**Sprint 2 Manual Testing Report**

Team 18: Chunao Liu, Anurag Shah, Jenna Zhang, Yierpan Abuduwaili, Michelle He, Jingyuan Yang

**User Story #2: As a user, I would like to have a way to reset my password.**

* Test Case 1:

Input: Click ‘Forgot password” on LoginPage

Output:

o Backend: N/A

o Frontend: Prompt users to enter their registered email

Acceptable? - Yes

* Test Case 2:

Input: email: janneynana

Output:

o Backend: N/A

o Frontend: Error message displayed: “Please enter a valid email address”.

Acceptable? - Yes

* Test Case 3:

Input: email: adm@gmail.

Output:

o Backend: Responds with code -2, msg: "Wrong username!"

o Frontend: Error message displayed: “Email does not exist ”

Acceptable? - Yes

* Test Case 4:

Input: email: janneynanana@gmail.

Output

o Backend: Responds with code 1, msg: "Email successfully sent"

o Frontend: Success message: “Password reset link successfully sent ”

o Mailbox: An email resetting email received

Acceptable? - Yes

* Test Case 5:

Input: new passwords

Output

o Backend: Responds with code 0, msg: "password successfully sent"

o Frontend: Successfully logged in with new password

Acceptable? - Yes

**User Story #3: As a developper, I would like to enhance the security of our application by identifying a user, machine, or device before granting access to a resource, network, application.**

* Test Case 1:

Input: Send getAvatar() request without token included in the header

Output:

o Backend: "GET /User/avatar/ HTTP/1.1" 401 58”

o Frontend: “Authentication credentials were not provided.”

Acceptable? - Yes

* Test Case 2:

Input: Send setAvatar() request without token included in the header

Output:

o Backend: "POST /User/avatar/ HTTP/1.1" 401 58”

o Frontend: “Authentication credentials were not provided.”

Acceptable? - Yes

* Test Case 3:

Input: Send setAvatar() request WITH token included in the header

Output:

o Backend: "POST /User/avatar/ HTTP/1.1" 200 63

o Frontend: “Avatar successfully set”

Acceptable? - Yes

**User Story 11: As a user, I would like to be able to see and edit my account information.**

* Test Case 1:

Input: checking account info page

Output:

o Backend: N/A

o Frontend: Redirected to account page when account is clicked

Acceptable? - Yes

* Test Case 2:

Input: editing account info at account info page

Output:

o Backend: N/A

o Frontend: Button clicked and editable but no backend update yet

Acceptable? - yes

* Test Case 3:

Input: redirecting between pages

Output:

o Backend: N/A

o Frontend: Button clicked redirects to correct pages

Acceptable? - Yes

**User Story #10: As a user, I would like to be able to take pictures of my code or import it from the gallery and receive result and save the result to backend**

* Test Case 1:

Input: Taking picture

Output:

o Backend: Responds with code http 200 OK

o Frontend: Picture static and good

Acceptable? - Yes

* Test Case 2:

Input: Importing image from gallery and sending

Output:

o Backend: Responds with code http 200 OK

o Frontend: Picture static and ready for processing

Acceptable? - Yes

* Test Case 3:

Input: received result and saved in backend

Output:

o Backend: Success

o Frontend: Saved

Acceptable? - Yes

**User story #4: As a user, I would like to see whether creating an account is successful or not.**

* Test case 1:

Inputs:

* + Username: michelle
  + Email: michelle
  + Password: abc
  + Confirm Password: abc

Outputs:

* + Backend: N/A
  + Frontend: error message: “invalid email!”

Acceptable? - Yes

* Test case 2:

Inputs:

* + Username: michelle
  + Email: michelle@gmail
  + Password: abc
  + Confirm Password: abc

Outputs:

* + Backend: N/A
  + Frontend: error message: “invalid email!”

Acceptable? - Yes

* Test case 3:

Inputs:

* + Username: michelle
  + Email: michelle@gmail.c
  + Password: abc
  + Confirm Password: abc

Outputs:

* + Backend: N/A
  + Frontend: error message: “Invalid Email!”

Acceptable? - Yes

* Test case 4:

Inputs:

* + Username: michelle
  + Email: michelle@gmail.com
  + Password: abc
  + Confirm Password: ab

Outputs:

* + Backend: N/A
  + Frontend: error message: “Password Mismatch!”

Acceptable? - Yes

* Test case 5:

Inputs:

* + Username: michelle
  + Email: michelle@gmail.com
  + Password:
  + Confirm Password: ab

Outputs:

* + Backend: N/A
  + Frontend: error message: “Password Mismatch!”

Acceptable? - Yes

* Test case 6:

Inputs:

* + Username: michelle
  + Email: michelle@gmail.com
  + Password: abcd
  + Confirm Password: abcd

Outputs:

* + Backend: successfully create the new user and add the user into database; send “success” response to frontend
  + Frontend: Alert showing “success” message

Acceptable? - Yes

* Test case 7 (assuming the new user from test case 6 is already in database):

Inputs:

* + Username: michelle
  + Email: michelle@gmail.com
  + Password: abcd
  + Confirm Password: abcd

Outputs:

* + Backend: send “both username and email in use” response to frontend
  + Frontend: Alert showing “both username and email in use” message

Acceptable? - Yes

* Test case 8 (assuming the new user from test case 6 is already in database):

Inputs:

* + Username: michelle
  + Email: michelle123@gmail.com
  + Password: abcd
  + Confirm Password: abcd

Outputs:

* + Backend: send “username in use” response to frontend
  + Frontend: Alert showing “username in use” message

Acceptable? - Yes

* Test case 9 (assuming the new user from test case 6 is already in database):

Inputs:

* + Username: michelle123
  + Email: michelle@gmail.com
  + Password: abcd
  + Confirm Password: abcd

Outputs:

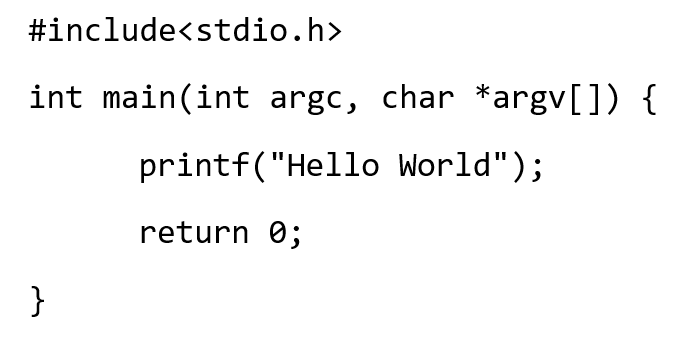
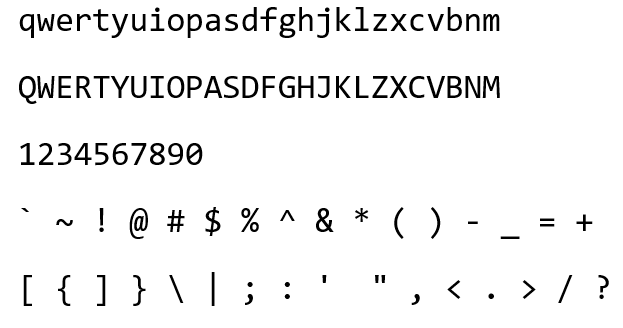
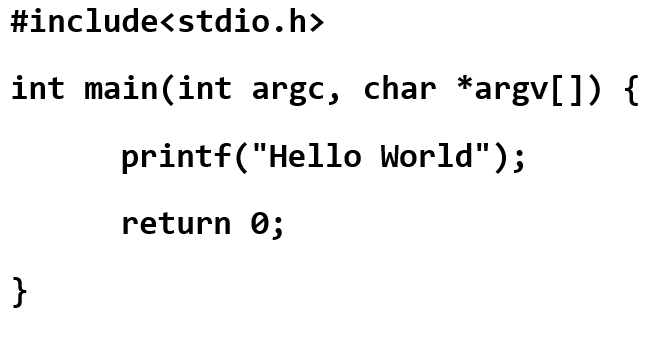
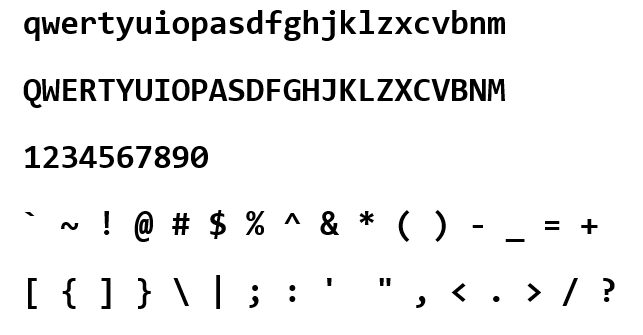
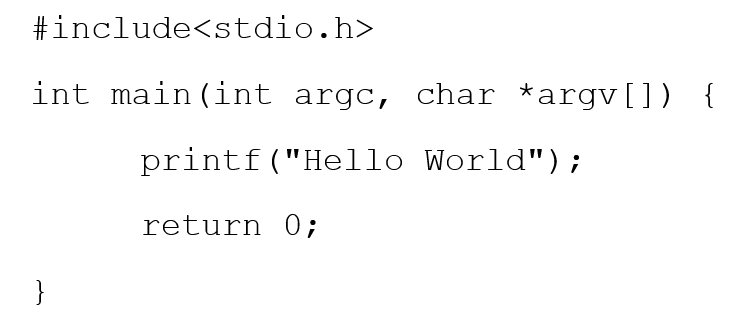
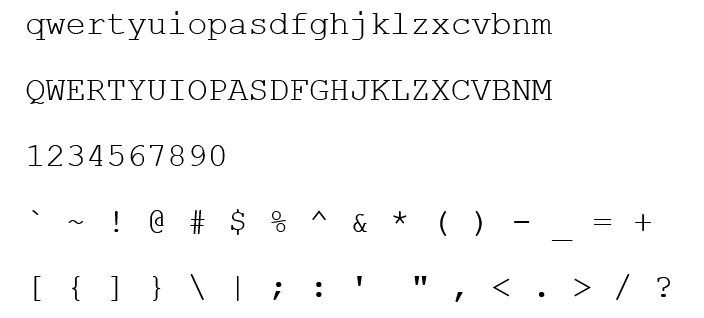
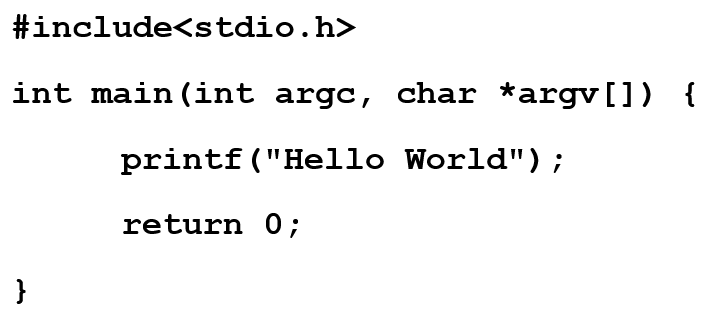
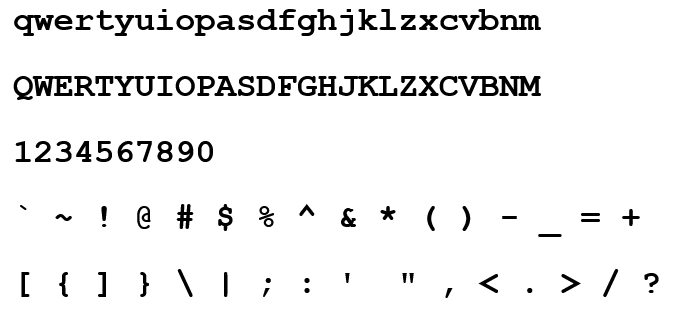
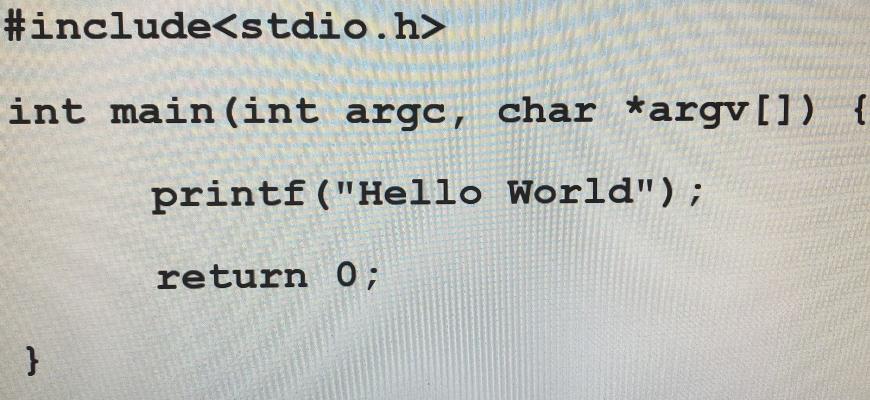
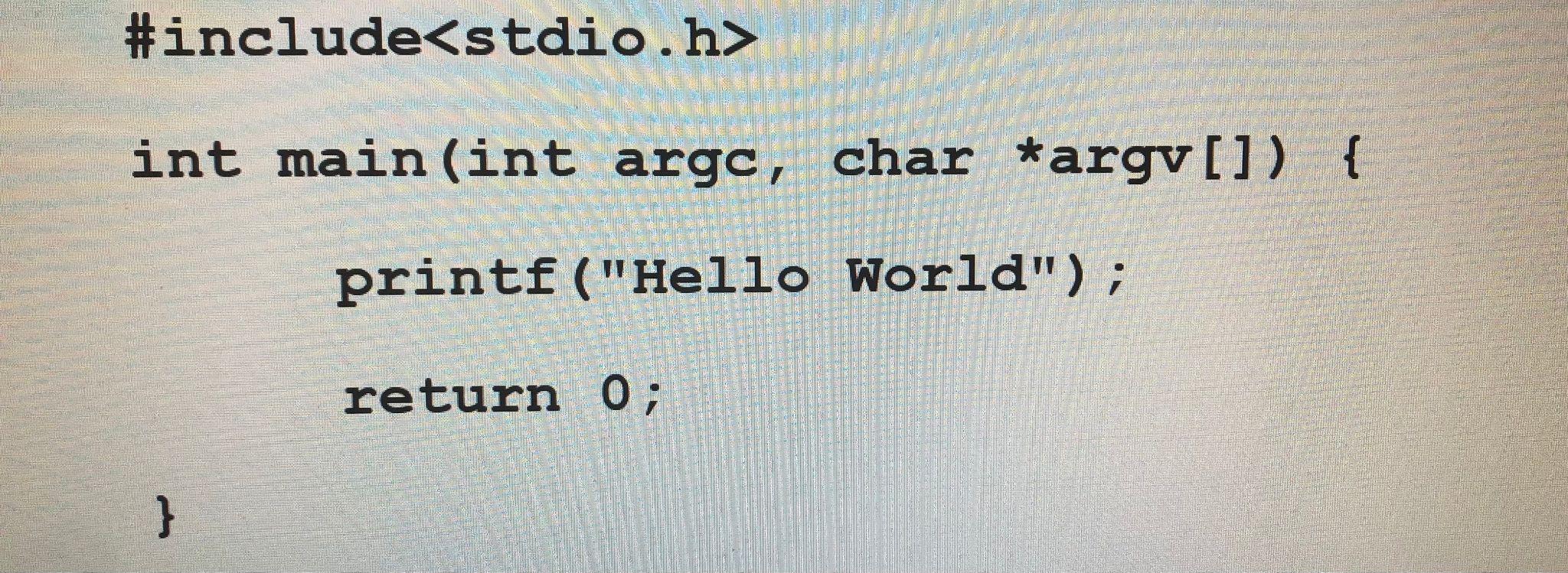
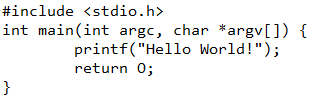
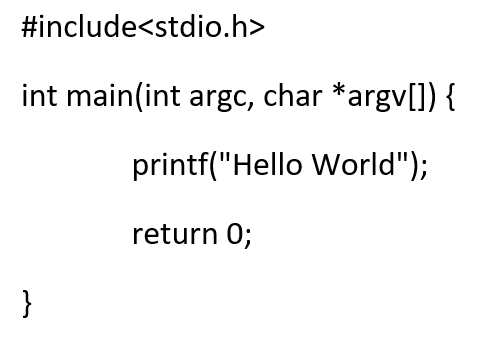
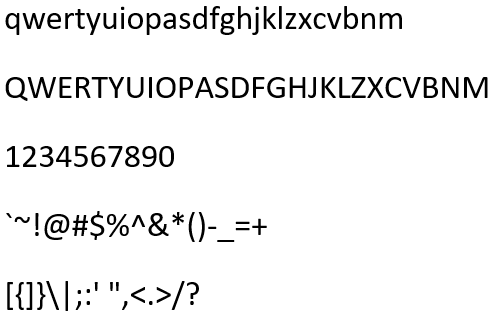
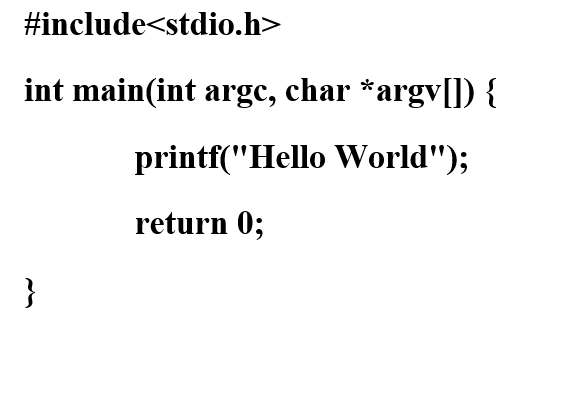
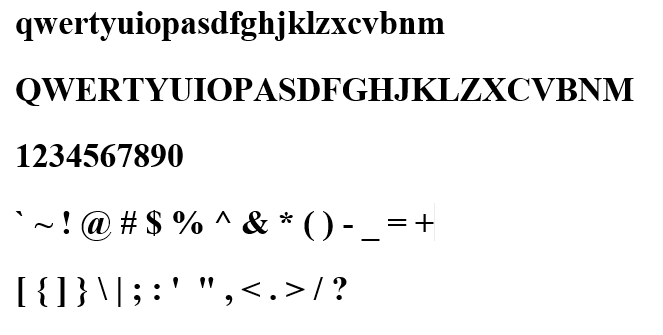
* + Backend: send “email in use” response to frontend
  + Frontend: Alert showing “email in use” message

Acceptable? - Yes

**User Story #8, #12, #13: As a developer, I would like to automatically compile the code given by the user and send the correct runtime result back to the user**

* Test Case 1:   
  Input: ‘#include<stdio.h>\n\nint main(int argc, char \*argv[]) {\nprintf ("Hello World") ;\nreturn 0;\n\n}’  
  Output: Hello World  
  Acceptable: Yes
* Test Case 2:  
  Input: ‘aaaaa#include<stdio.h>\n\nint main(int argc, char \*argv[]) {\nprintf ("Hello World") ;\nreturn 0;\n\n}’  
  Output: Error in various formats for the different stories  
  Acceptable: Yes

**User Story #9: As a user, I would like the application to process my image of typeform code, and convert it to the relevant textual representation.**

* All test cases for this portion are images. The goal is to ensure that the image produces the desired output. The test cases are located at WhiteBoard\Backend\OCR\images\tesseract\_tests. Certain characters and font styles do fail the test case, but are maintained in the folder for future use. These test cases are manual, the main function inside ocr.py (in WhiteBoard\Backend\WhiteBoardBackEnd\API) has to be manually updated to run them.  
    
  As noted above, the expected output for each test case is the text in that image. The actual output varies as tesseract is not deterministic. An output is “acceptable” when very few characters are wrong.
* Test case 1:  
    
  Acceptable: Yes
* Test case 2:  
    
  Acceptable: Yes
* Test case 3:  
    
  Acceptable: Yes
* Test case 4:  
    
  Acceptable: Yes
* Test case 5:  
    
  Acceptable: Yes
* Test case 6:  
    
  Acceptable: Yes
* Test case 7:  
    
  Acceptable: Yes
* Test case 8:  
    
  Acceptable: No
* Test case 9:  
    
  Acceptable: Yes
* Test case 10:  
    
  Acceptable: Yes
* Test case 11:  
    
  Acceptable: Yes
* Test case 12:  
    
  Acceptable: Yes
* Test case 13:  
    
  Acceptable: Yes
* Test case 14:  
    
  Acceptable: Yes
* Test case 15:  
    
  Acceptable: Yes
* Test case 16:

Transfer image via Postman

var formdata **=** **new** FormData();

formdata.**append**("", fileInput.files[0], "[PROXY]");

formdata.**append**("name", "Hello");

var requestOptions **=** {

method: 'POST',

body: formdata,

redirect: 'follow'

};

fetch("http://ec2-3-138-112-15.us-east-2.compute.amazonaws.com:8000/Images/1", requestOptions)

.**then**(response **=>** response.text())

.**then**(result **=>** console.**log**(result))

.**catch**(error **=>** console.**log**('error', error));

Server can receive the picture and store it into the database correctly

Acceptance: Yes