PART 2 SOLUTION

Text in red is for marker's information.

- You are looking for evidence of **logical** and **methodical** test design.
- As long as all columns are included, the format used in each column are irrelevant so long as you (as the tester) can easily execute the test (ie. set up the database in the system under test, send the HTTP request and verify the correctness of the response and subsequent data state). Exceptions to this are:
  - **Expected response**: should be correctly-formed JSON.
- The status codes shown below are the best options, but others are acceptable so long as they are sensible.
- It's okay if the "states" are shown in the main table, rather than abstracted into a key.

Notes

- All URLs are given from the root “/“.
- All Expected responses are given in JSON.
- Test cases have been kept to the absolute minimum to fully cover the spec, plus tests for disallowed methods and correct response to non-existant records.
- Preconditions and postconditions are given as one of the following states, representing the database setup on the system under test. The system should be in the state described by these tables before (for preconditions) and after (for postconditions) test execution.
- Edge cases have not been considered (but examples are provided for markers at the end).

State 0
No data in system

State 1

RESTAURANTS			
ID	**Name**	**City**	**URL Slug**
1	Joe's Diner	London	joes_diner

State 2

RESTAURANTS			
ID	**Name**	**City**	**URL Slug**
1	Joe's Diner	Leeds	joes_diner

State 3

RESTAURANTS			
ID	**Name**	**City**	**URL Slug**
1	Amy's Diner	London	amys_diner

State 4

RESTAURANTS			
ID	**Name**	**City**	**URL Slug**
1	Joe's Diner	London	joes_diner

REVIEWS					
ID	**Author**	**Published**	**Title**	**Rating**	**Text**
1	joe_bloggs	<Todays date>	Great Food!	5	The food was fantastic.

State 5

RESTAURANTS			
ID	**Name**	**City**	**URL Slug**
1	Joe's Diner	London	joes_diner

REVIEWS					
ID	**Author**	**Published**	**Title**	**Rating**	**Text**
1	jane_doe	<Todays date>	Terrible	1	The food was fantastic.

Required Test Cases

These test cases are the obvious tests based on knowledge of REST and the specification provided. This is also identical to an exercise done on launchpad and the example in the revision video.

Test No	Name	Preconditions	Method	URL	Request Params	Expected Status Code	Expected Response	Expected Postconditions	Notes and Assumptions
1	List all restaurants with none present.	State 0	GET	/restaurants	-	200 OK	[]	No change	

2	List all restaurants with one present.	State 1	GET	/restaurants	-	200 OK	[
							{
							id:1,
							name:"Joe'sDiner",
							city:"London",
							url_slug: "joes_diner",
							}
							]
								No change	

3	Get an individual restaurant that exists	State 1	GET	/restaurants/joes_diner	-	200 OK	{
							id:1,
							name:"Joe'sDiner",
							city:"London",
							url_slug: "joes_diner",
							}
								No change	

4	Get an individual restaurant that does not exist	State 0	GET	/restaurants/alis_diner	-	404 Not Found	No content (or suitable JSON error message)	No change	

5	Create a restaurant	State 0	POST	/restaurants	**name:** Joe's Diner
					city: London
					url_slug:
					joes_diner
						201 Created	{
							id:1,
							name:"Joe'sDiner",
							city:"London",
							url_slug: "joes_diner",
							}
								State 1	URL slug is auto-generated and not provided as a parameter.

6	Create a restaurant without a name	State 0	POST	/restaurants	**city:** London
					url_slug:
					joes_diner
						400 Bad Request	{
							error: "must provide a name"
							}
							(or similar)
								No change	URL slug is auto-generated and not provided as a parameter.

7	Create a restaurant without a city	State 0	POST	/restaurants	**name:** Joe's Diner
					url_slug:
					joes_diner
						400 Bad Request	{
							error: "must provide a city"
							}
							(or similar)
								No change	URL slug is auto-generated and not provided as a parameter.

8	Post to an individual restaurant	State 1	POST	/restaurants/joes_diner	-	405 Method Not Allowed	No content (or suitable JSON error message)	No change	

Test No	Name	Preconditions	Method	URL	Request Params	Expected Status Code	Expected Response	Expected Postconditions	Notes and Assumptions
9	Update the collection of restaurants	State 1	PATCH	/restaurants	-	405 Method Not Allowed	No content (or suitable JSON error message)	No change	

10	Update the name of a restaurant	State 1	PATCH	/restaurants/joes_diner	**city:** Leeds	200 OK	{
							id:1,
							name:"Joe'sDiner",
							city:"Leeds",
							url_slug: "joes_diner",
							}
								State 2	

11	Update the name of a restaurant	State 1	PATCH	/restaurants/joes_diner	**name:** Amy's Diner	200 OK	{
							id:1,
							name:"Amy's Diner",
							city:"London",
							url_slug: "amys_diner",
							}
								State 3	URL slug is auto-generated and not provided as a parameter.

12	Update a restaurant without providing parameters	State 1	PATCH	/restaurants/joes_diner	-	400 Bad Request	{
							error: "must provide fields to update"
							}
							(or similar)
								No change	

13	Update a restaurant that does not exist	State 0	PATCH	/restaurants/alis_diner	**name:** "Stuff"	404 Not Found	No content (or suitable JSON error message)	No change	
14	Delete the collection	State 1	DELETE	/restaurants	-	405 Method Not Allowed	No content (or suitable JSON error message)	No change	
15	Delete a restaurant	State 1	DELETE	/restaurants/joes_diner	-	204 No Content	No content	State 0	
16	Delete a restaurant that does not exist	State 1	DELETE	/restaurants/alis_diner	-	404 Not Found	No content (or suitable JSON error message)	No change	
	REVIEWS								
17	Get all reviews for a restaurant that has no reviews.	State 1	GET	/restaurants/joe_diner/reviews	-	200 OK	[]	No change	

Test No	Name	Preconditions	Method	URL	Request Params	Expected Status Code	Expected Response	Expected Postconditions	Notes and Assumptions

18	Get all reviews for a restaurant has a review.	State 4	GET	/restaurants/joe_diner/reviews	-	200 OK	[
							{
							id:1,
							name:"joe_bloggs",
							published:"<today date>",
							title: "Great food!",
							rating: 5,
							text: "The food was amazing"
							}
								No change	

19	Get a review that exists.	State 4	GET	/restaurants/joe_diner/reviews/1	-	200 OK	{
							id:1,
							name:"joe_bloggs",
							published:"<today date>",
							title: "Great food!",
							rating: 5,
							text: "The food was amazing"
							}
								No change	

20	Get a review that does not exist.	State 4	GET	/restaurants/joe_diner/reviews/2	-	404 Not Found	No content (or suitable JSON error message)	No change	

21	Create a review	State 1	POST	/restaurants/joe_diner/reviews	**author:** joe_bloggs
					title: Great food!
					rating: 5
						201 Created	{
							id:1,
							name:"joe_bloggs",
							published:"<today date>",
							title: "Great food!",
							rating: 5,
							text: "The food was amazing"
							}
								State 4	Assumes published at date and ID are auto-generated.

22	Create a review without an author	State 1	POST	/restaurants/joe_diner/reviews	**title:** Great food!
					rating: 5
						400 Bad Request	{
							error: "must provide author"
							}
							(or similar)
								No change	Assumes only author, title and rating are mandatory. So I don't have to repeat millions of tests - this is a

23	Create a review without a title	State 1	POST	/restaurants/joe_diner/reviews	**author:** joe_bloggs
					rating: 5
						400 Bad Request	{
							error: "must provide title"
							}
							(or similar)
								No change	

Test No	Name	Preconditions	Method	URL	Request Params	Expected Status Code	Expected Response	Expected Postconditions	Notes and Assumptions

24	Create a review without a rating	State 1	POST	/restaurants/joe_diner/reviews	**author:** joe_bloggs
					title: Great food!
						400 Bad Request	{
							error: "must provide rating"
							}
							(or similar)
								No change	reasonable assumption, though.

25	Post to an individual review	State 4	POST	/restaurants/joe_diner/reviews/1	-	405 Method Not Allowed	No content (or suitable JSON error message)	No change	
26	Update a collection of reviews	State 4	PATCH	/restaurants/joe_diner/reviews		405 Method Not Allowed	No content (or suitable JSON error message)	No change	

27	Update the author, title and rating of a review	State 4	PATCH	/restaurants/joe_diner/reviews/1	**author:** jane_doe
					titile: Terrible
					rating: 1
						200 OK	{
							id:1,
							name:"jane_doe",
							published:"<today date>",
							title: "Terrible",
							rating: 1,
							text: "The food was amazing"
							}
								State 5	Checks the 3 mandatory fields can be updated together, to save execution time. Verify the text remains unchanged.

28	Update a review without providing parameters	State 4	PATCH	/restaurants/joe_diner/reviews/1	-	400 Bad Request	{
							error: "must provide fields to update"
							}
							(or similar)
								No change	

29	Update a review that does not exist	State 4	PATCH	/restaurants/joe_diner/reviews/2	-	404 Not Found	No content (or suitable JSON error message)	No change	
30	Delete the review collection	State 4	DELETE	/restaurants/joe_diner/reviews	-	405 Method Not Allowed	No content (or suitable JSON error message)	No change	
31	Delete a review	State 4	DELETE	/restaurants/joe_diner/reviews	-	204 No Content	No content	State 1	
32	Delete a review that does not exist	State 4	DELETE	/restaurants/joe_diner/reviews/2	-	404 Not Found	No content (or suitable JSON error message)	No change	

Edge Case Tests

These test cases are some examples of less obvious tests - there are many others. Examples might include:

- *Using unusual record identifiers in URLs (particularly for reviews where the identifier is an integer) - very high or negative numbers.*
- *Repeating tests with multiple records on the system rather than single records.*
- *Tests for validating the format of parameters - since the spec doesn't specify any, these should have the assumptions listed.*

Test No	Name	Preconditions	Method	URL	Request Params	Expected Status Code	Expected Response	Expected Postconditions	Notes and Assumptions
x	Get all reviews for a restaurant that does not exist.	State 4	GET	/restaurants/alis_diner/reviews	-	404 Not Found	No content (or suitable JSON error message)	No change	

x	Create a review with text for a rating	State 1	POST	/restaurants/joe_diner/reviews	**author:** joe_bloggs
					title: Great food!
					rating: "Nonsense"
						400 Bad Request	{
							error: "Rating must be a number"
							}
							(or similar)
								No change	Assumes rating must be an integer.