API for Entity Services

Service Name	Capability	Who has access?	Input parameters	Process	Output
Member Entity Service	addMember (when the user registers as a WR member)	- user	- username - password - first_name - last_name - contact_number - email_address - address - zip_code - country - city	- Check for username/email address duplication - Check for password validation (password complexity) + validation for other fields - allocate a member_id for the new member	Return id of new user (member_id)
	getProfile	- member - employee (all)	Member: - username - password Employee: - member_id	- Check for whether username exists and if username and password match	Return the following: - username - first_name - last_name - phone - email_address - address - zip - country - city
	updateProfile	- member	- username - password - first_name - last_name - phone - email_address	- input validation check - Update database with new values	Return profile updated or not

			- address - zip - country - city		
	deleteMember (when member wants to deactivate account)	- member	- username - password	- Checks for authenticity by prompting user to enter password again - Checks for any pending payment If the password is correct and there is no pending payment, delete member from the database.	Return member deleted or not
	changePassword (member: when member wants to change password on his own, emloyee: when member forgets his password)	- member - employee (all)	User: - username - old_password - new_password Employee: - member_id	User: - Check if the old password is correct - Validation for new password Employee: - generate a password for the member and send an email to the member bearing the new password	Return password changed or not
	viewHistory	- member - employee (all)	User: - username - password Employee: - member_id	- Retrieve the records for the member_id in the videoQueue database	Returns history of rental made by the member
	viewRewardsPoints	- member	- username	- Retrieve from member	Return rewards points

			- password	database	
Video Entity Service	addVideo (adding a new video into the database)	- employee (purchasing executive)	- video_id	If the video_id exists Assign - video_quantity + 1 If the video_id does not exist, create a new video_id Assign - video_quantity == 1 - video_name - video_actors - video_description etc	Return video added or not
	getVideoDetails	- member - employee (purchasing executive)	- video_id	- Check whether the video_id exists - Retrieve details from the database based on the video_id if it exists	Return the following: - video_name - video_year - video_actors - video_description - video_quantity
	updateVideoDetails	- employee (purchasing executive)	- video_id - video_name - video_quantity - video_actors - video_description	Update database with new details where applicable	Return video updated or not
	deleteVideo (deleting the entire record of the video from the database)	- employee (purchasing executive)	- video_id	- Check if the video_id is in any user's video queue - If the video_id exists and is not in anyone's queue, delete that video_id from the database	Return video removed or not
Actor	addActor	- employee	- actor_name	- Update the database with a	Return actor added or

Entity Service		(purchasing executive)	- actor_DOB - actor_country - actor_city - actor_movies etc	new actor record	not
	getActorDetails	- member - employee (purchasing executive)	- actor_id	- Check whether the actor_id exists - Retrieve details from the database based on the actor_id if it exists	Return the following: - actor_name - actor_DOB - actor_country - actor_city - actor_movies etc
	updateActorDetails (to update details other than movies he acted in)	- employee (purchasing executive)	- actor_id - actor_name - actor_DOB - actor_country - actor_city - actor_movies etc	- Update database with new details where applicable	Return actor updated or not
	deleteActor (deleting the entire record of the actor from the database)	- employee (purchasing executive)	- actor_id	- If the actor_id exists, delete that actor_id from the database	Return actor removed or not
	addActorMovie (to add movies he acted in)	- employee (purchasing executive)	- actor_id - video_id	- Append video_id onto the list of actor_movies for that actor record	Return actor record updated or not
	removeActorMovie (to remove movies the actor has acted in)	- employee (purchasing executive)	- actor_id - video_id	- If the video_id exists in the list of actor_movies, Remove video_id from the list of actor_movies for that actor record	Return movie removed from actor's record or not
Employee Entity	addEmployee (to register a new	- employee (all)	- first_name - last_name	- Check for email address duplication	Return an assigned employee_id

Service	employee)		- contact_number - email_address - address - password - role	- Check for password validation (password complexity) + validation for other fields	
	getEmployeeDetails (to get employee's profile)	- employee (all)	- employee_id - password	Retrieve employee's details from employee database	Return the following: - first_name: first name of employee - last_name: last name of employee - role: employee role - address: address of employee - contact: contact number of employee - email: email of employee
	updateEmployeeDetails (to update things like contact number etc)	- employee (all)	- employee_id - password	- Input validation check - Update database with new values	Return employee detail updated or not
	deleteEmployee (to remove record of an employee who has resigned or is fired)	- employee (boss)	- employee_id - password	If employee exists, delete employee from database	Return employee deleted or not
	getEmployeeRole	- employee (all)	- employee_id - password	Retrieve from employee database	Return employee_role
	getAccessRights	- employee (all)	- employee_id - password	- If employee exists, get employee access rights	Return access_rights

	changePassword	- employee (all)	- email_address - old_password - new_password	- Validation check for new password	Return successful or not
Video Queue Entity Service	addVideoQueue (when the session starts, the member)	- member	- username - password	- Check if a video queue that is not checked out exists for that member If it does not exist, Create a new queue Assign and update video queue database with the following: - videoQueueId - dateCreated - userId: user id - contents: a list of videos; empty by default - totalAmountDue: $0 - videoQueueStatus: payment pending - dueDate: NULL by default - invoiceNumber: null Otherwise Display the existing video queue	Return videoQueueId allocated
	getVideoQueue	- employee (inventory coordinator) - member	- username - password - video_queue_id	If video queue exists for the user, display video queue	Return the following - contents: list of video id and titles of the videos in the queue - totalAmountDue: total price that user needs to pay - dateDue: the date

					that the user has to return the video if payment was made today
	removeVideoQueue (When user logs out of the session without making payment. Maybe can specify that the queue will remain up to a certain amount of time eg 1 hour)	- employee (inventory coordinator) - member	- username - password - video_queue_id	If videoQueue_id is not empty, add back the videos into the video database (by increasing corresponding video ids by 1) If invoiceNumber is NULL, Delete records for the video_queue_id from the database	Return video queue removed or not
	addVideoToQueue	- member	- username - password - video_queue_id - video_id	If the quantity of video with the videoId in the video database is greater than or equals to 1, and the quantity of videos in the video queue is smaller than 3, Add the video to the queue Update in the video queue: - contents - totalAmountDue Update in the video database: - quantity (decrease by 1) Otherwise display error “failed to add video”	Return video added to queue or not
	removeVideoFromQueue	- member	- username - password - video_id	If video_id exists in the video queue, Remove the video	Return video removed from queue or not

			- videoQueueId	Update in the video queue: - contents - totalAmountDue	
Comments Entity service	addComment	- member	- username - password - video_id	Allow user to add comment. Add comment to database, assign a comment_id	Return comment added or not
	editComment	- member	- username - password - comment_id	Allow user to edit a comment that he added. Update database with the change.	Return comment edited or not
	deleteComment	- member	- username - password - comment_id	Allow user to delete a comment that he added. Delete the entry from the database	Return comment deleted or not

API for Task Services

Service Name	Capability	Who has access?	Input parameters	Process	Output
Search	searchMember	- employee (inventory coordinator)	- username	Search within the member database for matches	Returns matches if available
	searchVideo	- employee (purchasing executive) - user (member or non-member)	- video name	Search within the video database for matches	Returns matches if available
	searchActor	- employee (purchasing	- actor name	Search within the actor database for matches	Returns matches if available

		executive) - user (member or non-member)			
	searchEmployee	- employee (all)	- employee name	Search within the employee database for matches	Returns matches if available
Video Rental	rentVideo	- member	- video_queue_id	Checks out the video queue and confirm the rental	Return success of no
	returnVideo	- employee (inventory coordinator)	- video_queue_id - video_ids	Updates record for video in video queue database: set status to “Returned” Increase quantity in the video database to add back the video	Return success of no
Stream Video	streamVideo	- member	username password video_id	Stream the video for the member	Return the direct link to the video
Payment	submit	- member	- username - password - video_queue_id - payment_detail	User has to enter the payment details eg credit card number, Paypal account info, etc.	Return successful or not Generate payment_id
Generate Receipt	submit	- member	- username - password - payment_id	Done after the payment, display payment details and videos rented	Return receipt's details Assign receipt_id Eg: - receipt_id - username - video queue contents - amount paid

					- date of payment
Generate Report	Submit (to generate monthly report – title performance summary)	- employee (sales executive)	- employee_id - password - date range (date start and date end)	Retrieve a list of entries from the video queue database which has invoiceNumber that is not NULL (indicates that the video queues have been paid for), for the specific date range. For each video queue, output the following: - date of payment - list of videos - amount paid	Return report details Assign report_id
Acknowledge comments	Retrieve	- employee (it executive)	- employee_id - password	Retrieve comments that have yet to be approved or rejected	Return list of unapproved comments
	approve	- employee it executive)	- employee_id - password - comment_id	Check against the comment and approve	Return successful or not
	reject	- employee (it executive)	- employee_id - password - comment_id	Check against the comment and reject	Return successful or not
Rewards Points	Allocate	- member	- username - password - payment_id	After payment is made, the system will calculate the amount of points to allocate for that user based on the payment_id, and update the database record.	Return successful or not
	deduct	- member	- username - password - claim_id (assigned to each free gift)	When the user claims his free gift, the system checks the amount to be deducted based on the claim_id, and how much	Return successful or not

				points the user has based on the user_id. If number of points to be deducted < points of the user, Deduct the points accordingly and notify the user that the free gift has been claimed and it will be mailed to his address.	
Obtain Help	Submit	- member - user (non-member)	Member: - username - password - question Non-member: - email address - question	The question will be sent to the WR's email.	Return successful or not
Update video status	sendSMS	- employee Inventory coordinator)	- employee_id - password - member mobile number - message	To update customer via SMS about the current status of the video (can be posted/pending postage)	Return successful or not
	sendEmail	- employee (inventory coordinator)	- employee_id - password - member email address - message	To update customer via email about the current status of the video (can be posted/pending postage)	Return successful or not

API for utility services

Service Name	Capability	Who has access?	Input parameters	Process	Output
Print (to set the reports in a printable/ printer-friendly manner)	PrintReceipt	- member	- receipt_id	Create a PDF file	Returns successful or not
	printReport	- employee (sales executive	- report_id	Create a PDF file	Returns successful or not
	printHistory	- employee (all) - member	Employee: - user_id Member: - username - password	Create a PDF file	Returns successful or not
Exception	NullPointer	- member (all) - employee - user	Any input	To catch exceptions	Return exception
	IndexOutOfBound	- member (all) - employee - user	Any input	To catch exceptions	Return exceptions
Validation	Validate Input	- employee (all) - member - user	Any input that needs to be validated	To ensure that an input is valid eg – password must be >8 characters etc	Returns true or false
Notification	Notify	- employee (all) - member - user		To show a user when an action is completed eg – payment made, video added to queue, etc	Returns notification page or pop-up