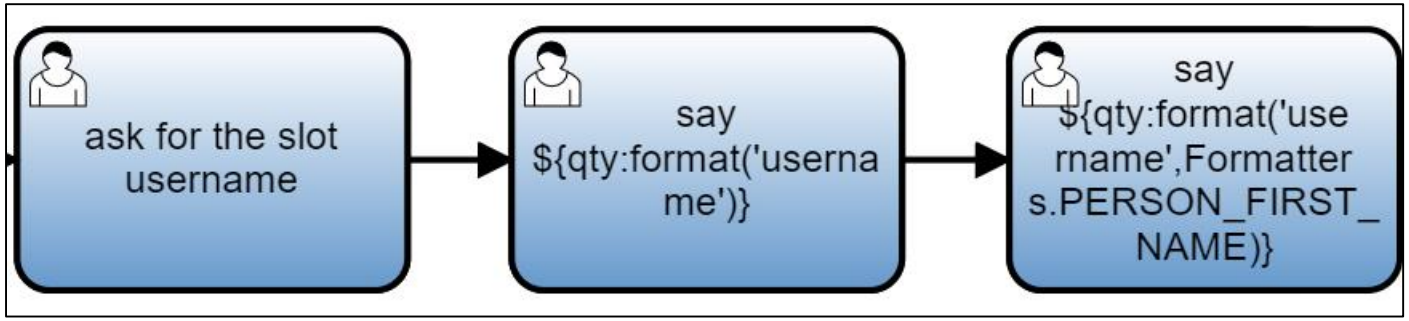




Activity

quantityService & qty:

- Add a formatter either in a script task using quantityService or in a task using qty: prefix.
- Test your revised BPN in Process Memory in mind view.



Timing: 20 minutes
Validation: Mind View Testing

Clarifying Question Asker (CQA)

Clarifying Question Asker (CQA)

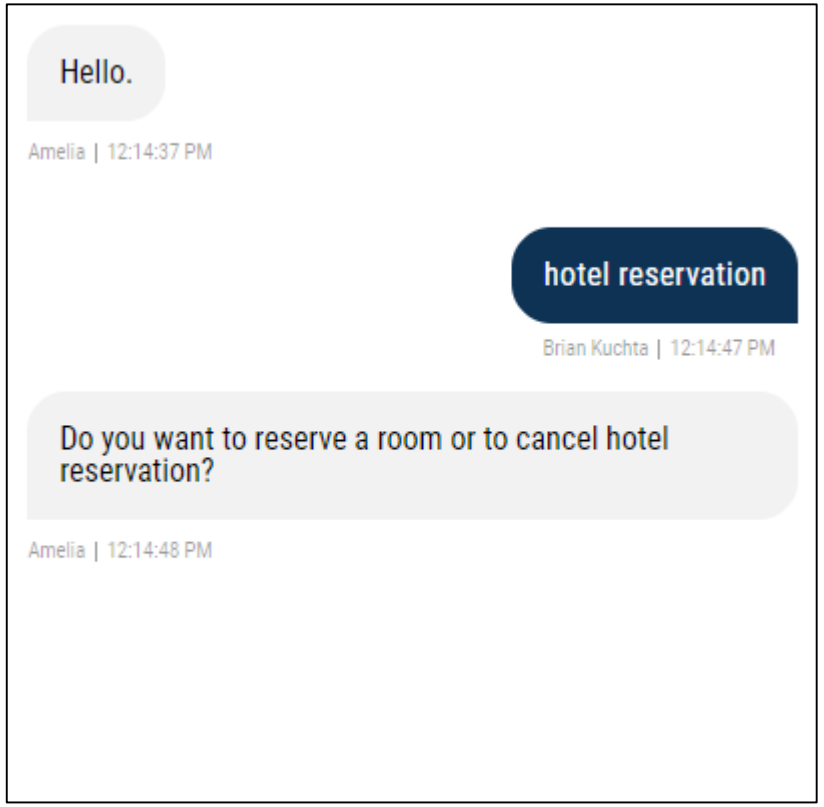
Generates a clarifying question when possible ambiguity between two similar intents

Example intents with potential for ambiguity:

- Cancel a hotel reservation
- Book a hotel reservation

- Open checking account
- Open saving account

- Order room service
- Order pay per view movie

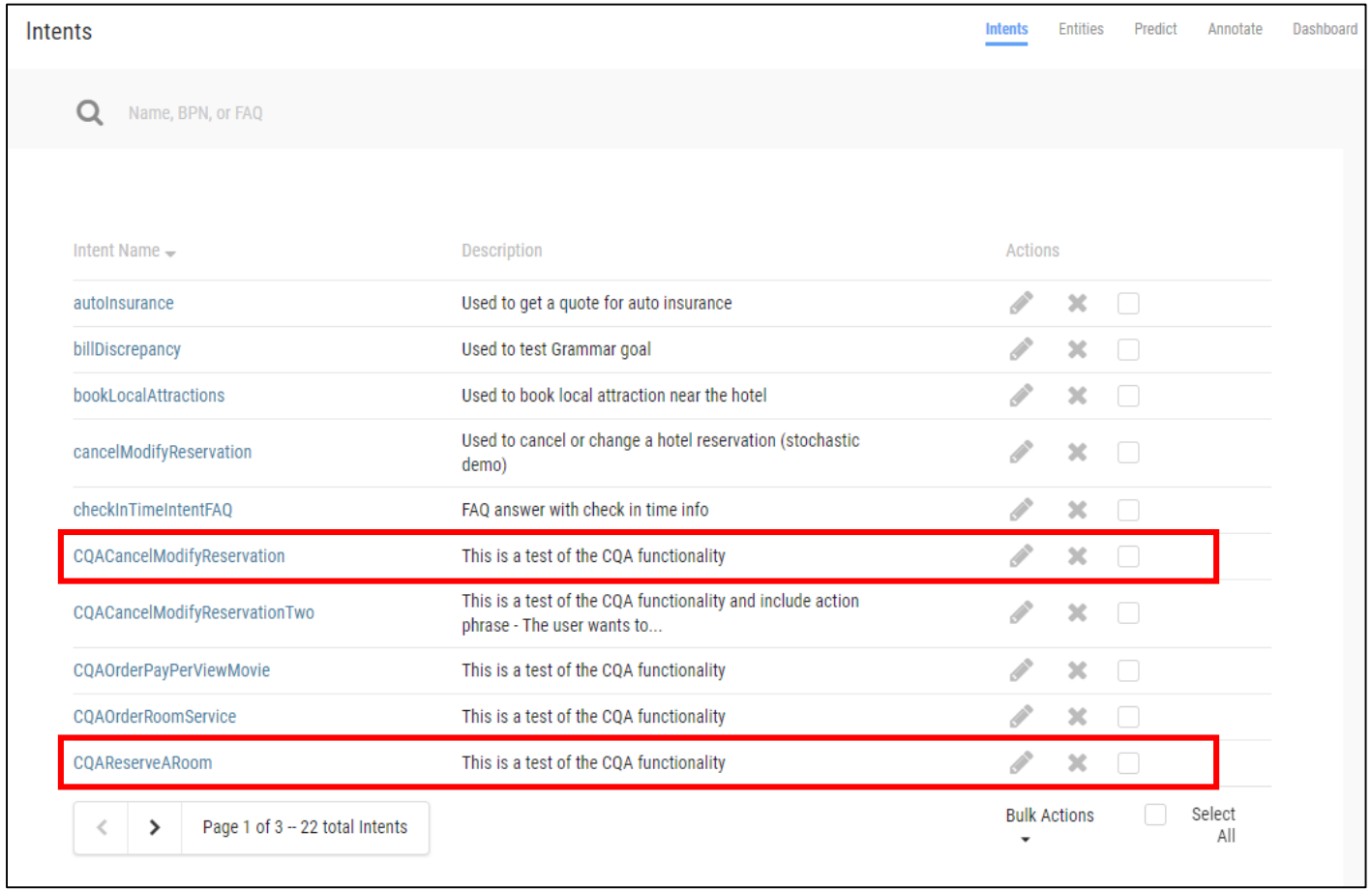


Clarifying Question Asker (CQA)

Create at least two intents in which there could be ambiguity

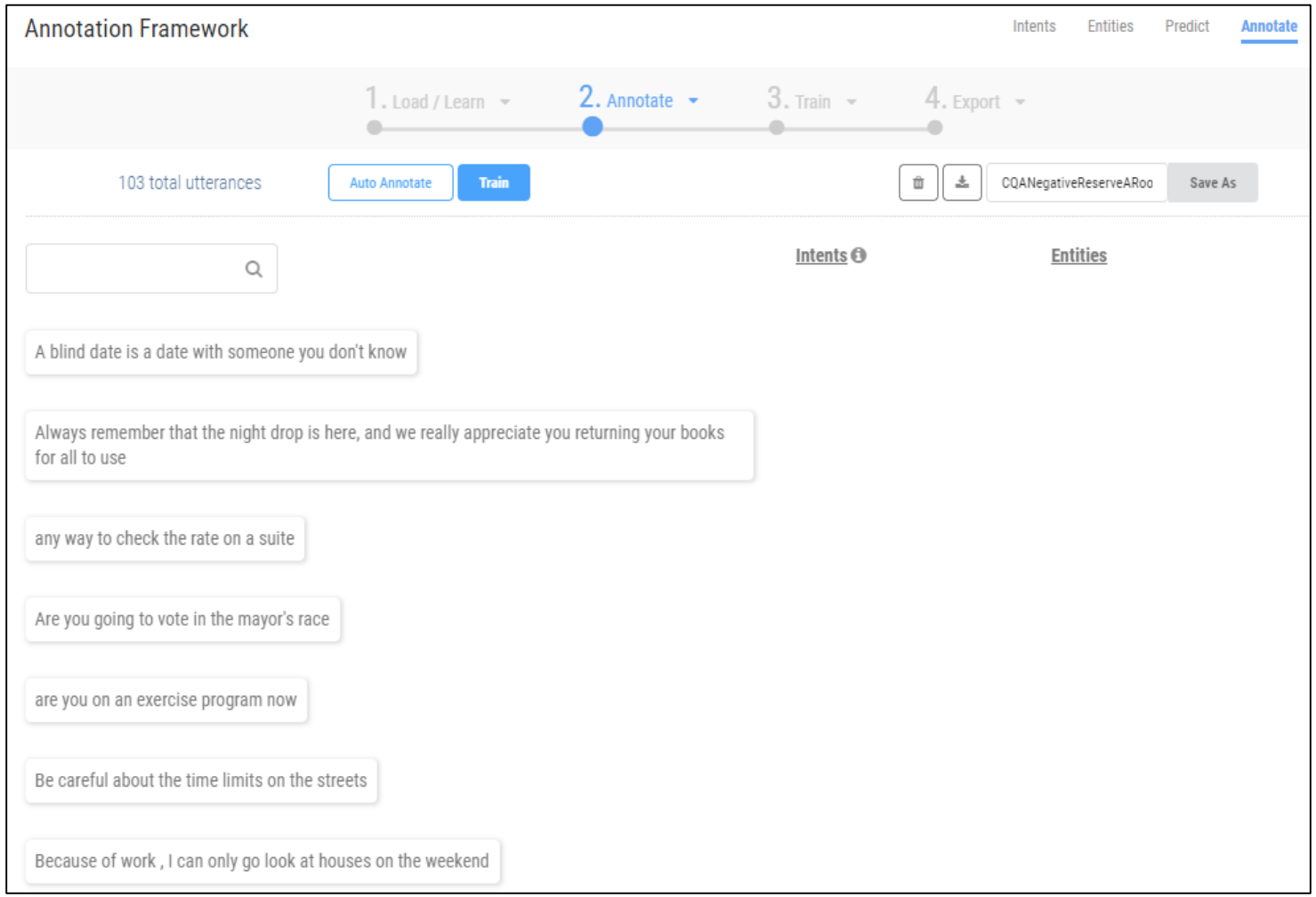
- Make a hotel reservation
- Cancel a hotel reservation

Include the relevant example utterances for the intents in the intents or as a separate training corpus uploaded into annotation framework



Clarifying Question Asker (CQA)

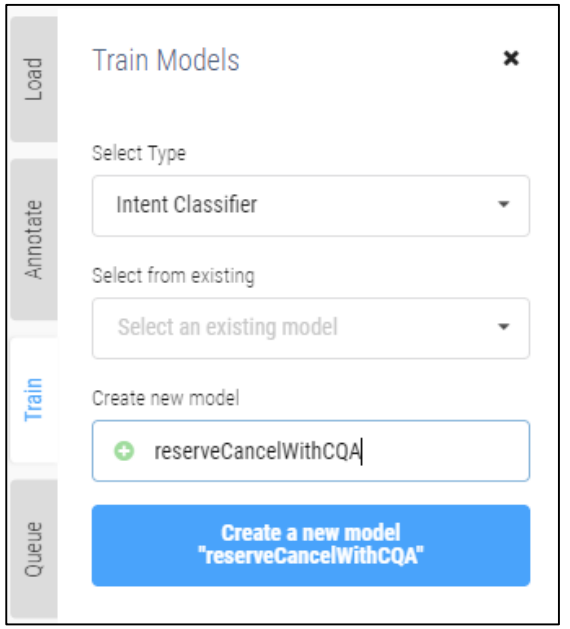
Upload in-domain and out-of-domain negative utterances into annotation framework

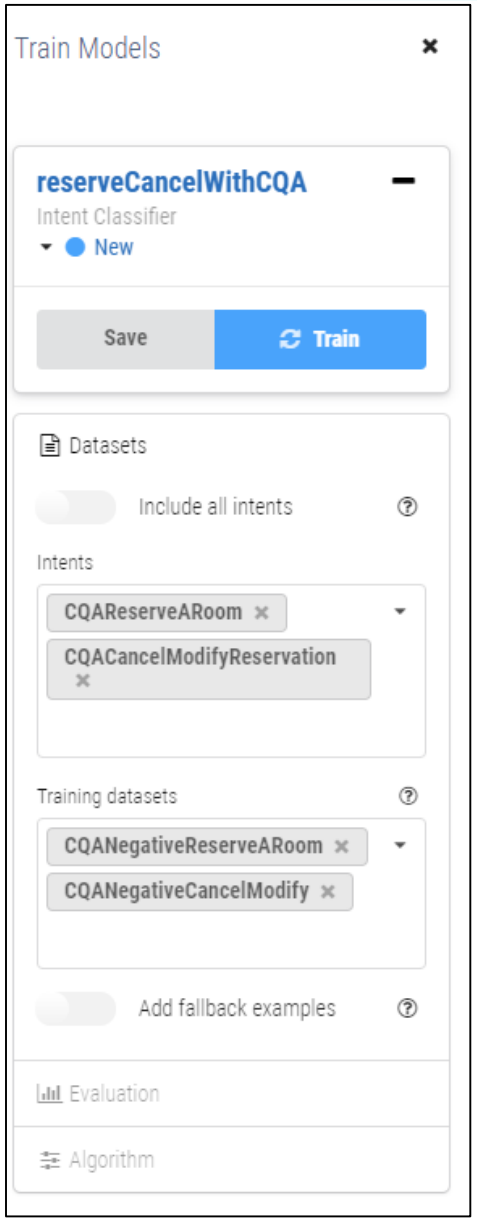


Clarifying Question Asker (CQA)

Train the intent classifier model

- Select the appropriate datasets – example utterances within the intent and/or intent utterances uploaded into annotation framework
- Include the appropriate negative dataset(s)





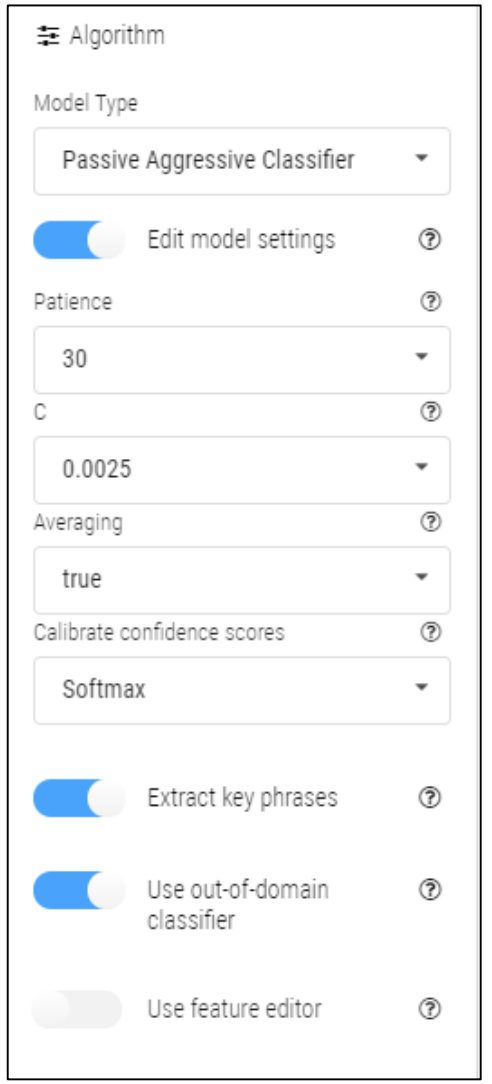
Clarifying Question Asker (CQA)

In the Algorithm panel

- Select your chosen algorithm
  - If using Passive Aggressive, LSVM, or Logistic Regression, set Calibrate Confidence Score to Softmax (recommended)
- Turn on Extract key phrases
- Turn on Use out-of-domain classifier

Ambiguity detection uses intent classifier scores

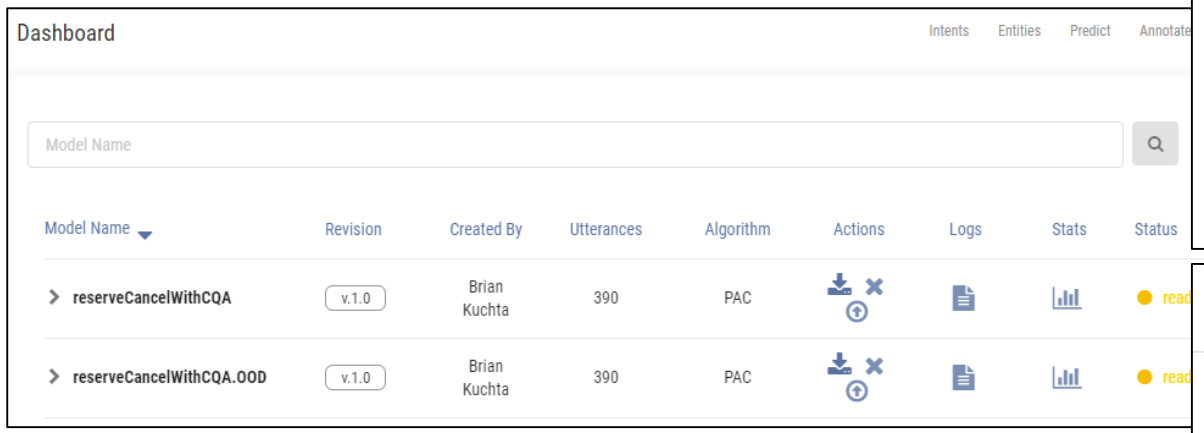
- Scores need to be calibrated (preferably using SoftMax) – allows for output of probabilities
- Key phrases extracted to distinguish intents/as intent descriptors
- Out-of-domain classifier distinguishes between all in-domain utterances and out-of-domain (negative) to support ambiguity detection



Clarifying Question Asker (CQA)

After training completes, review Dashboard

- Two models created – intent model (distinguishes the two intents) and out-of-domain model (distinguishes between in-domain and out-of-domain)
- Both models need to be deployed to take advantage of CQA



reserveCancelWithCQA_5-fold_cv: Statistics

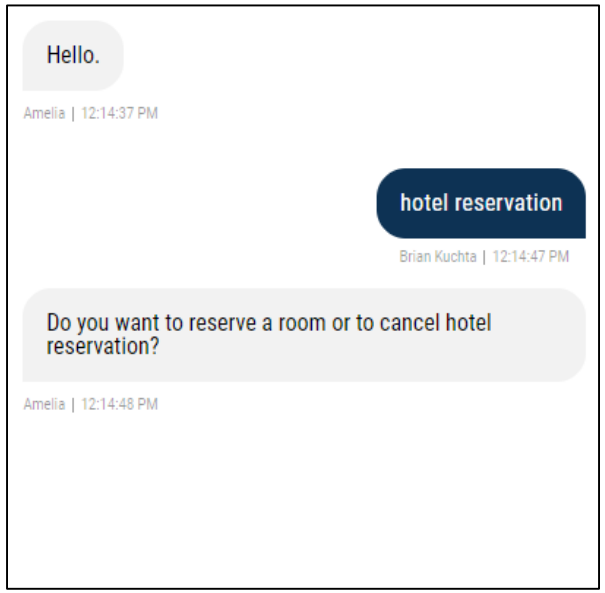
	Correct	Gold	System	Precision	Recall	F1
CQACancelModifyReservation	76	91	83	91.57	83.52	87.36
CQAReserveARoom	84	91	99	84.85	92.31	88.42
Overall	160	182	182	87.91	87.91	87.91

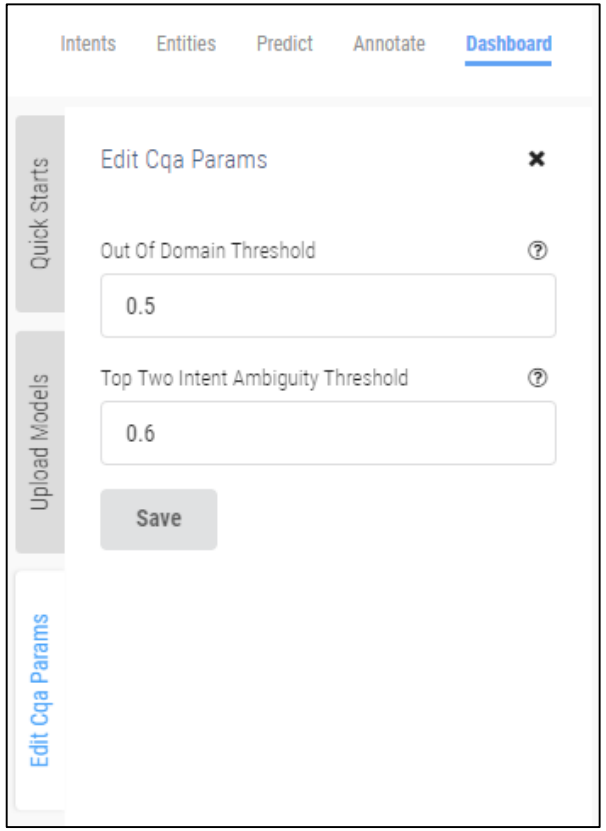
reserveCancelWithCQA.OOD_5-fold_cv: Statistics

	Correct	Gold	System	Precision	Recall	F1
None	199	208	211	94.31	95.67	94.99
in_domain	170	182	179	94.97	93.41	94.18
Overall	369	390	390	94.62	94.62	94.62

How CQA Works

- End user utterance analyzed by Out-Of-Domain model to determine if In-Domain given the Out-Of-Domain threshold
  - If out of domain, negative class is triggered
  - If in domain, determines if there is ambiguity given the confidence score of top two intents
    - If difference between top two intent scores **is <u>not</u> less than Top-Two-Intent-Ambiguity-Threshold**, there is NO ambiguity and winning intent triggers
    - If difference between top two intent scores **is less than Top-Two-Intent-Ambiguity-Threshold**, there is ambiguity and CQA triggers

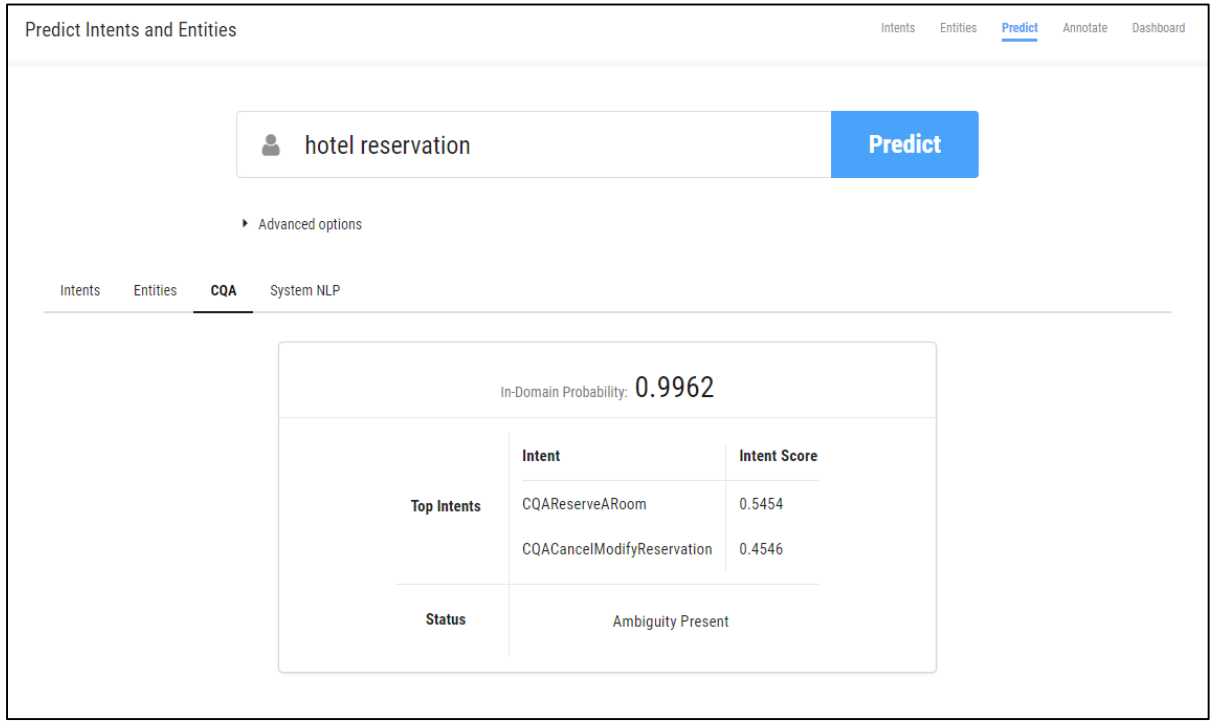


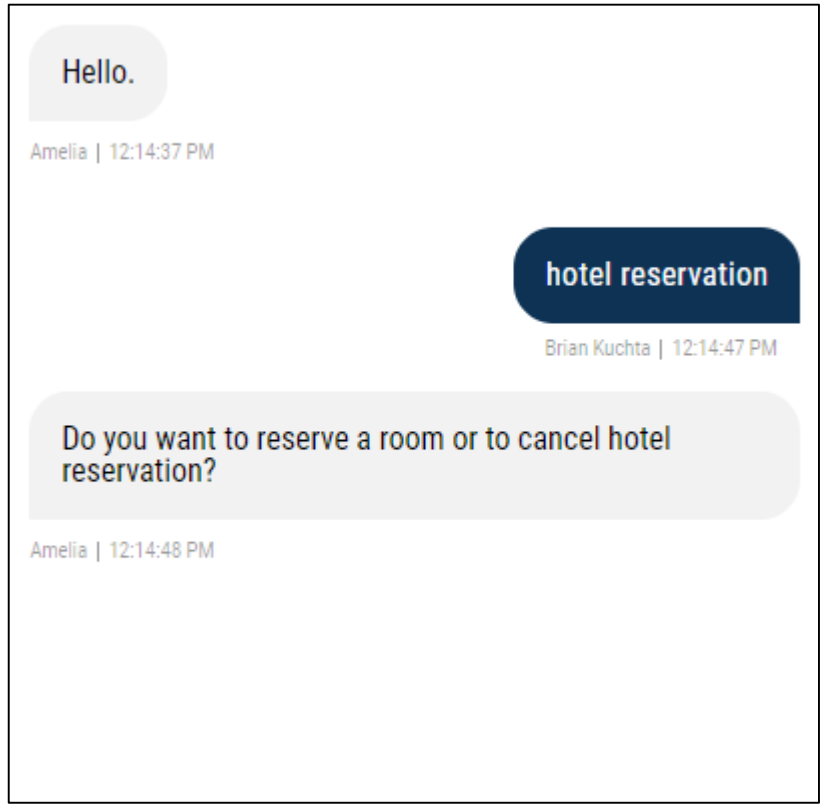


- Out-Of-Domain Threshold and Top-Two-Intent-Ambiguity threshold set in Dashboard -> Edit Cqa Params

Testing CQA in Predict

- Deploy both CQA models in Dashboard
- Enter test utterance in Predict
- Select CQA tab
- If in-domain and difference between top two intents is less than the **Top-Two-Intent-Ambiguity-Threshold** => AMBIGUITY exists

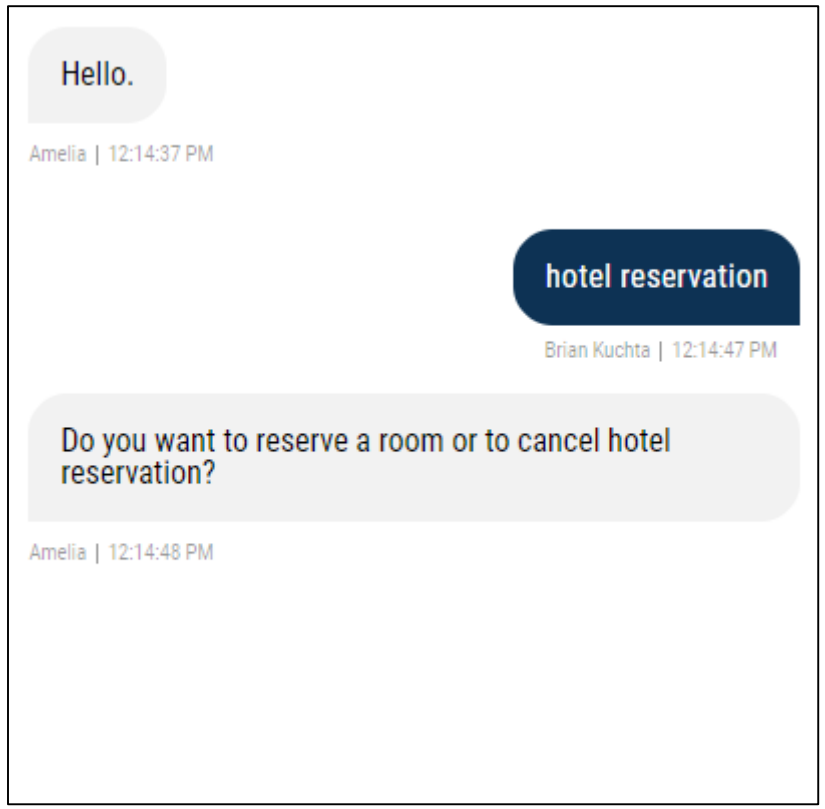


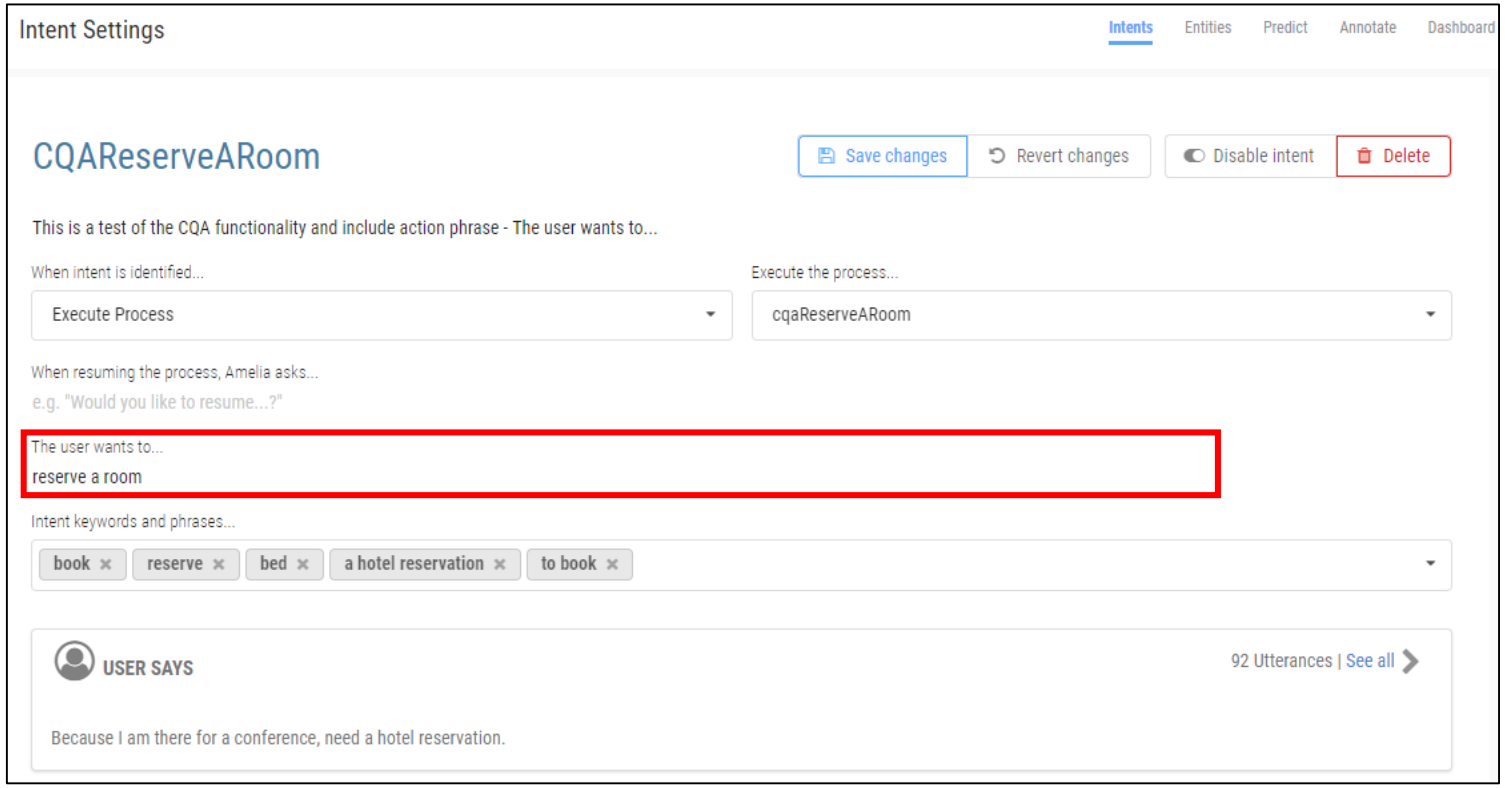


- In mind view, Amelia responds with a clarifying question

Modifying Clarifying Questions

- If ambiguity is detect, Amelia automatically responds with a clarifying question (i.e., a discriminative question)
  - Built from Intent keywords and phrases extracted during model training
  - Can be overwritten by specifying word/phrase in intent -> “The user wants to…”

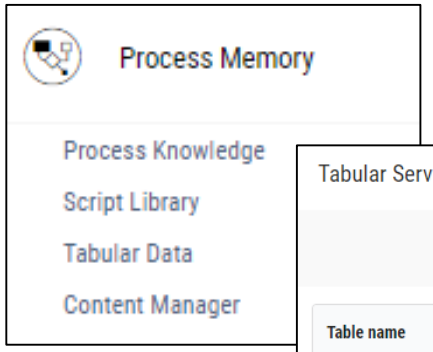


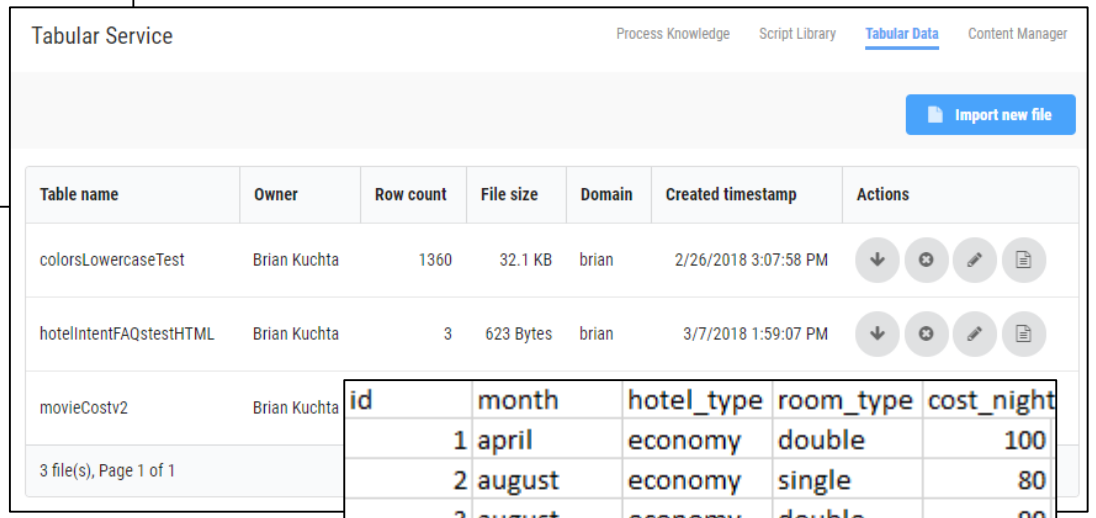


Tabular Data

Tabular Data

- Allows script tasks to query tabular data tables
  - Static data
- Comma-delimited format recommended
  - < 20 MB
  - Encoding must be UTF-8
  - File must have unique name
  - Column headers mandatory
  - Needs at least one row besides headers
  - Best practice to include ID as first column
  - Note: lowercase Strings are easier to work with



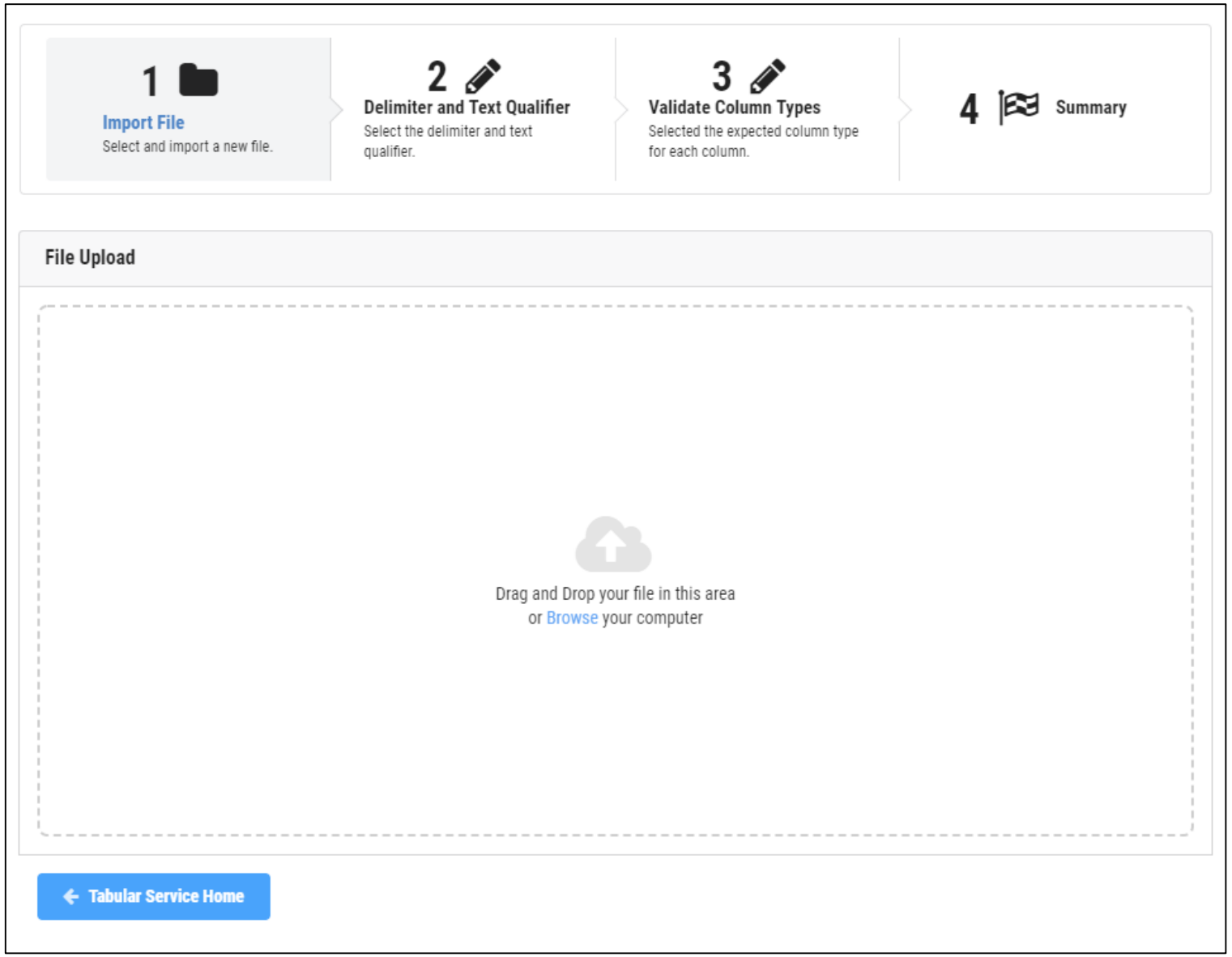


id	month	hotel_type	room_type	cost_night
1	april	economy	double	100
2	august	economy	single	80
3	august	economy	double	90
4	december	economy	single	75
5	february	economy	double	110
6	january	economy	single	95
7	january	economy	double	115
8	july	economy	single	85
9	june	economy	double	110
10	march	economy	single	83
11	march	economy	double	112
12	may	economy	single	89
13	november	economy	double	104
14	october	economy	single	67
15	october	economy	double	92
16	september	economy	single	84

- CSV file contents retrieved/stored in a data structure
- Data structure queried for relevant output
- Caching in place to enable faster querying of CSV files queried earlier

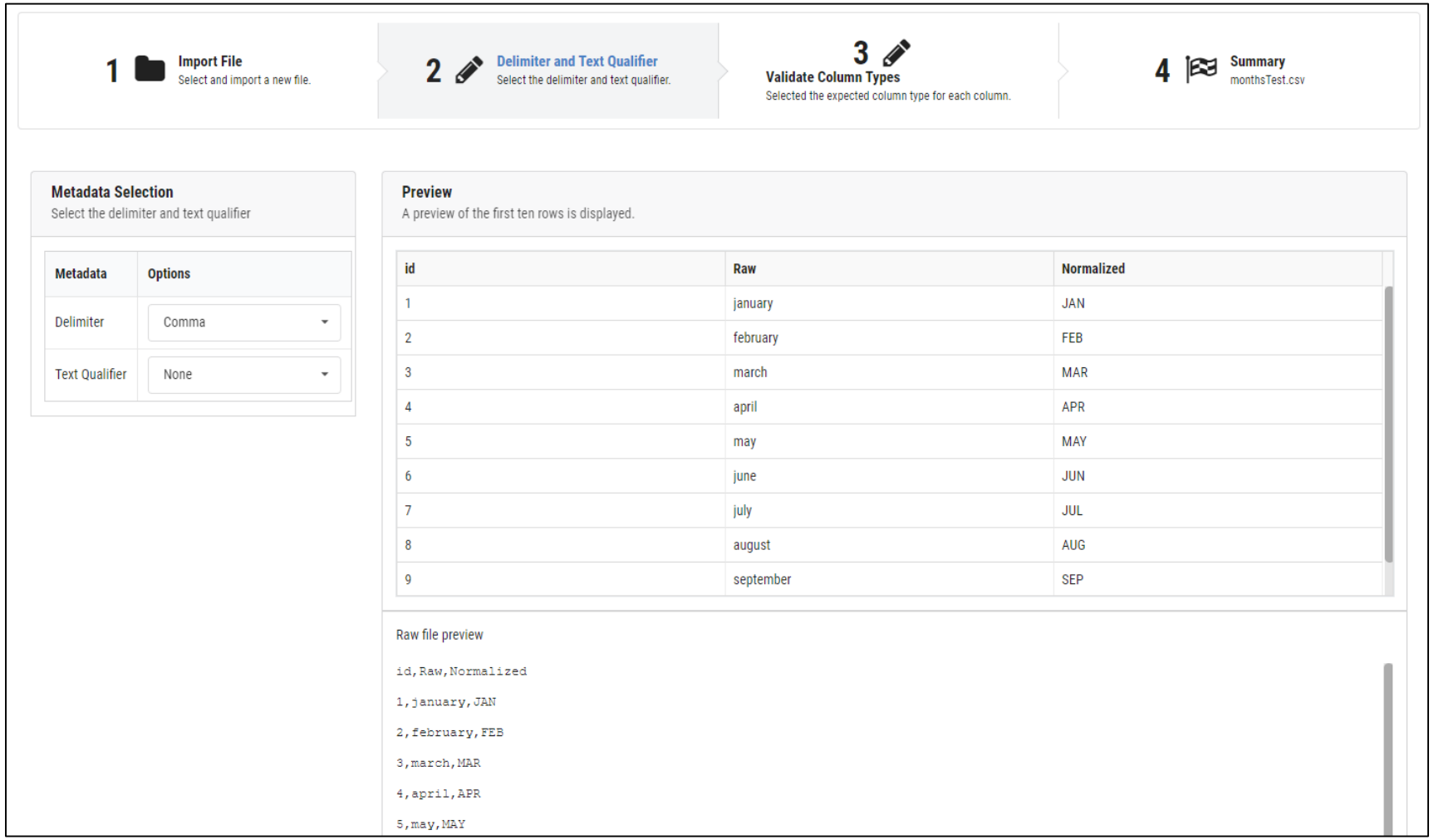
Tabular Data

Import the CSV file

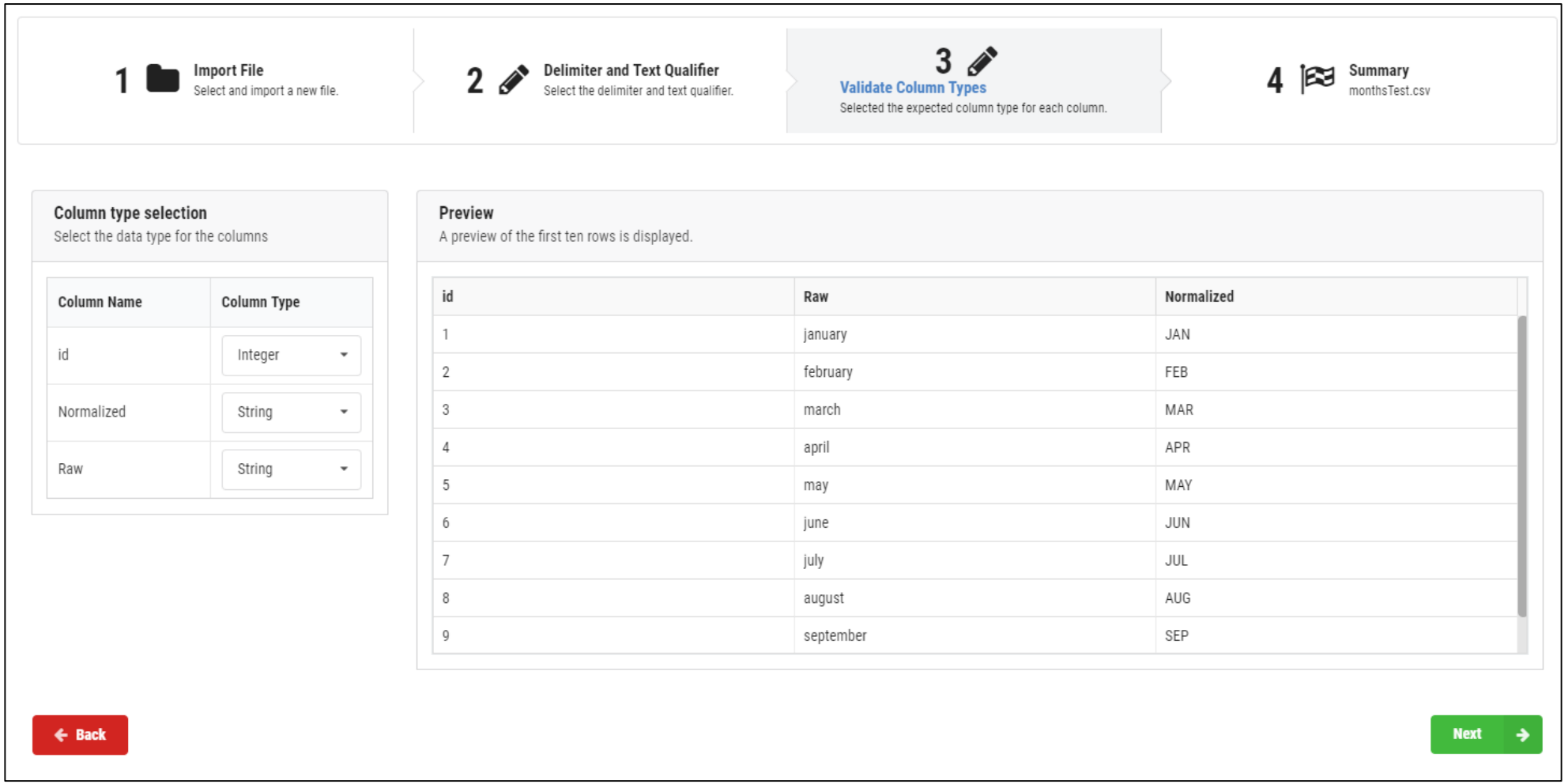


Tabular Data

Preview delimiter and text qualifier

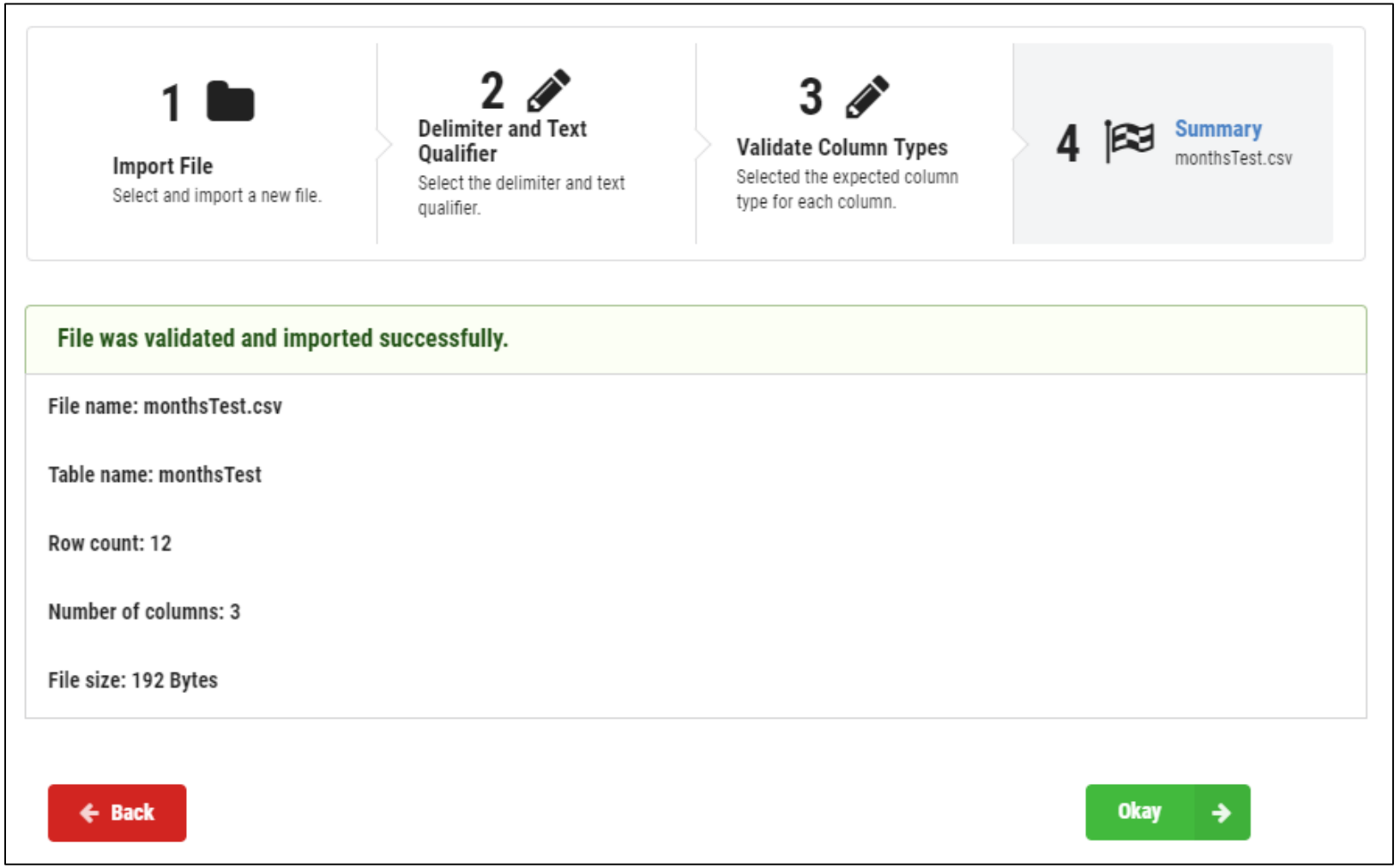


Validate Column Types



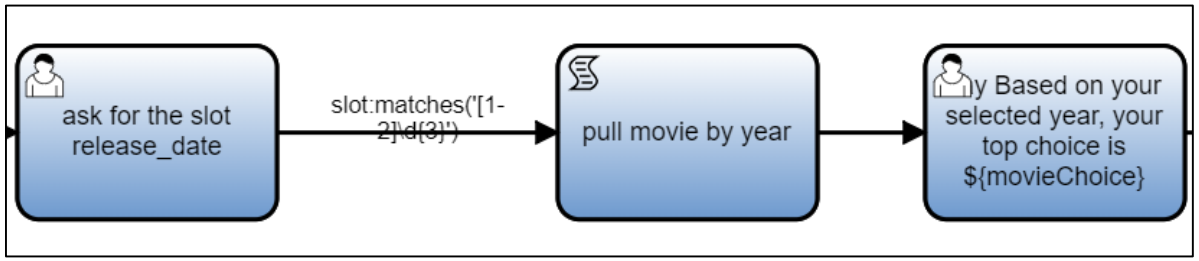
Tabular Data

Review Summary



Tabular Data Syntax

- tabularDataService.query('tableName', 'columnName1', 'columnName2', ...)
  - Empty parameter(s) throw error
  - Array notation used to access values provided by queries
  - Table name = file name (unless explicitly changed)
  - Must include columns from which values will be extracted as parameter




```
def releaseDateOne = contextService.latestSlot('release_date')

def formattedValueReleaseDate = quantityService.format(releaseDateOne,
    Formatters.INTEGER)

def movieChoice = tabularDataService.query('movieCostv2','release_date'
    ,'title','cost').is('release_date',formattedValueReleaseDate).first
    ()['title'];

execution.setVariable("movieChoice", movieChoice)
```

Tabular Data – Operators

is (columnName, value)
- Like equals operation
- Example: is('cost',formattedValueCost)

isNot (columnName, value)
- Like not equals operation
  Example: isNot('beds',2)

in (columnName, anyAdditionalParameters)
- Like IN operation in SQL -- minimum of two parameters
- All rows that satisfy at least one parameter returned
- Example: in('baths',1,1.5,2)

notIn (columnName, anyAdditionalParameters)
- Like NOT IN operation in SQL -- minimum of two parameters
- Example: notIn('release_date',2015,2016,2017)

isGt (columnName, value)
Example: isGt('baths',0)

isGte (columnName, value)
Example: isGte('baths',1)

isLt (columnName, value)
Example: isLt('cost',formattedValueCost)

isLte (columnName, value)
Example: isLte('area',1200)

like (columnName, value)
likeAny (columnName, values)
- ? -> single character
- %->multiple characters

Example: like('title','wond?r woman')
Example: like('title','wonder wom%')

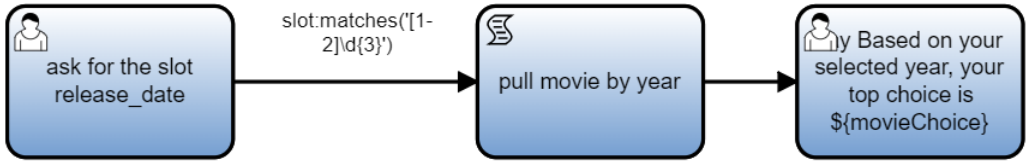
sortAsc(columnName)
sortDesc(columnName)
Example: sortDesc('cost')

```
def movieCost = contextService.latestSlot('cost')

def formattedValueCost = quantityService.format(movieCost, Formatters
    .INTEGER)

def movieChoice = tabularDataService.query('movieCostv1.2','release_date'
    ,'title','cost').is('cost',formattedValueCost).first()['title'];

execution.setVariable("movieChoice", movieChoice)
```



Tabular Data – Operators

first()['columnName']
last()['columnName']

- Returns a map representing the first/last row that satisfies query criteria
- Example: first()['title']
- Example: last()['release_date']

list operator

- Returns map of maps representing all rows that satisfy the query criteria
- Each inner map = row

firstN(n) ['columnName']
lastN(n) ['columnName']

- Returns a map of maps representing the first/last N rows that satisfy the query criteria
- Each inner map = row
- N = integer

index(N)

- Returns a map representing a row at index N
- N = integer

first(), last() and index(N) return one row:

...first()['column_name']
...last()['column_name']
...index(10)['column_name']

list(), firstN(n) and lastN(n) return a map of map <index_number, <column_name, column_value>:

...list()[index]
...firstN(10)[index]
...lastN(10)[index]

Value of a particular row:

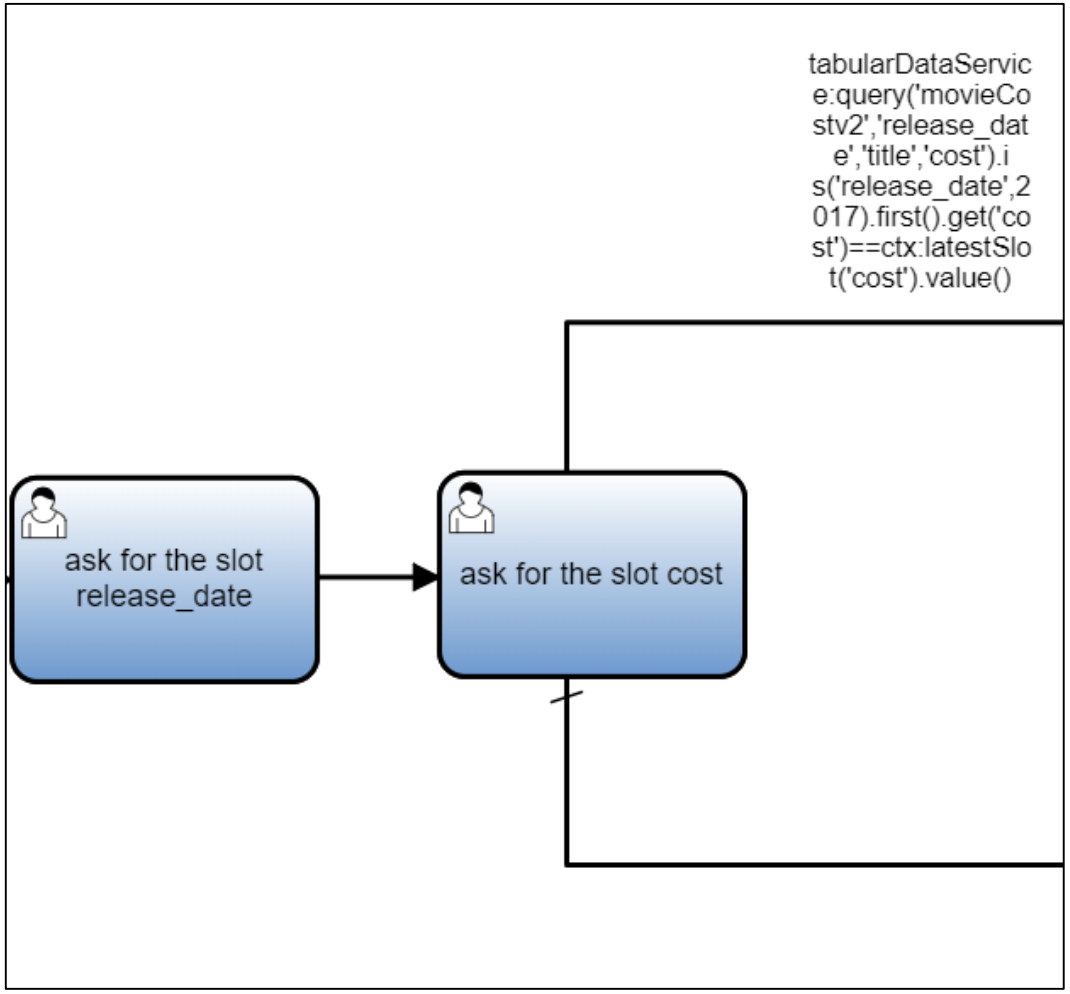
...list()[index]['column_name']
...firstN(10)[index]['column_name']
...lastN(10)[index]['column_name']

tabularDataService Prefix

- Used for testing data from query of tabular data and responses from end user

Syntax:

tabularDataService:query(‘tabularDataSetName',‘column1',‘release_date','cost').is('release_date',2017).first().get('cost')==ctx:latestSlot('cost').value()



Activity

movieId	release_date	title	cost
1	1971	Willy Wonka & the Chocolate Factory	5
2	2013	Frozen	8
3	1985	Back to the Future	5
4	1990	Home Alone	3
5	1992	The Mighty Ducks	6
6	1987	Dirty Dancing	7
7	2011	Drive	4
8	1993	Jurassic Park	8
9	1985	The Goonies	6
10	1997	Titanic	3

```
def movieCost = contextService.latestSlot('cost')

def formattedValueCost = quantityService.format(movieCost, Formatters
    .INTEGER)

def movieChoice = tabularDataService.query('movieCostv1.2','release_date'
    ,'title','cost').is('cost',formattedValueCost).first()['title'];

execution.setVariable("movieChoice", movieChoice)
```

Timing: 20 minutes
Validation: Mind View Testing

Tabular Data

- Import one of the tabular data sets provided or create one of your own.
- Query the data set in a BPN.
- Say that value in a BPN task.

Questions

AMELIA VEI

Thank You

Presented by
IPsoft Global Training and Development