CS 1530 - Group 4

Milestone 5 for CodeFeed

Alec, Tim, Jeremy, Joey, and Bert

# 1 Purpose

## This test plan is to test a social media interface CodeFeed. The testing will consist of front-end and back-end, focusing on the back-end components and the front-end interfaces. We will test the front-end with Katalon, and test the back-end with python unittest import library.

# 2 Unit Testing

### 2.1 App Config: the configuration file varies based on if the app is in debug, release, or testing mode

#### 2.1.1 Equivalence classes for the App Config module

##### 2.1.1.1 The equivalence classes for item “DEBUG”

###### 1 If the app is in release mode, the DEBUG flag will be set to false

###### 2 If the app is in debug mode, the DEBUG flag will be set to true

###### 3 If the app is in testing mode, the DEBUG flag will be set to true

###### 4 Anything else is an error

##### 2.1.1.2 The equivalence classes for item “TESTING”

###### 1 If the app is in release mode, the TESTING flag will be set to false

###### 2 If the app is in debug mode, the TESTING flag will be set to false

###### 3 If the app is in testing mode, the TESTING flag will be set to true

###### 4 Anything else is an error

##### 2.1.1.3 The equivalence classes for item “SECRET\_KEY”

###### 1 The SECRET\_KEY flag will always be set to binascii.hexlify(os.urandom(24)) regardless of the program mode

###### 2 Anything else is an error

##### 2.1.1.4 The equivalence classes for item “USERNAME”

###### 1 If the app is in release mode, the USERNAME flag will be set to 'admin'

###### 2 If the app is in debug mode, the USERNAME flag will be set to 'debug'

###### 3 If the app is in testing mode, the USERNAME flag will be set to 'testing'

###### 4 Anything else is an error

##### 2.1.1.5 The equivalence classes for item “SQLALCHEMY\_TRACK\_MODIFICATIONS”

###### 1 If the app is in release mode, the SQLALCHEMY\_TRACK\_MODIFICATIONS flag will be set to false

###### 2 If the app is in debug mode, the SQLALCHEMY\_TRACK\_MODIFICATIONS flag will be set to false

###### 3 If the app is in testing mode, the SQLALCHEMY\_TRACK\_MODIFICATIONS flag will be set to true

###### 4 Anything else is an error

##### 2.1.1.6 The equivalence classes for item “SQLALCHEMY\_DATABASE\_URI”

###### 1 If the app is in release mode, the SQLALCHEMY\_DATABASE\_URI flag will be set to ‘sqlite:///codefeed\_release.db'

###### 2 If the app is in debug mode, the SQLALCHEMY\_DATABASE\_URI flag will be set to sqlite:///codefeed\_debug.db'

###### 3 If the app is in testing mode, the SQLALCHEMY\_DATABASE\_URI flag will be set to sqlite:///codefeed\_testing.db'

###### 4 Anything else is an error

#### 2.1.2 Checklist: the following grey box tests will be performed

##### Load the app in debug mode and assert all variables are correct

##### Load the app in testing mode and assert all variables are correct

##### Load the app in release mode and assert all variables are correct

### 2.2 login\_required: used to assert the user is logged in before attempting to view a page or sensitive data

#### 2.2.1 Equivalence classes for the login\_required module

##### 2.2.1.1 The equivalence classes for item “session['user\_id']”

###### 1 If the session variable is nothing, the user will be redirected to the login page

###### 2 If the session variable is set, the user will continue to the requested page

###### 3 Anything else is an error

#### 2.2.2 Checklist: the following black box tests will be performed

##### Login to CodeFeed and attempt to view the profile page

##### Logout of CodeFeed and attempt to view the profile page

##### Login to CodeFeed and attempt to view the profile/messages page

##### Logout of CodeFeed and attempt to view the profile/messages page

##### Login to CodeFeed attempt to send a friend request

##### Logout of CodeFeed and attempt to send a friend request

##### Login to CodeFeed and attempt to add a category

##### Logout of CodeFeed and attempt to add a category

##### Login to CodeFeed and attempt to add a thread

##### Logout of CodeFeed and attempt to add a thread

##### Login to CodeFeed and attempt to add a comment

##### Logout of CodeFeed and attempt to add a comment

##### Login to CodeFeed and attempt to vote on a thread

##### Logout of CodeFeed and attempt to vote on a thread

##### Log in CodeFeed and attempt to vote on a comment

##### Logout of CodeFeed and attempt to vote on a comment

### 2.3 initdb: used to create an empty database for the application to use or clear an existing database

#### 2.3.1 Equivalence classes for initdb

##### 2.3.1.1 The equivalence classes for item “release database”

###### 1 If the application is set in release mode, a new database called “codefeed\_release.db” will be created

###### 2 Anything else is and error

##### 2.3.1.2 The equivalence classes for item “debug database”

###### 1 If the application is set in debug mode, a new database called “codefeed\_debug.db” will be created

###### 2 Anything else is an error

##### 2.3.1.3 The equivalence classes for item “testing database”

###### 1 If the application is set in testing mode, a new database called “codefeed\_testing.db” will be created

###### 2 Anything else is an error

##### 2.3.1.4 The equivalence classes for item “database exists”

###### 1 Even if the database already exists, it will still be blank after the module is executed

###### 2 Anything else is an error

#### 2.3.2 Checklist: the following white box tests will be performed

##### Run init\_db in debug mode

##### Run init\_db in testing mode

##### Run init\_db in release mode

### 2.4 Landing Page: used to introduce the platform to the user

#### 2.4.1. Equivalence classes for the Landing Page

##### 2.4.1.1 The equivalence classes for item "Create Account"

###### 1. on click, redirect to Register Page, acceptable

###### 2. anything else is an error

##### 2.4.1.2 The equivalence classes for item "Login":

###### 1. on click, redirect to Login Page, acceptable

###### 2. Anything else is an error

##### 2.4.1.3 The equivalence classes for item "Registered Users":

###### 1. If the number of registered users in the database is 0, display 0 acceptable

###### 2. Else display the count of registered users acceptable

###### 3. Anything else is an error

##### 2.4.1.4 The equivalence classes for item "Current Groups":

###### 1. If the number of categories in the database is 0, display 0 acceptable

###### 2. Else display the count of categories acceptable

###### 3. Anything else is an error

##### 2.4.1.5 The equivalence classes for item "Posts Made":

###### 1. If the number of threads + the number of comments in the database is 0, display 0 acceptable

###### 2. Else number of threads + the number of comments acceptable

###### 3. Anything else is an error

##### 2.4.1.6 The equivalence classes for item "Happy Programmers":

###### 1. If the number of votes in the database is 0, display 0 acceptable

###### 2. Else display the count of votes acceptable

###### 3. Anything else is an error

#### 2.4.2 Checklist: the following black box tests will be performed

##### Assert page loads elements correctly

##### Click on Create Account

##### Click on Login

##### Assert Registered Users count when database is empty

##### Assert Registered Users count when database is not empty

##### Assert Current Groups count when database is empty

##### Assert Current Groups count when database is not empty

##### Assert Posts Made count when database is empty

##### Assert Posts Made count when database is not empty

##### Assert Happy Programmers count when database is empty

##### Assert Happy Programmers count when database is not empty

### 2.5 Register Page: used to log the user into the CodeFeed application

#### 2.5.1 Equivalence classes for the Register Page

##### 2.5.1.1. The equivalence classes for item "Return to Login":

###### 1. redirect to Login Page acceptable

###### 2. anything else error

##### 2.5.1.2. The equivalence classes for item "Full Name":

###### 1. Combination of 1 or more letters, spaces, apostrophes, and hyphens not exceeding 64 characters acceptable

###### 2. anything else error

##### 2.5.1.3. The equivalence classes for item "Email Address":

###### 1. xxx@xxx.xxx, not exceeding length of 64 acceptable

###### 2. anything else error

##### 2.2.1.4. The equivalence classes for item "Username":

###### 1. string of 1-32 letters acceptable

###### 2. string of at least 1-32 numbers acceptable

###### 3. String composed of at least 1-32 symbols acceptable

###### 4. String composed of at least 1-32 alphanumeric/other characters acceptable

###### 5. String does not match any username entry in database acceptable

###### 6. anything else error

##### 2.2.1.5. The equivalence classes for item "Password":

###### 1. string of 8-64 characters acceptable

###### 2. Anything else error

##### 2.2.1.6. The equivalence classes for item "Confirm Password":

###### 1. string is the same as Password acceptable

###### 2. Anything else error

#### 2.5.2 Checklist: the following black box tests will be performed

##### Click on Return to Login

##### Fill out Full Name with a string of letters only

##### Fill out Full Name with a string of numbers only

##### Fill out Full Name with alphanumeric characters

##### Fill out Full Name with non-alphanumeric characters and symbols

###### Fill out Full name with 1 character

###### Fill out Full Name with 64 characters

###### Fill out Full Name with greater than 64 characters

##### Fill out Email Address with a string of valid characters

##### Fill out Email Address with a string of invalid characters

##### Fill out Username with numbers only

##### Fill out Username with alphanumeric characters

##### Fill out Username with non-alphanumeric characters and symbols

##### Fill out Password a string less than 8 characters

##### Fill out Password a string equal to 8 characters

##### Fill out Password a string greater than 8 characters

##### Fill out Verify Password with a string that matches Password

##### Fill out Verify Password with a string that does not match Password

### 2.6 Logout Route: used to log the user out of the CodeFeed application

#### 2.6.1 Equivalence classes for the Logout Route module

##### 2.6.1.1 Class “user logged in”

###### 1 If the user is logged in, the session will be cleared and they will be redirected to the login page

###### 2 Anything else is a failure

##### 2.6.1.2 Class “user not logged in”

###### 1 If the user is not logged in, they will be redirected to the login page

###### 2 Anything else is a failure

#### 2.6.2 Checklist: the following black box tests will be performed

###### Log in to CodeFeed, then invoke /logout route

###### Log out of CodeFeed, then invoke /logout route

### 2.7 Login Page: used to log the user into the CodeFeed application

#### 2.7.1. Equivalence classes for the Login Page":

##### 2.3.1.1. The equivalence classes for item "Click Here to Register":

1. redirect to Login Page, acceptable

2. anything else error

##### 2.3.1.2. The equivalence classes for item "Username":

1. string that matches the username in database acceptable

2. Anything else error

##### 2.3.1.3. The equivalence classes for item "Password":

1. string that matches the password for the username’s password in database acceptable

2. Anything else error

##### 2.3.1.4. The equivalence classes for item "Login to CodeFeed":

1. redirect to Profile Page, acceptable

2. Anything else error

#### 2.7.2 Checklist: The following black-box tests will be performed

##### Click register

##### Enter username and password that do not match

##### Enter username and password that do match

### 2.8 Profile Page: used to show the user their basic information and edit the information as well

2.4.1. Equivalence classes for the Profile Page

2.4.1.1. The equivalence classes for item "name":

1. check for correct name acceptable

2. Anything else error

2.4.1.2. The equivalence classes for item "username":

1. check for correct username acceptable

2. Anything else error

2.4.1.3. The equivalence classes for item "biography":

1. check for correct biography acceptable

2. Anything else error

2.4.1.4. The equivalence classes for item "creation\_date":

1. check for correct creation\_date acceptable

2. Anything else error

2.4.1.5. The equivalence classes for item "Friends":

1. redirect to Friends Page, acceptable

2. Anything else error

2.4.1.6. The equivalence classes for item "Edit Profile":

1. pop up new registration form in same page, acceptable

2. Anything else error

2.4.1.6.1. The equivalence classes for item "Name":

1. Combination of letters, spaces, apostrophes, and hyphens acceptable

2. anything else error

2.4.1.6.2. The equivalence classes for item "Email Address":

1. xxx@xxx.xxx acceptable

2. anything else error

2.4.1.6.3. The equivalence classes for item "Password":

1. string of 8 or more characters acceptable

2. Anything else error

2.4.1.6.4 The equivalence classes for item "Confirm Password":

1. string is the same as Password acceptable

2. Anything else error

2.4.1.6.5 The equivalence classes for item "Bio":

1. string of any size and any character(s) acceptable

2. Anything else error

2.4.1.6.6 The equivalence classes for item "Update Profile":

1. update my Profile, acceptable

2. anything else error

2.4.1.7. The equivalence classes for item "email":

1. check for correct email acceptable

2. open an email terminal acceptable

3. Anything else error

2.4.1.8. The equivalence classes for item "My Profile":

1. redirect to current user Profile, acceptable

2. Anything else error

2.4.1.9. The equivalence classes for item "Messages":

1. redirect to Messages Page, acceptable

2. Anything else error

2.4.1.10. The equivalence classes for item "Categories":

1. redirect to Categories Page, acceptable

2. Anything else error

2.4.1.11. The equivalence classes for item "Logout":

1. redirect to Login Page, acceptable

2. Anything else error

2.4.2. Equivalence classes for the Profile Page (profile/messages):

2.4.2.1. The equivalence classes for item "New Message":

1. Pop-up Compose Message form acceptable

2. Anything else error

2.4.2.1.1. The equivalence classes for item "Send To":

1. Username in database acceptable

2. Anything else error

2.4.2.1.2. The equivalence classes for item "Message Body":

1. String between 1 and 256 characters acceptable

2. Anything else error

2.4.2.1.3. The equivalence classes for item "Send Message":

1. Send message to other existing username acceptable

2. Send message to own username error

3. Anything else error

2.4.2.2. The equivalence classes for item "My Profile":

1. redirect to current user Profile, acceptable

2. Anything else error

2.4.2.3. The equivalence classes for item "Messages":

1. redirect to Messages Page, acceptable

2. Anything else error

2.4.2.4. The equivalence classes for item "Categories":

1. redirect to Categories Page, acceptable

2. Anything else error

2.4.2.5. The equivalence classes for item "Logout":

1. redirect to Login Page, acceptable

2. Anything else error

2.4.4. Checklist: The following black-box tests will be performed.

( ) Click on Friends.

( ) Click on Edit Profile, then click x.

( ) Click Edit Profile, then click outside the field, then click on Edit Profile again.

( ) Click on Edit Profile, Fill out 5 entries including 1 faulty field and click on Update. (For each respective text entry field)

( ) Click on Edit Profile, Fill out all entries correctly and click on Update

( ) Click on email.

( ) Click on My Profile.

( ) Click on Messages.

( ) Click on Categories.

( ) Click on Logout.

### 2.10 Profile/Friends Page: used to show the user their friend list and accept incoming requests

2.5.1. Equivalence classes for the Friends Page(profile/friends):

2.5.1.1. The equivalence classes for item "Friend Requests":

1. pop up friend request list in same page, acceptable

2. anything else error

2.5.1.1.1. The equivalence classes for item "Accept Request":

1. add friend object to list, acceptable

2. anything else error

2.5.1.1.2. The equivalence classes for item "request.user\_id":

1. redirect to Profile Page (User ID), acceptable

2. anything else error

2.5.1.2. The equivalence classes for item "Add Friend":

1. pop up Add Friend form in same page, acceptable

2. Anything else error

2.5.1.2.1. The equivalence classes for item "Add User":

1. string of at least 1 letter acceptable

2. string of at least 1 number acceptable

3. String composed of at least 1 symbol acceptable

4. String composed of at least 1 alphanumeric/other characters acceptable

5. String matches username entry in database acceptable

6. String matches your own username error

6. Anything else error

2.5.1.2.2. The equivalence classes for item "Add":

1. send request to user\_id acceptable

2. anything else error

2.5.1.3. The equivalence classes for item "Compose Message":

1. pop up Compose Message form in same page acceptable

2. Anything else error

2.5.1.3.1. The equivalence classes for item "Send To":

1. string of at least 1 letter acceptable

2. string of at least 1 number acceptable

3. String composed of at least 1 symbol acceptable

4. String composed of at least 1 alphanumeric/other characters acceptable

5. String matches username entry in database acceptable

6. Anything else error

2.5.1.3.2. The equivalence classes for item "Message Body":

1. string of at least 1 character acceptable

2. anything else error

2.5.1.3.3. The equivalence classes for item "Send Message":

1. send message to user\_id, acceptable

2. anything else error

2.5.1.4. The equivalence classes for item "Show Entries":

1. drop list showing 10, 25, 50, 100 acceptable

2. Click 10 acceptable

3. Click 25 acceptable

4. Click 50 acceptable

5. Click 100 acceptable

6. Anything else error

2.5.1.5. The equivalence classes for item "Search":

1. string matches name in database acceptable, (search items according to input)

2. Anything else error

2.5.1.6. The equivalence classes for item "Previous":

1. go to previous page of friends when number of user’s friends exceeds the entries shown acceptable,

2. Anything else error

2.5.1.7. The equivalence classes for item "1,2,3,..., x,...":

1. go to specified page of friends when number of user’s friends exceeds the entries shown acceptable

2. Anything else error

2.5.1.8. The equivalence classes for item "Next":

1. go to next page of friends when number of user’s friends exceeds the entries shown acceptable

2. Anything else error

2.5.1.9. The equivalence classes for item "Friend Table":

1. Empty Database should have nothing in friends table, no date, page is of size 1 acceptable.

2. If database contains entries for the user’s friends should be friends displayed as table entries with their respective fields acceptable.

3. Anything else error

2.5.1.9.1. The equivalence classes for item "Friend Table Name":

1. sort name by ascending/descending acceptable

2. Anything else error

2.5.1.9.2. The equivalence classes for item "Friend Table Friends Since":

1. sort date by ascending/descending acceptable

2. Anything else error

2.5.1.9.3. The equivalence classes for item "Friend Table Friend name/username":

1. redirect to Profile Page (User ID) acceptable

2. Anything else error

2.5.1.10. The equivalence classes for item "My Profile":

1. redirect to current user Profile acceptable,

2. Anything else error

2.5.1.11. The equivalence classes for item "Messages":

1. redirect to Messages Page acceptable

2. Anything else error

2.5.1.12. The equivalence classes for item "Categories":

1. redirect to Categories Page acceptable

2. Anything else error

2.5.1.13. The equivalence classes for item "Logout":

1. redirect to Login Page acceptable

2. Anything else error

2.5.1.14. The equivalence classes for item "Friend Requests":

1. Popup “Accept Friend Request” form box acceptable

2. Anything else error

2.5.1.14.1 The equivalence classes for item “Accept Request”:

1. Add {{username}} to friends acceptable

2. Anything else error

2.5.1.14.2 The equivalence classes for item “Done”:

1. Close the popup form box acceptable

2. Anything else error

2.5.2. Checklist: The following black-box tests will be performed.

( ) Click on Friend Requests, then click x.

( ) Click on Friend Requests, then click Accept Request, then click outside.

( ) Click on Add Friend, then click outside the field, then click on Add Friend again.

( ) Click on Add Friend, leave Add User empty and click on Add.

( ) Click on Add Friend, Fill out all entries correctly and click on Add.

( ) Click on Add Friend, Fill out the user’s username and click on Add.

( ) Click on Compose Message, then click x.

( ) Click on Compose Message, leave entries empty then click Send Message.

( ) Click on Compose Message, fill entries with faults then click Send Message.

( ) Click on Compose Message, fill entries correctly empty then click Send Message.

( ) Click on friend.name ( friend.username ) in Table.

( ) Click on name in Table, date in table and repeat.

( ) Click on Show Entries in table and select 10, 25, 50, then 100.

( ) Click on Previous, Next, and then 1.

( ) Enter String in Search and return.

( ) Click on My Profile.

( ) Click on Messages.

( ) Click on Categories.

( ) Click on Logout.

### 2.11 Categories Page: used to view a list of categories stored in the CodeFeed database

#### 2.13.1 Equivalence classes for the Categories Page module

##### 2.13.1.1 “page load”

###### 1 A navbar with links - my profile, messages, categories, and logout – and the CodeFeed logo will be present on page load

###### 2 The page will be titled: Categories

###### 3 The page header will read: Select a Category

###### 4 A table with a list of categories or the message: “no table data present”, sorting features, and a search box will be centered on the page

###### 5 A button labeled “Add category” will be present on the upper right of the page

###### 6 Anything else is a failure

##### 2.13.1.2 “click add category”

###### 1 When the “Add category” button is clicked, there should be a modal popup with the follow fields: desired name, description, and a “Post Category” button

###### 2 Anything else is a failure

##### 2.13.1.3 “search table”

###### 1 When the search box is typed into with any character, the table will be filtered to display entries that contain that string

###### 2 If no category contains the entered text, the text “No matching records” will be shown

###### 3 When the search box is cleared, the table will be returned to its original, unfiltered form

###### 4 Anything else is a failure

##### 2.13.1.4 “desired name”

###### 1 combination of letters, numbers, and symbols between 1 and 64 characters acceptable

###### 2 anything else error

##### 2.13.5 “description”

###### 1 any combination of any character between 1 and 64 characters acceptable

###### 2 else error

##### 2.13.6 “category links”

###### 1 clicking on a category link will redirect to /category/?category\_id=<id> acceptable

###### 2 else error

#### 2.13.2 Checklist the following black box tests will be performed

##### Navbar loads properly

##### Title loads properly

##### Table loads when blank

##### Table loads when full

##### Search shows correct results when not found

##### Search shows correct results when found

##### Desired name exists in database

##### Desired name blank

##### Desired name ok

##### Description ok

##### Description blank

##### Category link clicked

### 2.12 Category Page: used to view a single category within the list of categories

#### 2.13.1 Equivalence classes for the Category Page module

##### 2.13.1.1 “page load”

###### 1 A navbar with links - my profile, messages, categories, and logout – and the CodeFeed logo will be present on page load

###### 2 The page will be titled: Threads in thread\_name

###### 3 The page header will read: Select a Thread to View

###### 4 A table with a list of threads or the message: “no table data present”, sorting features, and a search box will be centered on the page

###### 5 A button labeled “New thread” will be present on the upper right of the page

###### 6 Anything else is a failure

##### 2.13.1.2 “click new thread”

###### 1 When the “Add new” button is clicked, there should be a modal popup with the follow fields: thread titel, body, and a “Submit thread” button

###### 2 Anything else is a failure

##### 2.13.1.3 “search table”

###### 1 When the search box is typed into with any character, the table will be filtered to display entries that contain that string

###### 2 If no thread contains the entered text, the text “No matching records” will be shown

###### 3 When the search box is cleared, the table will be returned to its original, unfiltered form

###### 4 Anything else is a failure

##### 2.13.1.4 “thread name”

###### 1 combination of letters, numbers, and symbols between 1 and 64 characters acceptable

###### 2 anything else error

##### 2.13.5 “description”

###### 1 any combination of any character between 1 and 64 characters acceptable

###### 2 else error

##### 2.13.6 “thread links”

###### 1 clicking on a thread link will redirect to /category/thread/?thread\_id=<id> acceptable

###### 2 else error

#### 2.13.2 Checklist the following black box tests will be performed

##### Navbar loads properly

##### Title loads properly

##### Table loads when blank

##### Table loads when full

##### Search shows correct results when not found

##### Search shows correct results when found

##### Thread title blank

##### Thread title ok

##### Body ok

##### Body blank

##### Thread link clicked

### 2.13 Category/Thread Page: used to view a single thread associated with a category

#### 2.13.1 Equivalence classes for the Thread Page module

##### 2.13.1.1 “page load”

###### 1 A navbar with links - my profile, messages, categories, and logout – and the CodeFeed logo will be present on page load

###### 2 The page will be titled: Threads in thread\_name

###### 3 The page header will read: Select a Thread to View

###### 4 A table with a list of threads or the message: “no table data present”, sorting features, and a search box will be centered on the page

###### 5 A button labeled “New thread” will be present on the upper right of the page

###### 6 Anything else is a failure

##### 2.13.1.2 “click new thread”

###### 1 When the “Add new” button is clicked, there should be a modal popup with the follow fields: thread titel, body, and a “Submit thread” button

###### 2 Anything else is a failure

##### 2.13.1.3 “search table”

###### 1 When the search box is typed into with any character, the table will be filtered to display entries that contain that string

###### 2 If no thread contains the entered text, the text “No matching records” will be shown

###### 3 When the search box is cleared, the table will be returned to its original, unfiltered form

###### 4 Anything else is a failure

##### 2.13.1.4 “thread name”

###### 1 combination of letters, numbers, and symbols between 1 and 64 characters acceptable

###### 2 anything else error

##### 2.13.5 “description”

###### 1 any combination of any character between 1 and 64 characters acceptable

###### 2 else error

##### 2.13.6 “thread links”

###### 1 clicking on a thread link will redirect to /category/thread/?thread\_id=<id> acceptable

###### 2 else error

#### 2.13.2 Checklist the following black box tests will be performed

##### Navbar loads properly

##### Title loads properly

##### Table loads when blank

##### Table loads when full

##### Search shows correct results when not found

##### Search shows correct results when found

##### Thread title blank

##### Thread title ok

##### Body ok

##### Body blank

##### Thread link clicked

### 2.14 The unit tests will be performed by Alec, Jeremy, and Bert

# 3 Integration Testing

## 3.1 The purpose of the integration tests is to test the functionality of CodeFeed. This includes all the features required in the documentation and possible future improvements. These are test cases to test the depth and degree of integration, aka: Does it work for all edge cases and corner cases? We will group specific components of the system by their routes.

## 3.2 Checklist the following black box tests will be performed

#### Add new account with invalid input

#### Add new account with valid input

#### Login with invalid input

#### Login with valid input

#### While logged in:

#### Update profile with invalid input

#### Update profile with valid input

#### Add friend with invalid input

#### Add friend with valid input

#### Send message to non-friend

#### Send message to friend

#### View messages

#### Respond to message with invalid input

#### Respond to message with valid input

#### View categories

#### View threads in a category

#### View comments in a thread

#### Vote on thread

#### Vote on comment

#### Add category with invalid input

#### Add category with valid input

#### Add thread with invalid input

#### Add thread with valid input

#### Add comment with invalid input

#### Add comment with valid input

#### While not logged in:

#### Update profile with invalid input

#### Update profile with valid input

#### Add friend with invalid input

#### Add friend with valid input

#### Send message to non-friend

#### Send message to friend

#### View messages

#### Respond to message with invalid input

#### Respond to message with valid input

#### View categories

#### View threads in a category

#### View comments in a thread

#### Vote on thread

#### Vote on comment

#### Add category with invalid input

#### Add category with valid input

#### Add thread with invalid input

#### Add thread with valid input

#### Add comment with invalid input

#### Add comment with valid input

## 3.3 The tests will be performed by Alec

# 4 System Testing

## 4.1 These tests will make sure the basics of the program (startup, initialization, configuration, routes creation, etc.) will work.

## 4.2 Checklist the following white and grey box tests will be performed

### Initialize database in debug mode

### Initialize database in testing mode

### Initialize database in release mode

### Run the application in debug mode

### Run the application in testing mode

### Run the application in release mode

### View / route

### View /login route

### View /profile route

### View /category route

## 4.3 The system tests will be performed by Alec

# 5 Acceptance Testing

## 5.1 The purpose of the acceptance tests is to test just the end-user-focused of the feature set. This includes all the features required in the documentation and possible future improvements. These tests should be more user experience based, rather than technical.

## 5.2 Checklist: the following black box tests will be performed

### Register for an account

### Login to new account

### Logout of account

### View profile

### Edit profile

### View friends

### Add new friend

### View messages

### Create new message

### Respond to message

### View categories

### Add new category

### View threads in a category

### Add thread to category

### Vote on category

### View comments on a thread

### Add new comment on a thread

### Comment on thread

## 5.3 The acceptance tests will be performed by Jeremy