*Florida International University*

*School of Computing and Information Sciences*

Software Engineering Focus

Feature Document

User Story #**221**

**Team Member:**

Pierre Jimenez

**Product Owner(s)**:

Mohsen Taheri

**Mentor(s)**:

Mohsen Taheri

**Instructor**: Masoud Sadjadi

**User Story** Create Backend Endpoints for Database

* As a programmer, I want to hit API endpoints so that I can access database information

Acceptance Criteria

* Must be able to request data from all database tables
* Must be able to post a new user
* Must be able to post to all model tables
* Must follow the database schema

**Use Case** #**221 – Create Backend Endpoints For Database**

Details:

Actor: Programmer

Preconditions:

1. Internet connection
2. Token Authentication for given tables

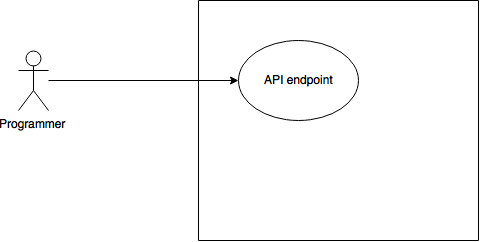
Description:

1. Programmer sends a POST request to API endpoint
2. API deserializes the JSON request object
3. API adds data to database
4. API creates a response for OK
5. API sends back response

Post-Condition:

1. Programmer gets back an OK status code.

**Use Case Diagram**



**Sequence Diagram**

APIRequests.png

**Class Diagram**

**Unit Test**

**Test Case 1 (Sunny Day)**

**Purpose**

* Login is OK with superuser

**Precondition**

* User in system

**Input**

* Hit login endpoint

**Expected Result**

* HTTP\_200\_OK

**Actual Result**

* HTTP\_200\_OK

**Test Case 2 (Rainy Day)**

**Purpose**

* Login is not-OK with non existent credentials

**Precondition**

* User not in system

**Input**

* Hit login endpoint

**Expected Result**

* HTTP\_400\_BAD\_REQUEST

**Actual Result**

* HTTP\_400\_BAD\_REQUEST

**Test Case 3 (Sunny Day)**

**Purpose**

* Test user can be inserted

**Precondition**

* None

**Input**

* Hit register-list endpoint

**Expected Result**

* HTTP\_200\_OK

**Actual Result**

* HTTP\_200\_OK

**Test Case 4 (Sunny Day)**

**Purpose**

* Test that user data can be fetched

**Precondition**

* User logged in

**Input**

* Hit user-detail endpoint

**Expected Result**

* HTTP\_200\_OK

**Actual Result**

* HTTP\_200\_OK

**Test Case 5 (Sunny Day)**

**Purpose**

* Test that fetched user data is correct

**Precondition**

* User logged in
* Data located in repository

**Input**

* Chatroom icon pressed on TabBar

**Expected Result**

* testData equals repository data

**Actual Result**

* testData equals repository data

**Test Case 6 (Sunny Day)**

**Purpose**

* Test that username and email can be updated

**Precondition**

* User logged in

**Input**

* PUT to user list endpoint

**Expected Result**

* testData equals repository data

**Actual Result**

* testData equals repository data

**Test Case 7 (Sunny Day)**

**Purpose**

* UserInfo can be created during register

**Precondition**

* User logged in

**Input**

* POST userInfo endpoint

**Expected Result**

* HTTP\_200\_OK

**Actual Result**

* HTTP\_200\_OK

**Test Case 8 (Sunny Day)**

**Purpose**

* userInfo can be gotten and is correct

**Precondition**

* User logged in
* userInfo data in repository

**Input**

* Hit GET userInfo endpoint

**Expected Result**

* testData equals repository data

**Actual Result**

* testData equals repository data

**Test Case 9 (Sunny Day)**

**Purpose**

* Test user info can be changed

**Precondition**

* User logged in
* userInfo data in repository

**Input**

* Hit PUT userInfo endpoint

**Expected Result**

* getData equals repository data

**Actual Result**

* getData equals repository data

**Test Case 10 (Sunny Day)**

**Purpose**

* Test School created successfully

**Precondition**

* User in system

**Input**

* Hit POST School endpoint

**Expected Result**

* HTTP\_201\_CREATED

**Actual Result**

* HTTP\_201\_CREATED

**Test Case 11 (Sunny Day)**

**Purpose**

* Test School gotten successfully

**Precondition**

* User in system
* School data in repository

**Input**

* Hit GET school endpoint

**Expected Result**

* School data equal to repository data

**Actual Result**

* School data equal to repository data

**Test Case 12 (Sunny Day)**

**Purpose**

* Test School can be updated successfully

**Precondition**

* User in system
* School data in repository

**Input**

* Hit PUT school endpoint

**Expected Result**

* Test data equal to repository data

**Actual Result**

* Test data equal to repository data

**Test Case 13 (Sunny Day)**

**Purpose**

* Test school roster can be created

**Precondition**

* User in system

**Input**

* Hit POST schoolRoster endpoint

**Expected Result**

* HTTP\_200\_OK

**Actual Result**

* HTTP\_200\_OK

**Test Case 14 (Sunny Day)**

**Purpose**

* Test that school roster data can be fetched

**Precondition**

* User logged in
* SchoolRoster in repository

**Input**

* Hit GET schoolRoster endpoint

**Expected Result**

* SchoolRoster data equal to repository data

**Actual Result**

* SchoolRoster data equal to repository data

**Test Case 15 (Sunny Day)**

**Purpose**

* Test SchoolRoster can be updated successfully

**Precondition**

* User in system
* SchoolRoster data in repository

**Input**

* Hit PUT school endpoint

**Expected Result**

* Test data equal to repository data

**Actual Result**

* Test data equal to repository data

**Test Case 16 (Sunny Day)**

**Purpose**

* Test class can be created

**Precondition**

* User in system

**Input**

* Hit POST class endpoint

**Expected Result**

* HTTP\_200\_OK

**Actual Result**

* HTTP\_200\_OK

**Test Case 17 (Sunny Day)**

**Purpose**

* Test that class data can be fetched

**Precondition**

* User logged in
* Class in repository

**Input**

* Hit GET class endpoint

**Expected Result**

* Class data equal to repository data

**Actual Result**

* Class data equal to repository data

**Test Case 18 (Sunny Day)**

**Purpose**

* Test class can be updated successfully

**Precondition**

* User in system
* Class data in repository

**Input**

* Hit PUT class endpoint

**Expected Result**

* Test data equal to repository data

**Actual Result**

* Test data equal to repository data

**Test Case 19 (Sunny Day)**

**Purpose**

* Test classRoster can be created

**Precondition**

* User in system

**Input**

* Hit POST classRoster endpoint

**Expected Result**

* HTTP\_200\_OK

**Actual Result**

* HTTP\_200\_OK

**Test Case 20 (Sunny Day)**

**Purpose**

* Test that classRoster data can be fetched

**Precondition**

* User logged in
* ClassRoster in repository

**Input**

* Hit GET classRoster endpoint

**Expected Result**

* ClassRoster data equal to repository data

**Actual Result**

* ClassRoster data equal to repository data

**Test Case 21 (Sunny Day)**

**Purpose**

* Test classRoster can be updated successfully

**Precondition**

* User in system
* ClassRoster data in repository

**Input**

* Hit PUT classRoster endpoint

**Expected Result**

* Test data equal to repository data

**Actual Result**

* Test data equal to repository data

**Test Case 22 (Sunny Day)**

**Purpose**

* Test assignments can be created

**Precondition**

* User in system

**Input**

* Hit POST assignments endpoint

**Expected Result**

* HTTP\_200\_OK

**Actual Result**

* HTTP\_200\_OK

**Test Case 23 (Sunny Day)**

**Purpose**

* Test that assignments data can be fetched

**Precondition**

* User logged in
* Assignments in repository

**Input**

* Hit GET assignments endpoint

**Expected Result**

* Assignments data equal to repository data

**Actual Result**

* Assignments data equal to repository data

**Test Case 24 (Sunny Day)**

**Purpose**

* Test assignments can be updated successfully

**Precondition**

* User in system
* Assignments data in repository

**Input**

* Hit PUT assignments endpoint

**Expected Result**

* Test data equal to repository data

**Actual Result**

* Test data equal to repository data

**Test Case 25 (Sunny Day)**

**Purpose**

* Test tasks can be created

**Precondition**

* User in system

**Input**

* Hit POST tasks endpoint

**Expected Result**

* HTTP\_200\_OK

**Actual Result**

* HTTP\_200\_OK

**Test Case 26 (Sunny Day)**

**Purpose**

* Test that tasks data can be fetched

**Precondition**

* User logged in
* Tasks in repository

**Input**

* Hit GET tasks endpoint

**Expected Result**

* Tasks data equal to repository data

**Actual Result**

* Tasks data equal to repository data

**Test Case 27 (Sunny Day)**

**Purpose**

* Test tasks can be updated successfully

**Precondition**

* User in system
* Tasks data in repository

**Input**

* Hit PUT tasks endpoint

**Expected Result**

* Test data equal to repository data

**Actual Result**

* Test data equal to repository data