**Lab 01- Calculator and Twitter**

**Calculator:**

**DESCRIPTION:**

**The goal is designing a basic calculator to demonstrate stateless web service and giving it the basic functionality also taking care of the errors**

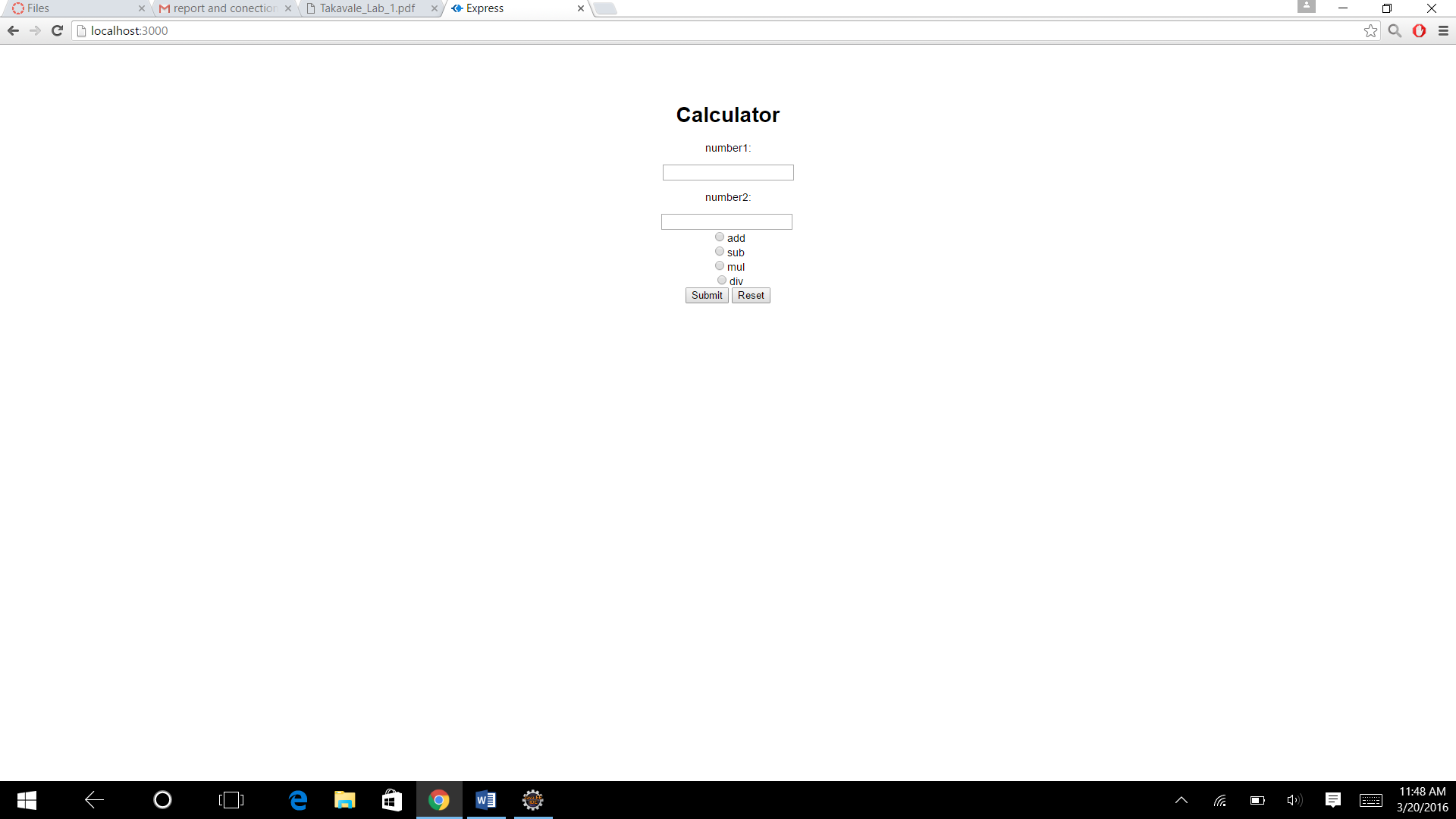
**System Design:**

**I have choosen node js and HTML5 for this application.**

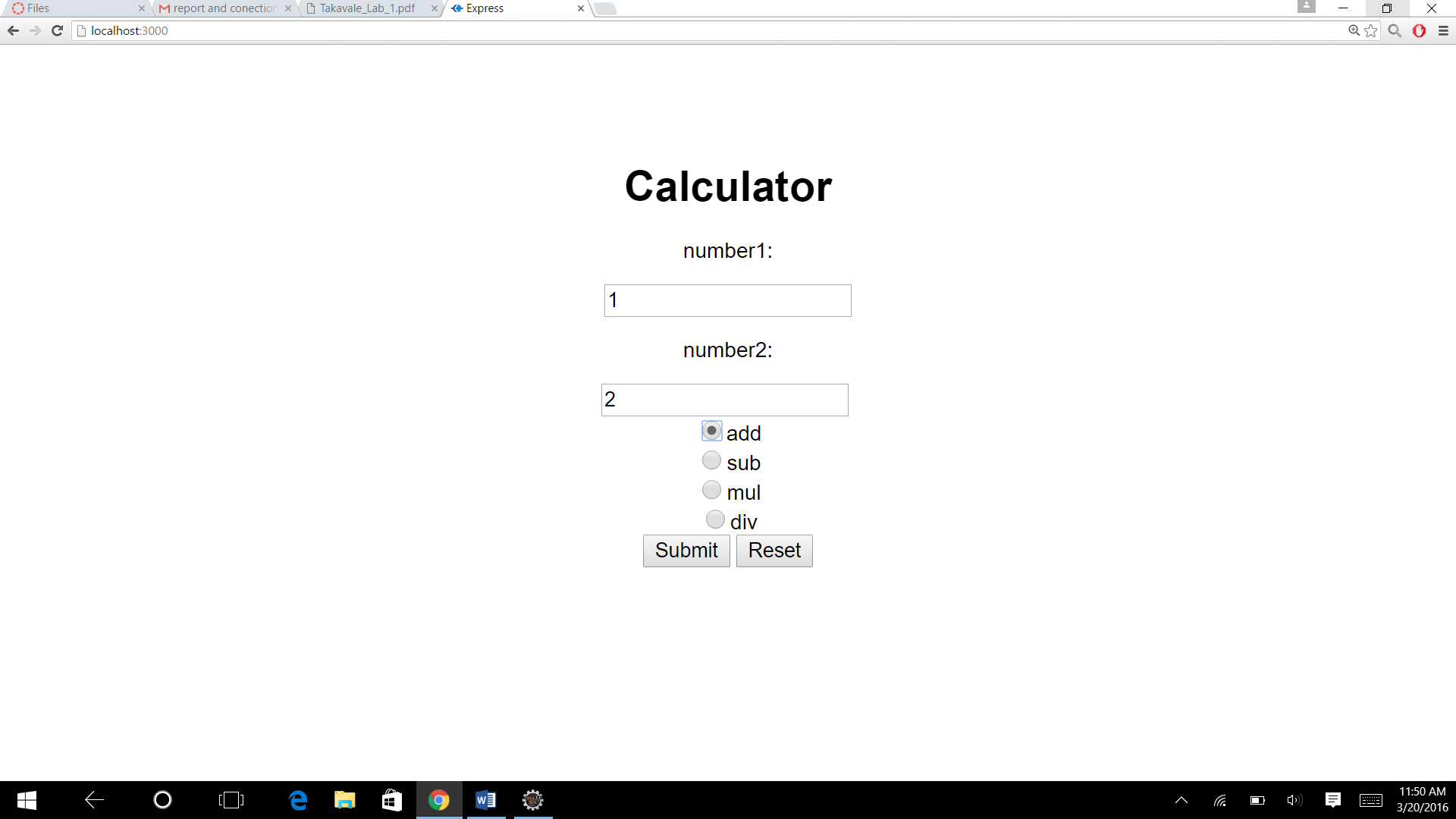
**HTML5 is implemented in the frontend of the application from we get the input and depending on the operation selected we give them the output in different page and after it we bring them in the same page to perform another calculation**

**Calculator**

**Application Home screen**



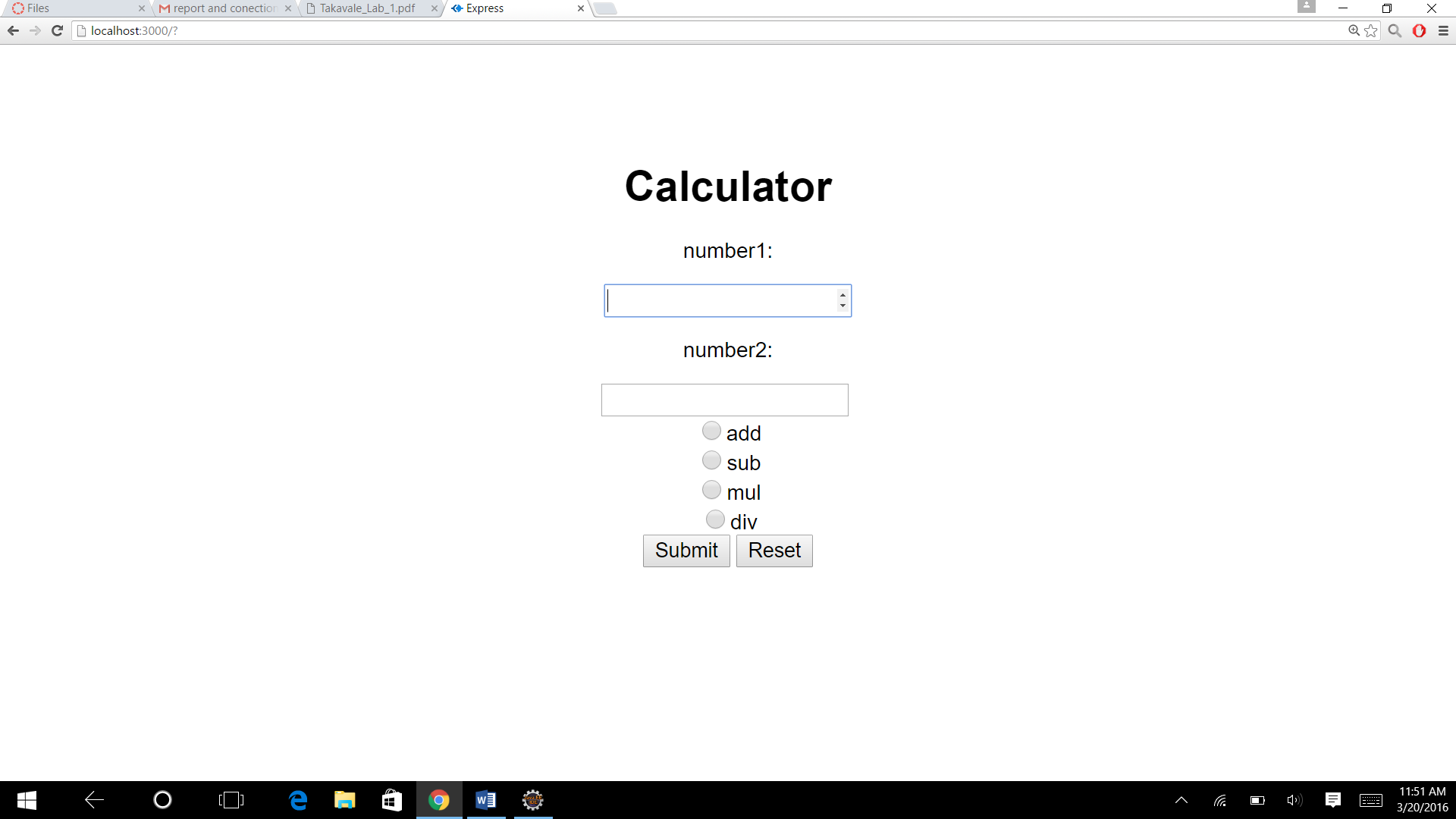
**Addition**



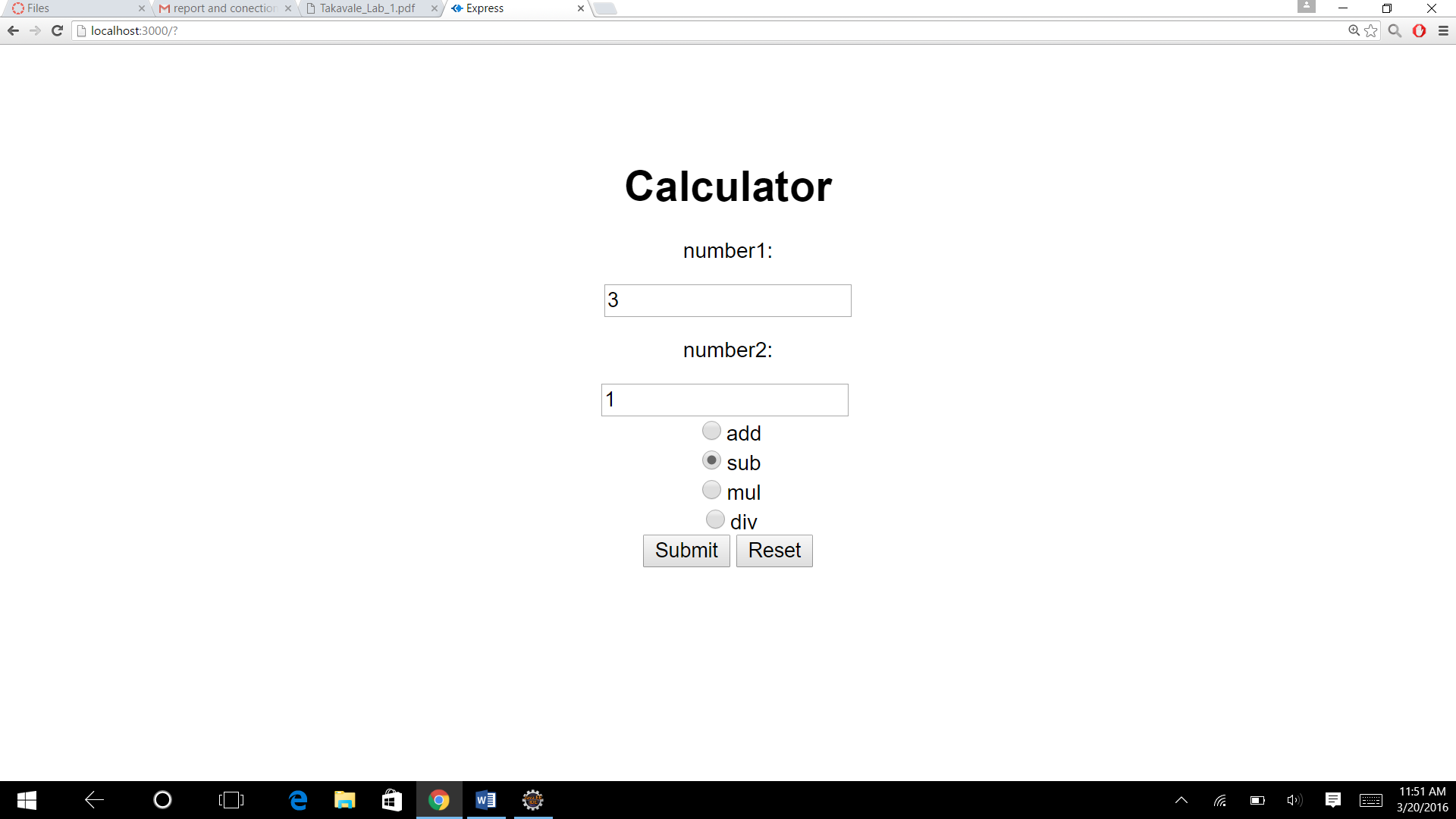
**Result:**



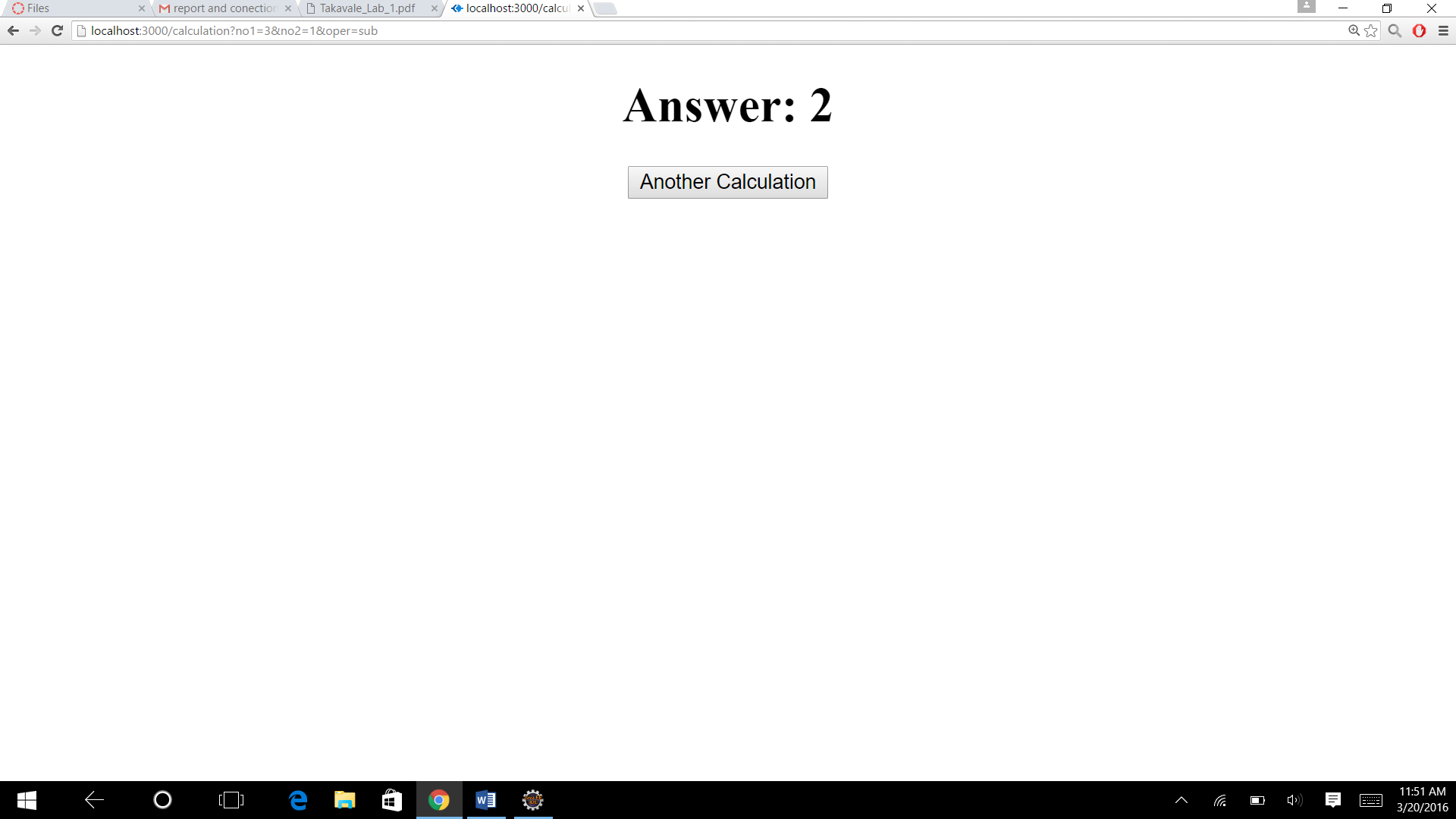
**Pressing another calculation:**



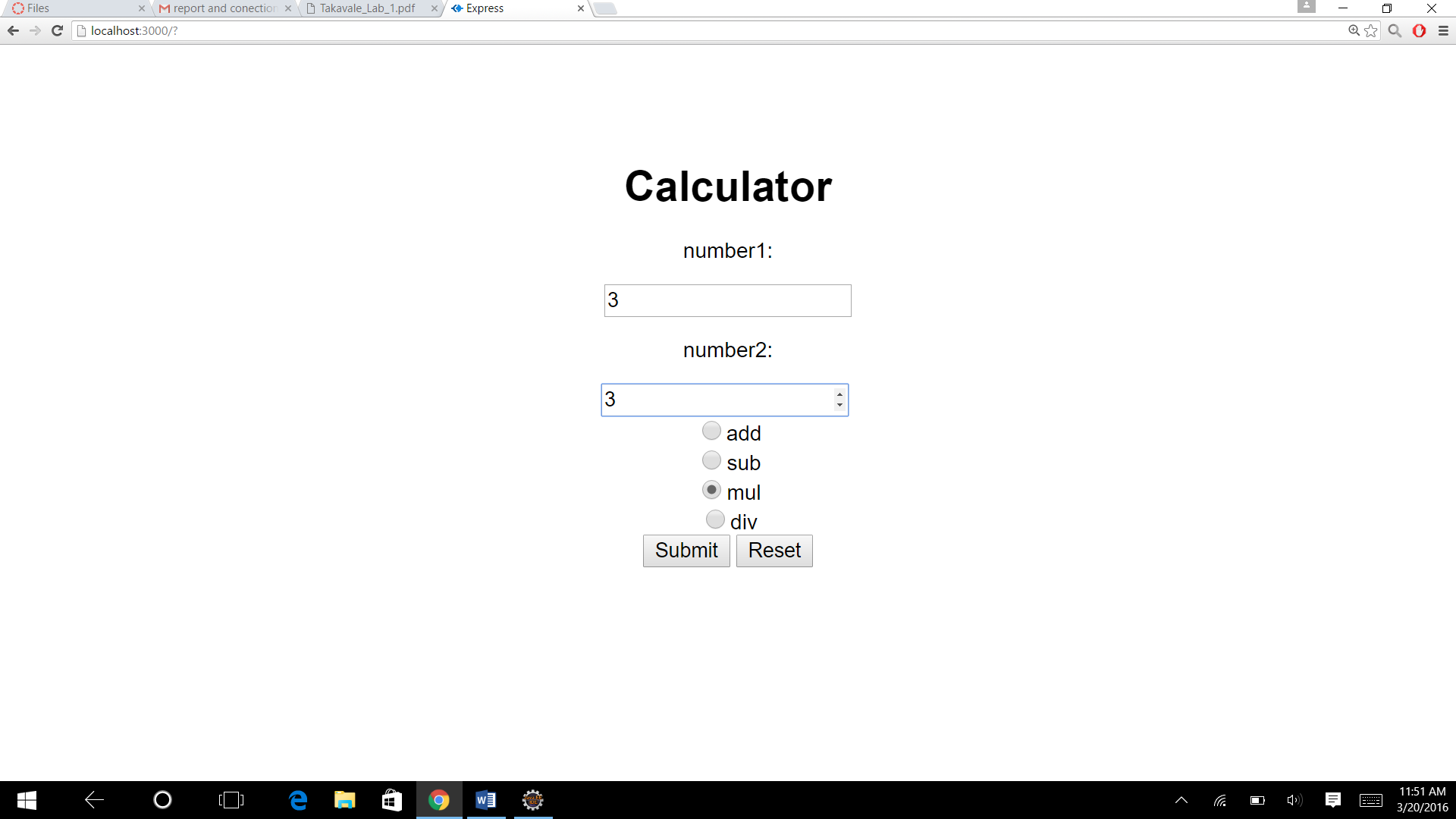
**Subtraction:**



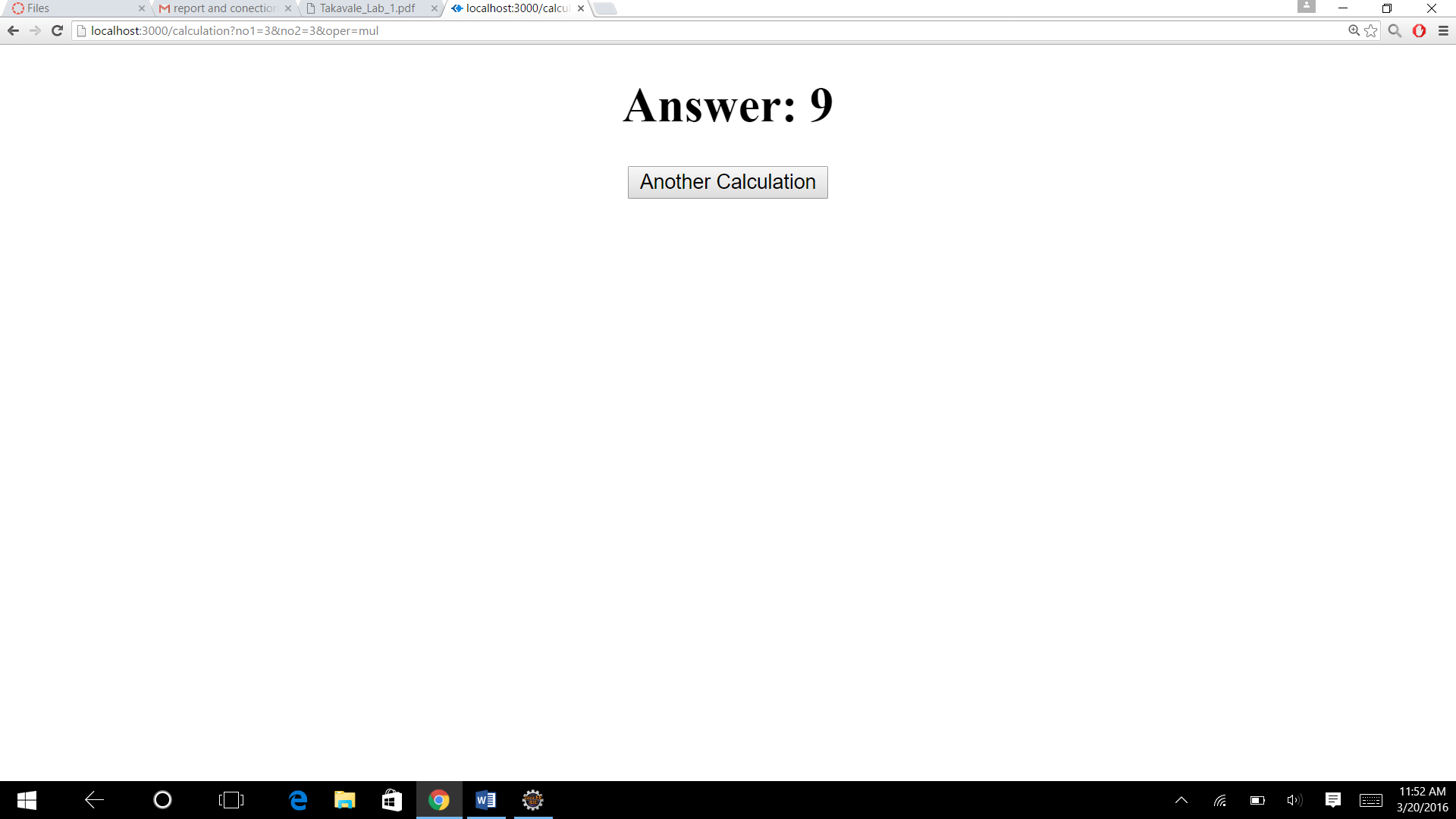
**Result:**



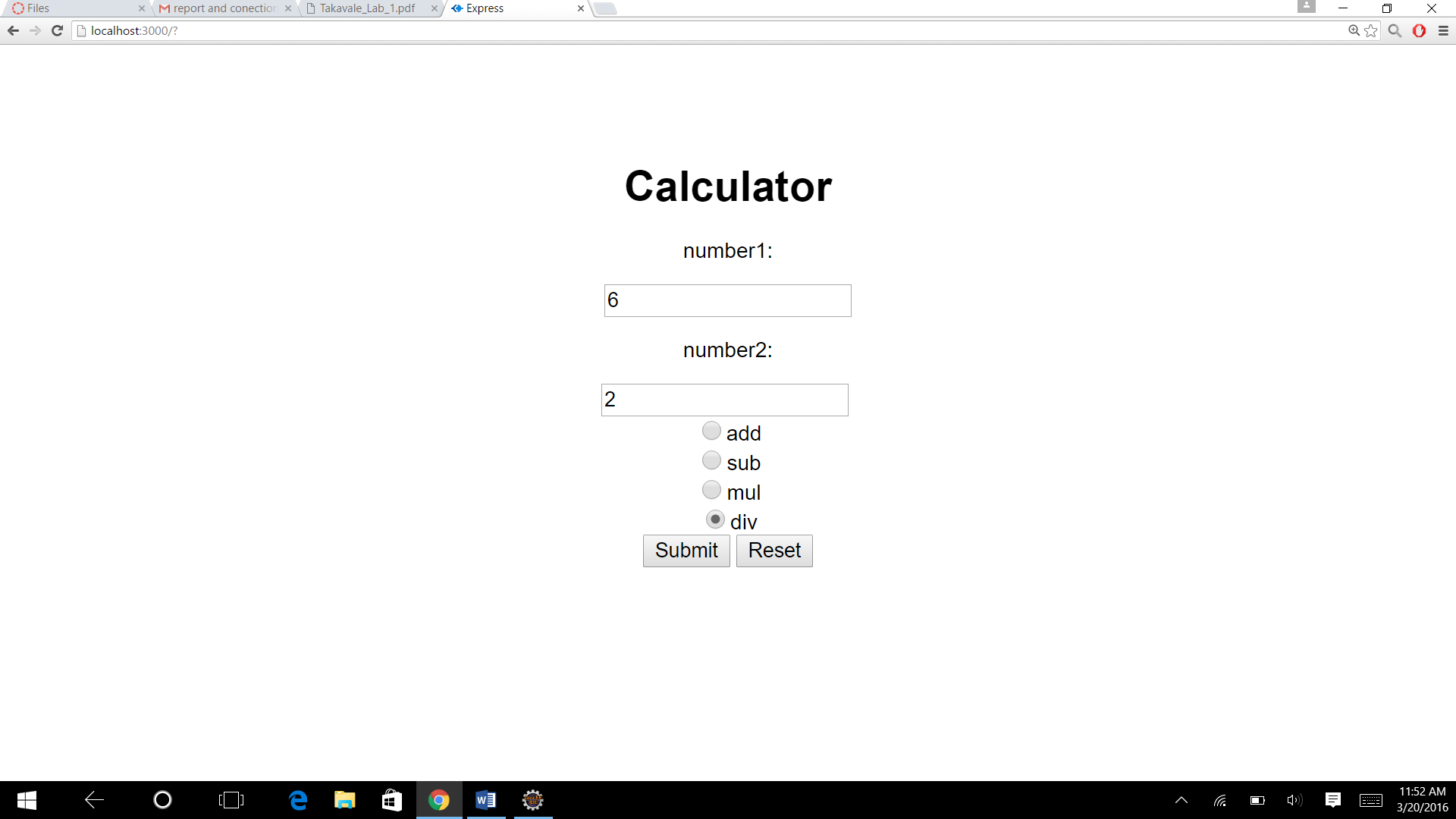
**Multiplication:**



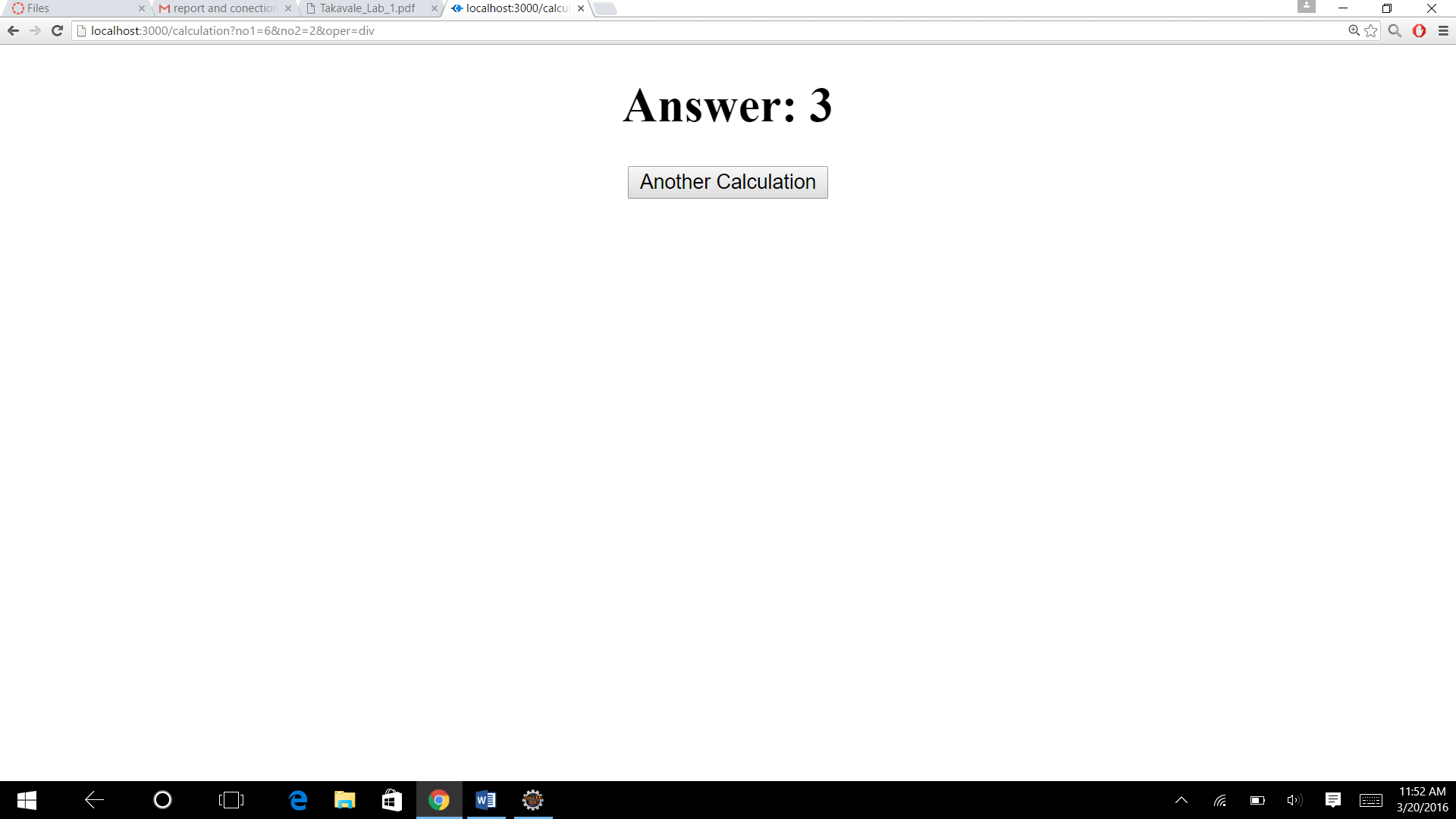
**Result:**



**Division:**

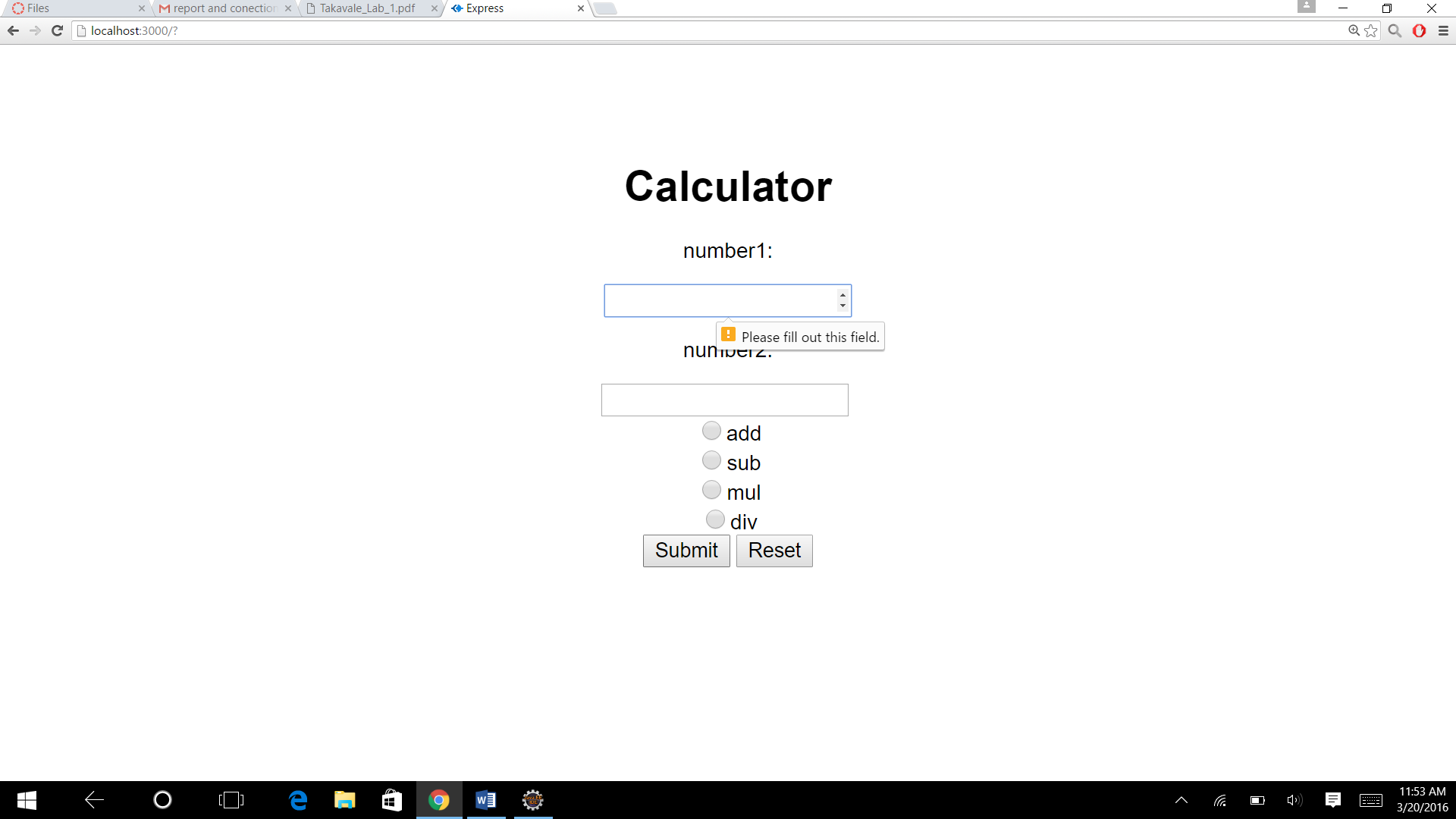


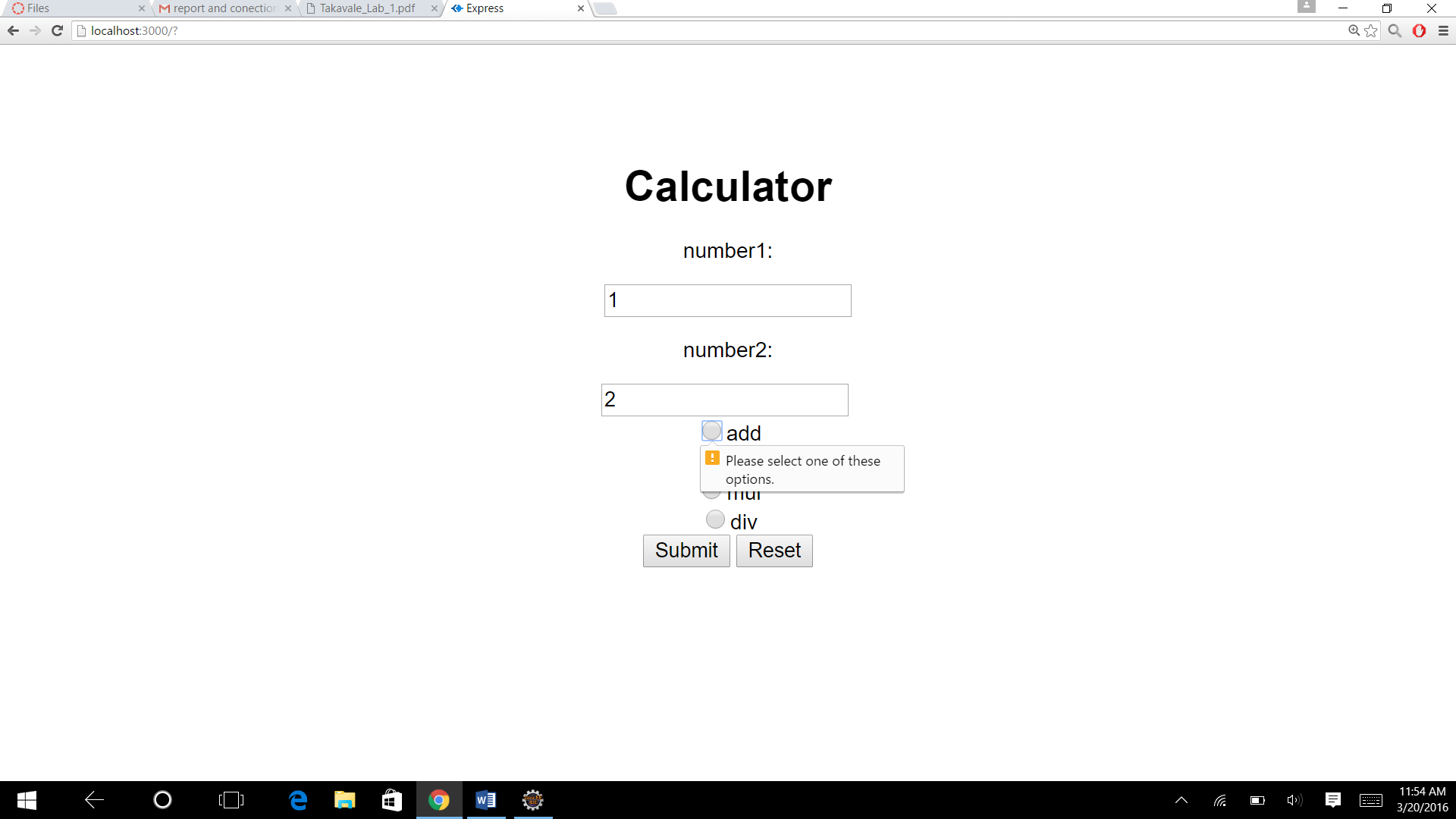
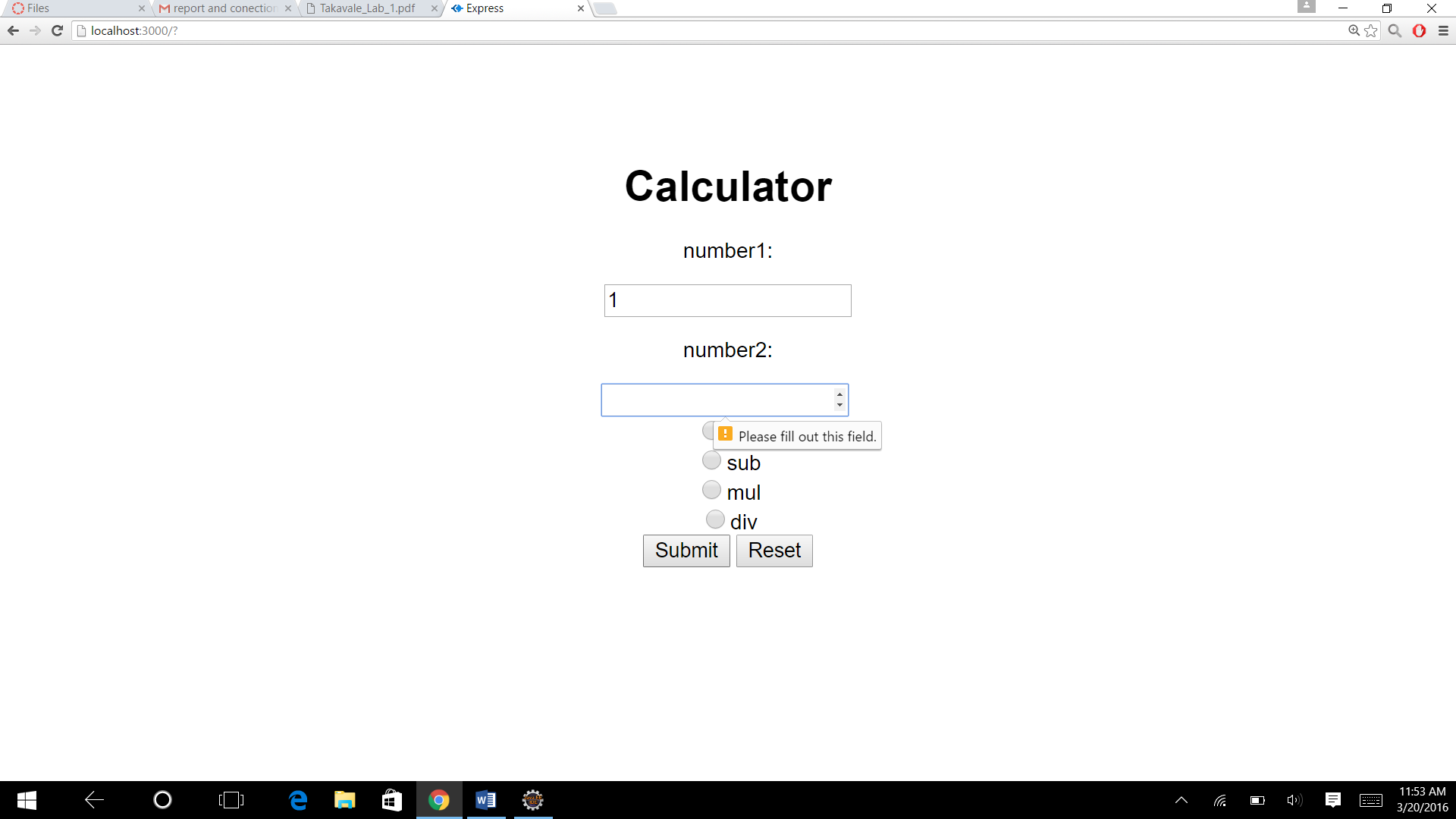
**Result:**



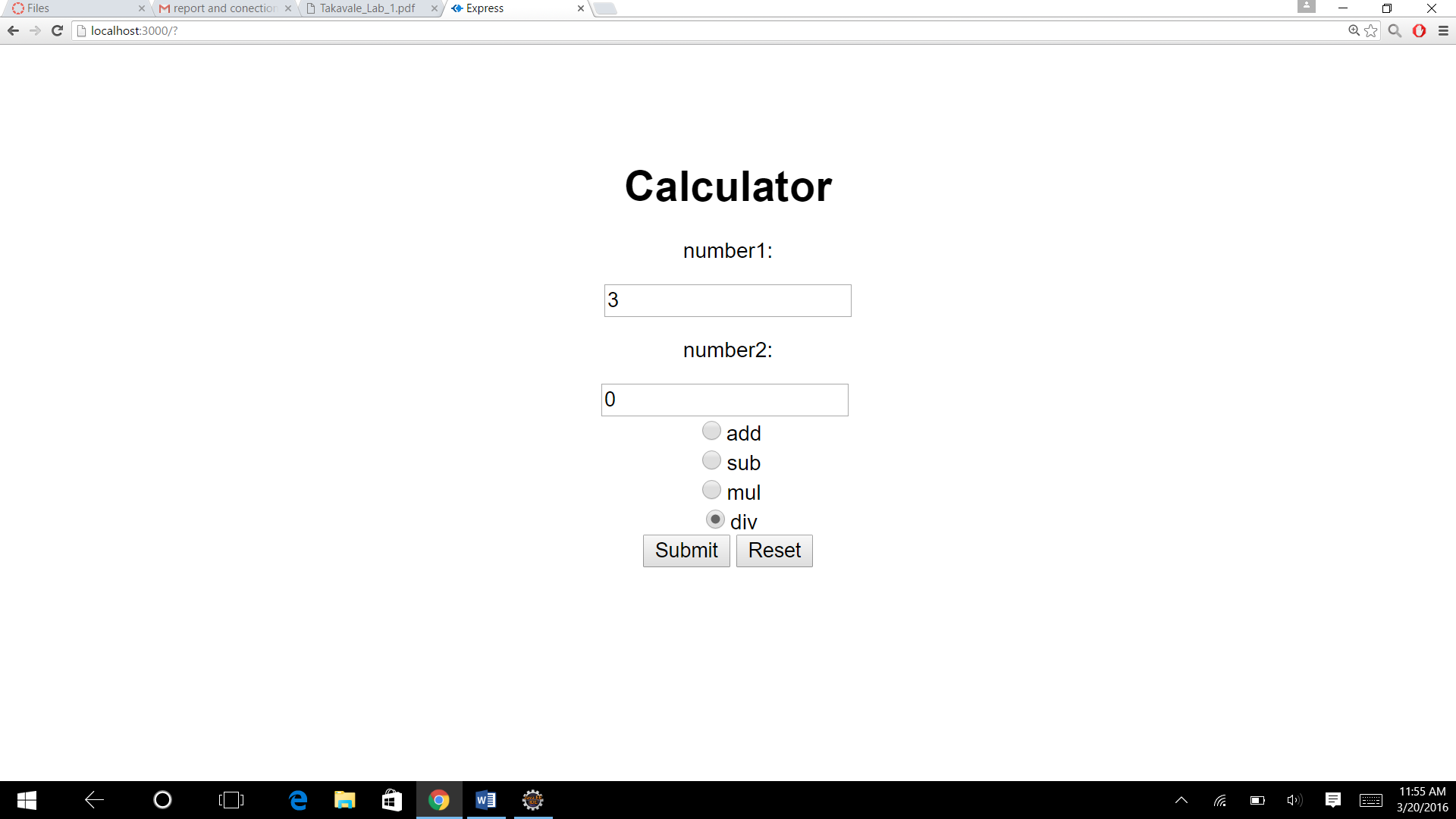
**Error handling**

**1taking no input and pressing submit:**





**2 dividing by zero:**



**Result:**



**Jmeter**

|  |
| --- |
|  |

Graph Discription:

the architecture of Node.Js which supports Non BLocking, Asynchronous and Single Event Loop framework and it also caches the requests from the same user . The above result has been

|  |  |
| --- | --- |
|  | Calculate |
| 1 user 1000 calls | 5 |
| 1 user 5000 calls | 4 |
| 100 user 1000 calls | 20 |

**Twitter**

**Goals:**

We have to implement a twitter application and we have to implement the below features

**1. Basic User functionalities:**

a. Sign up the new users (First name, Last name, Email, Password). Passwords

have to be encrypted

b. Sign in with existing users

c. Sign out

**2. Profile:**

a. About: Birthday, Twitter handle, contact information and location

b. Followers and Following list. You should be able to follow people.

c. Show your tweets and re-tweets

3. **Twitter feed functionality** showing tweets of people you are following and option

to re-tweet.

4. **Implementing Hashtag** (#) functionality (in Search and Tweets).

5. Implement **Connection pooling** for database access

**System Design:**

We have implemented angular js in the front end and bootstrap as well which will help us in the client side and we will be able to get attractive ui.

Node js in the middleware where we can call different type of queries and we are able to perform functions which will help us to basic programing in the middle

The backend is mysql which is connected with the middleware throw server.

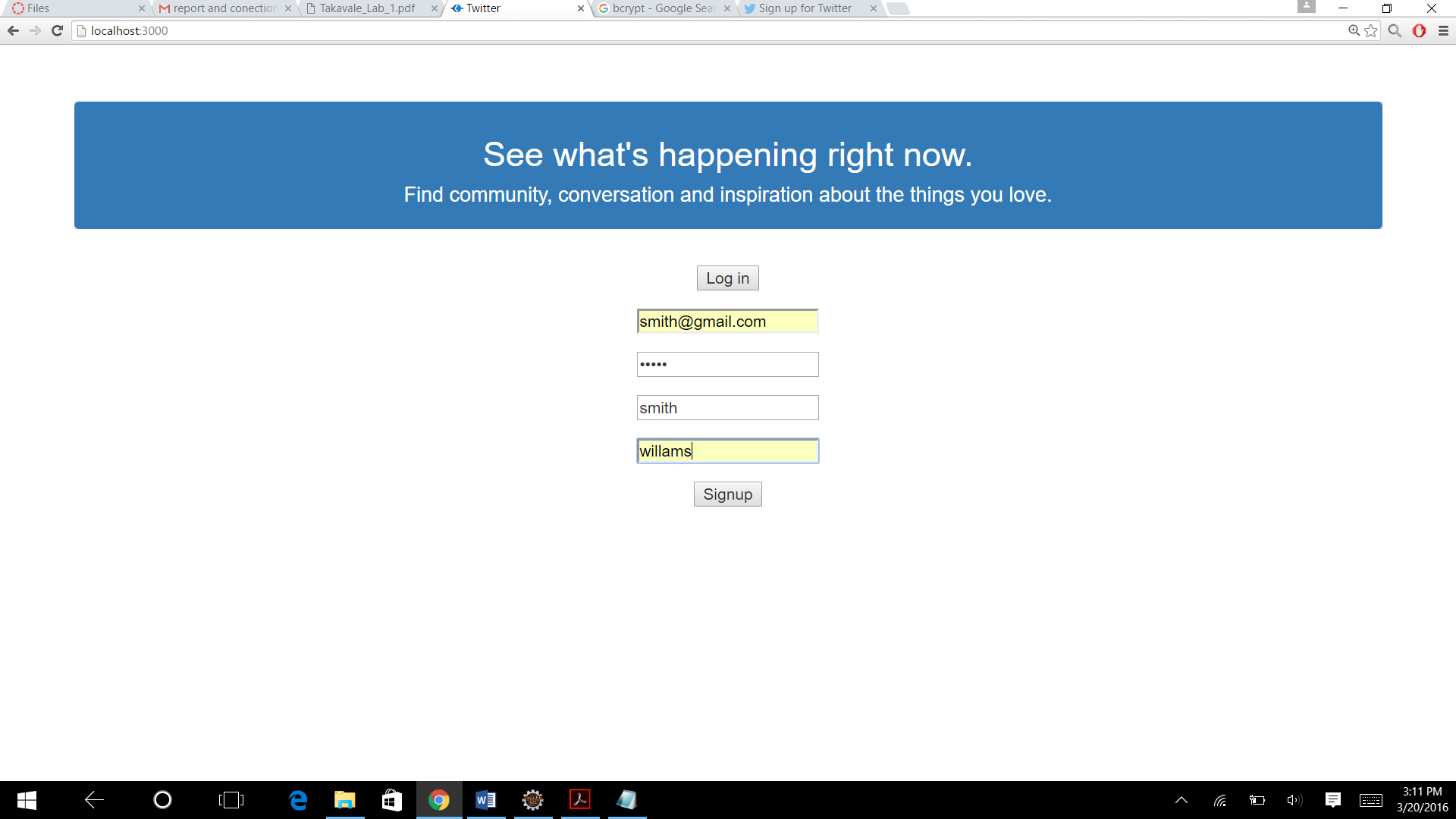
This is how the client and the server is connected.

**1 Basic User functionalities**

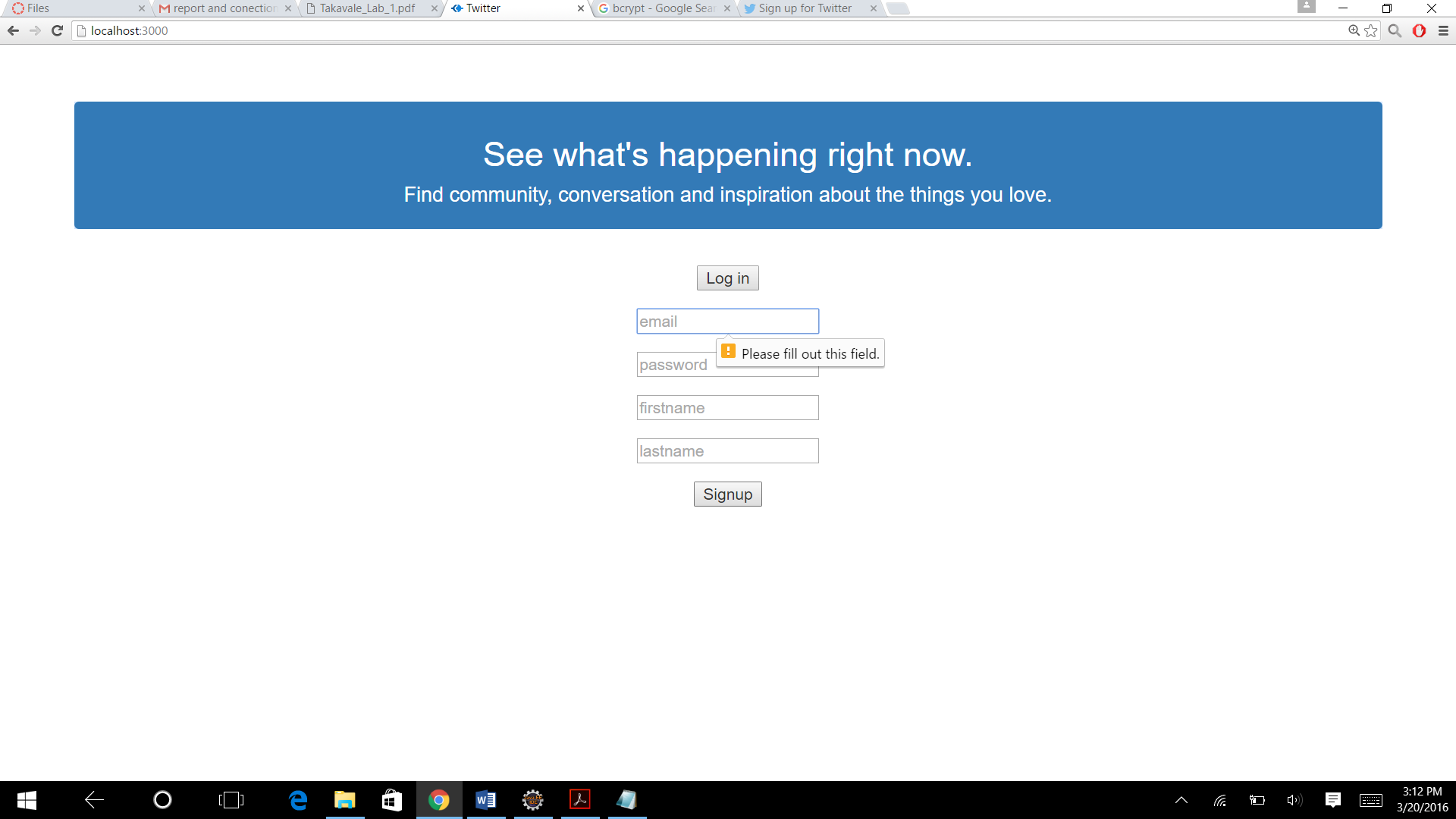
**Signup**

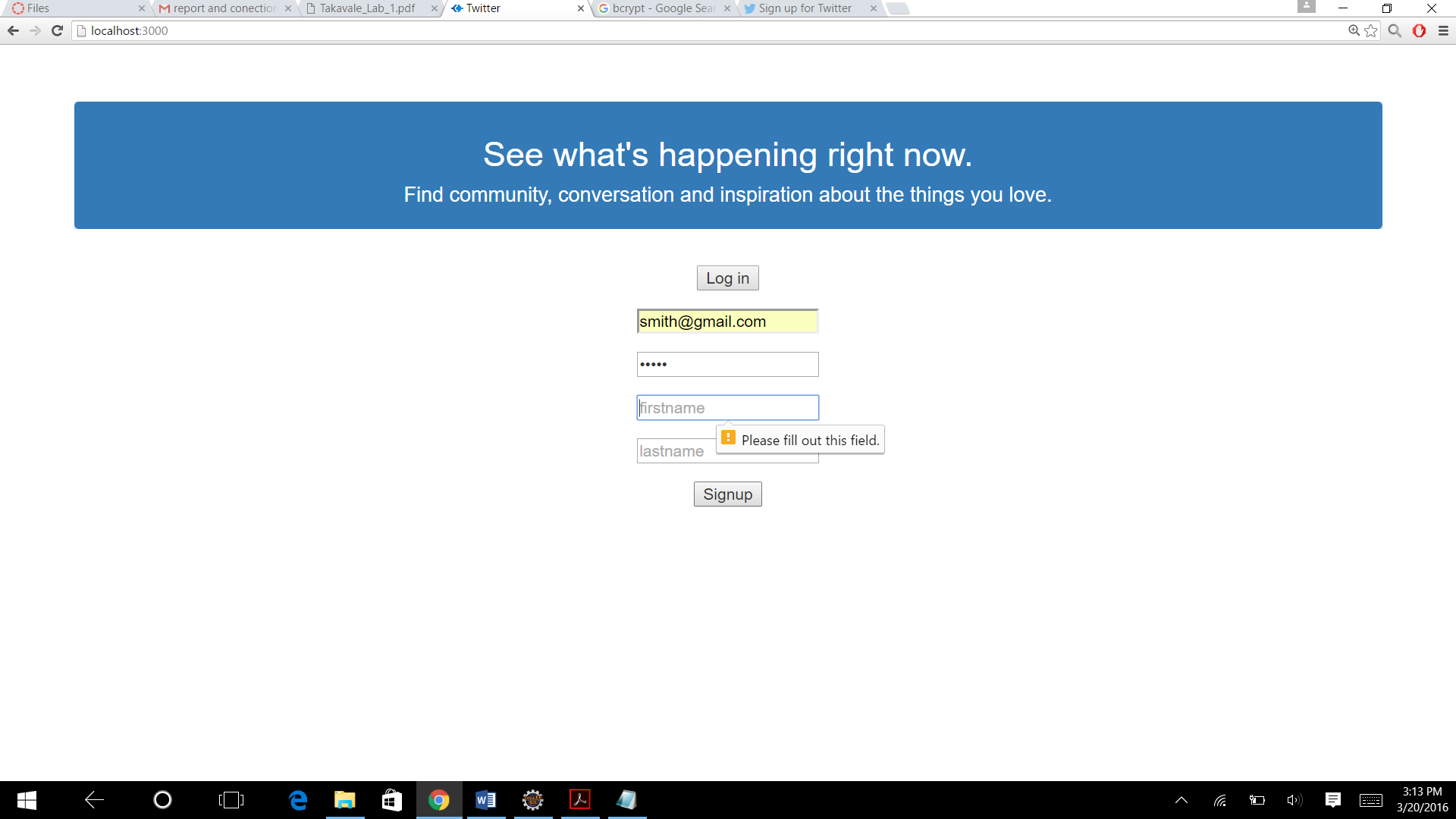
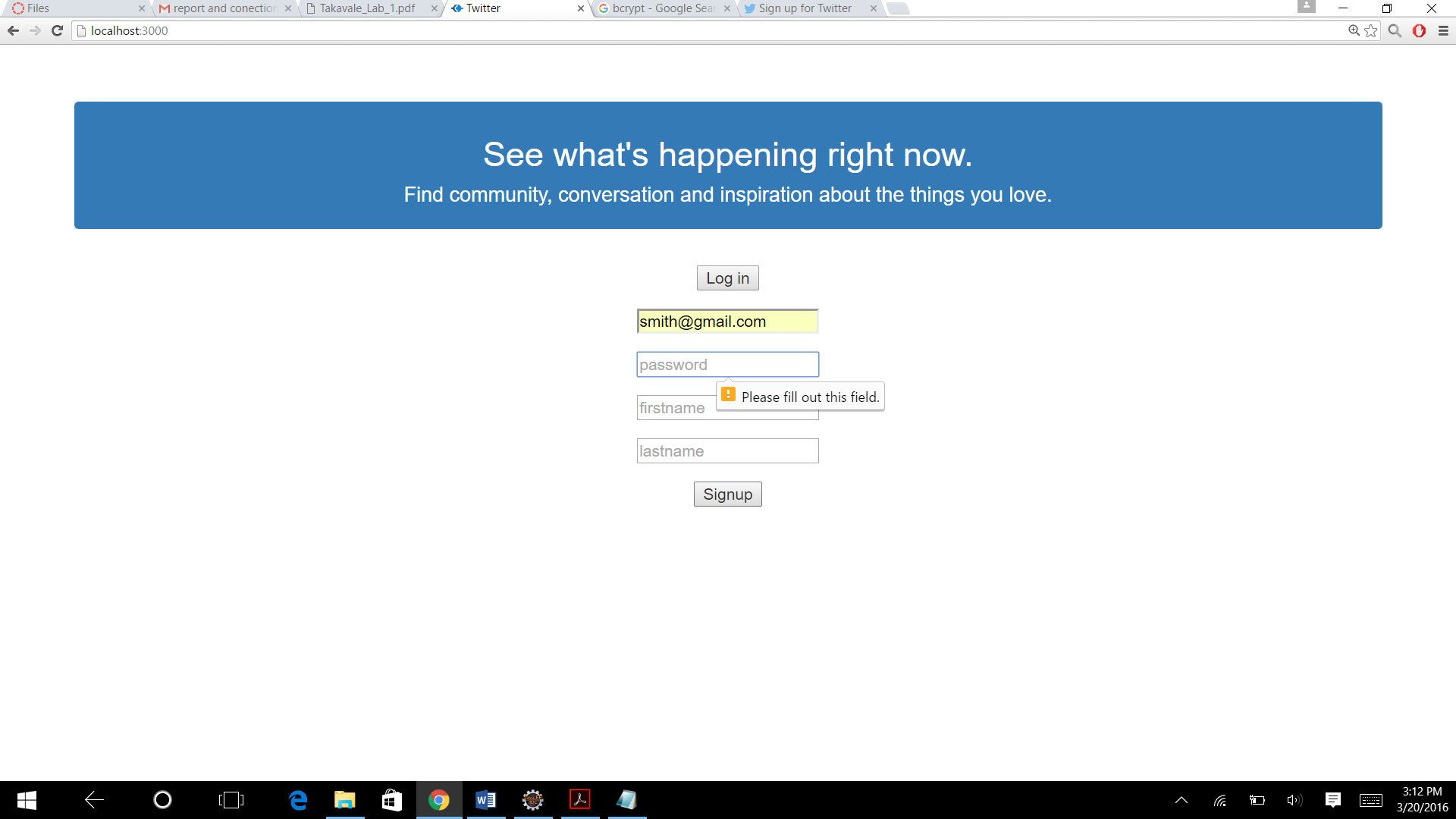


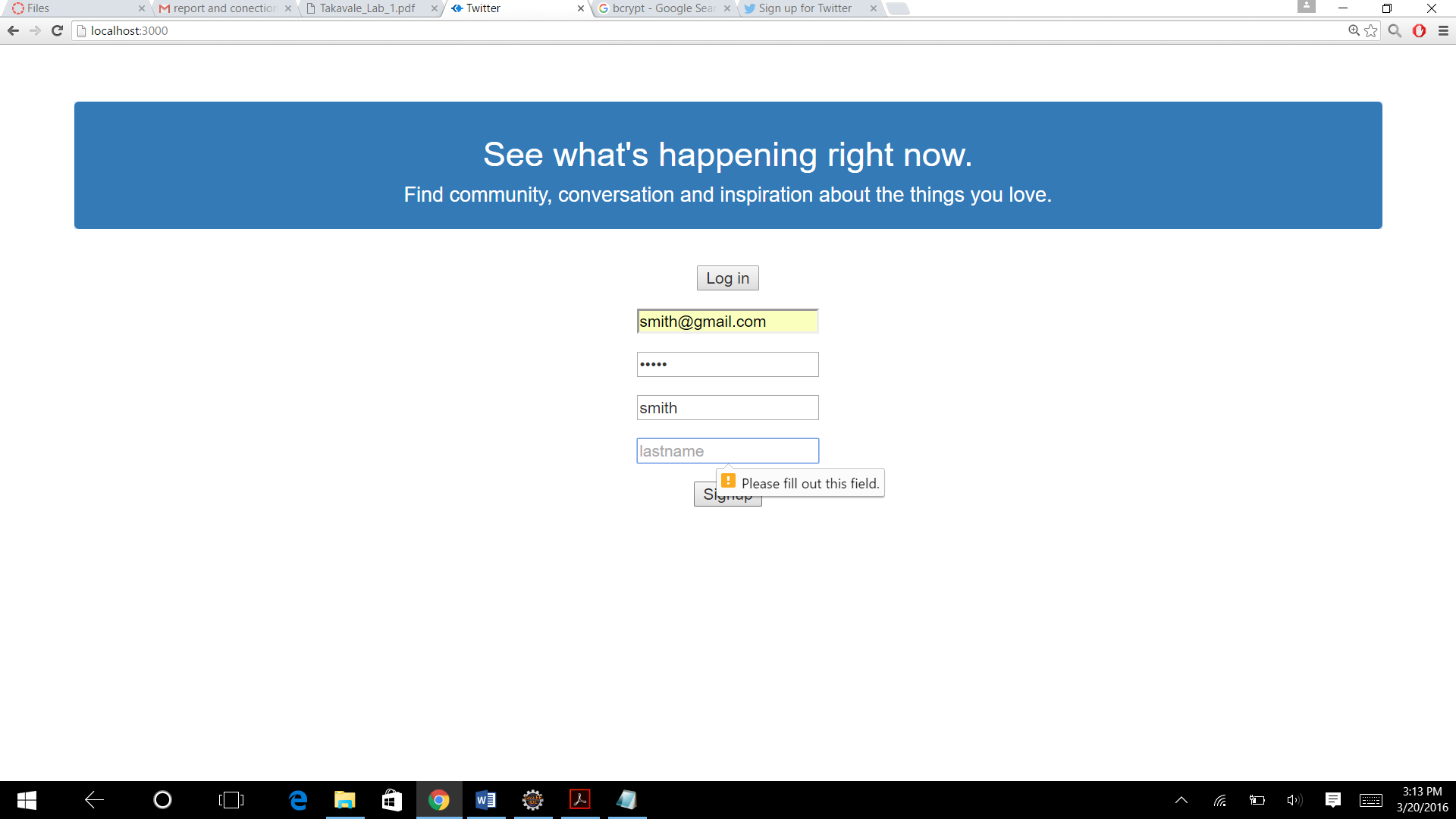
**Entering details:**



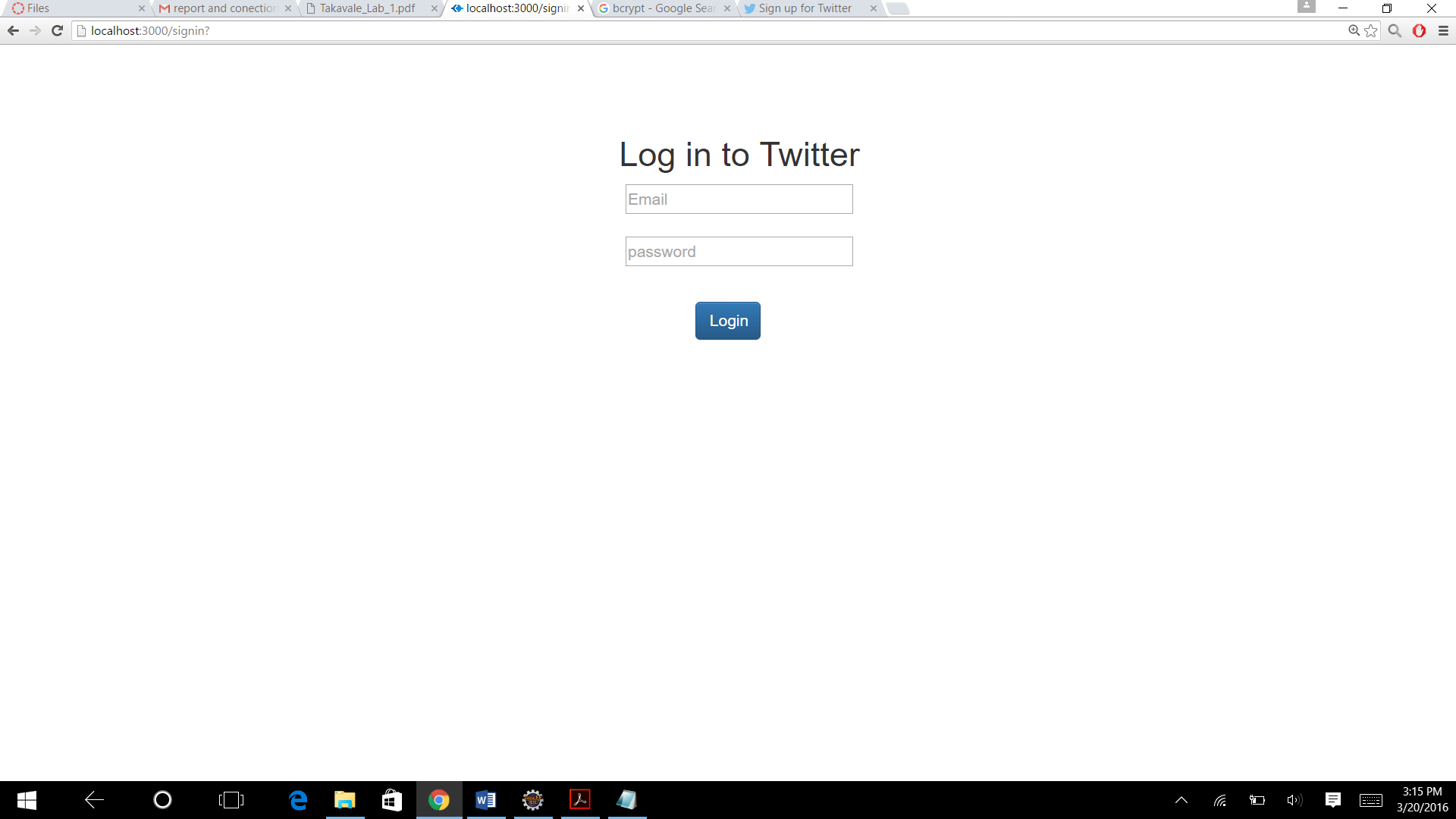
**Error handling:**

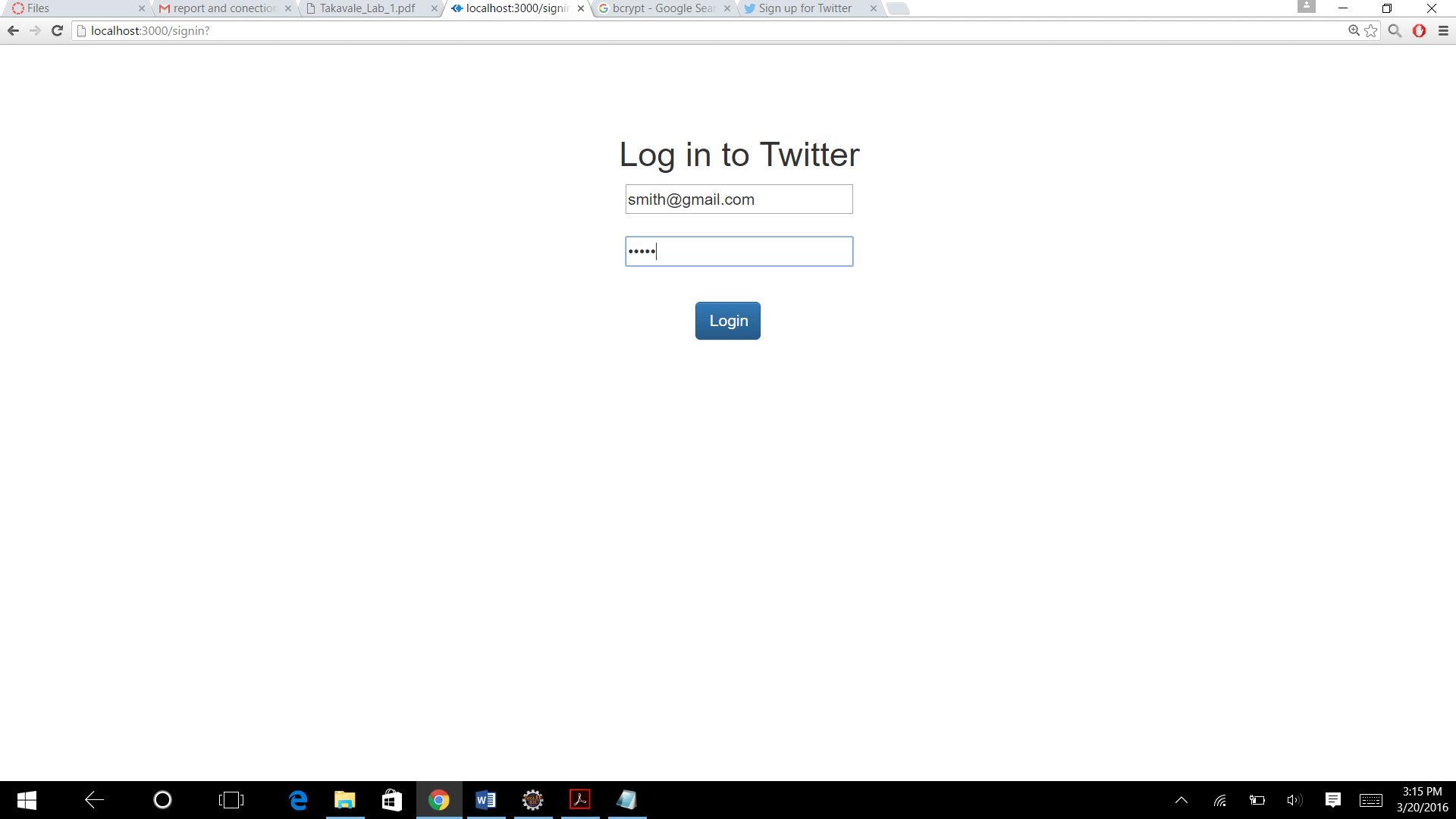




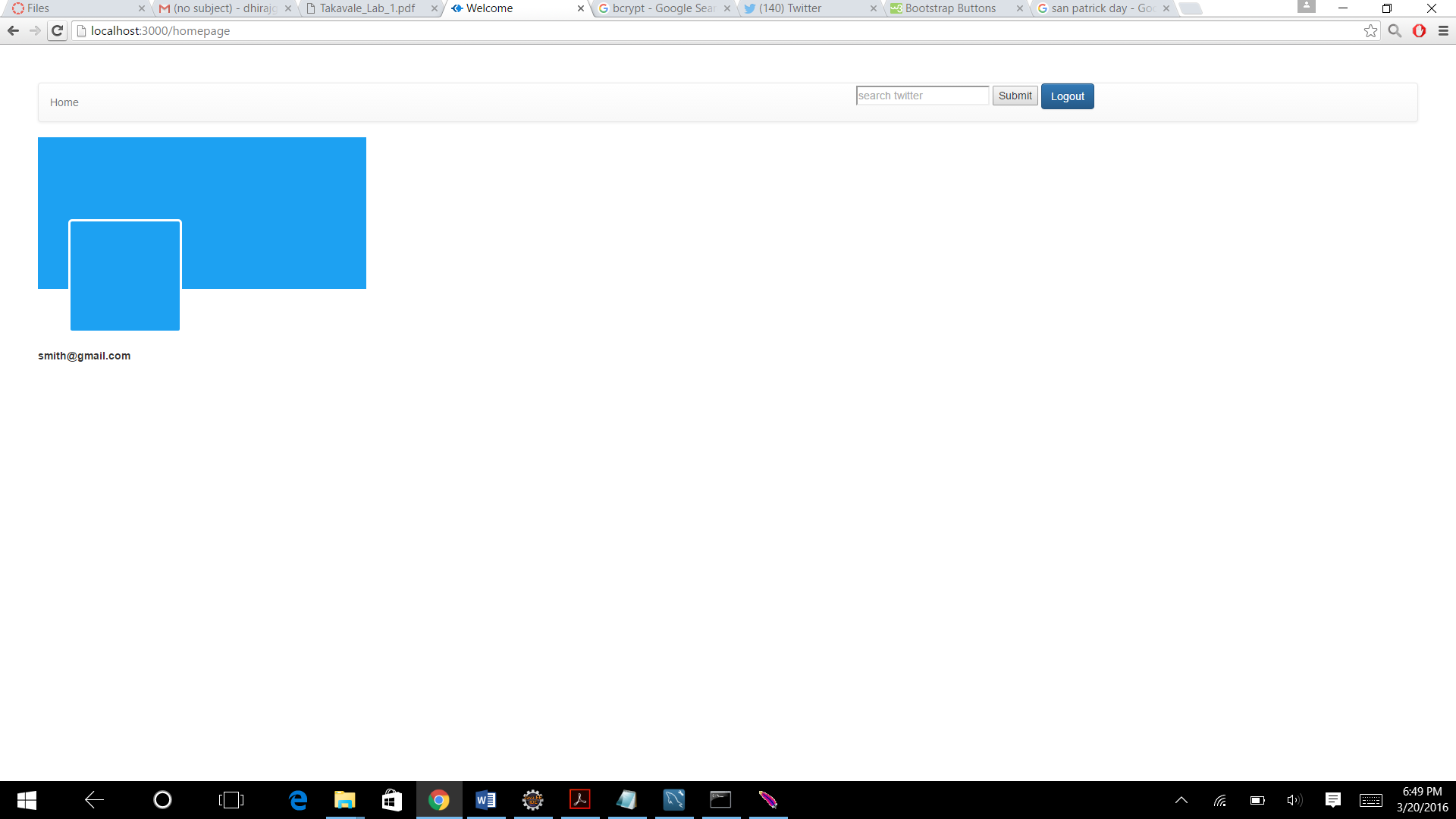


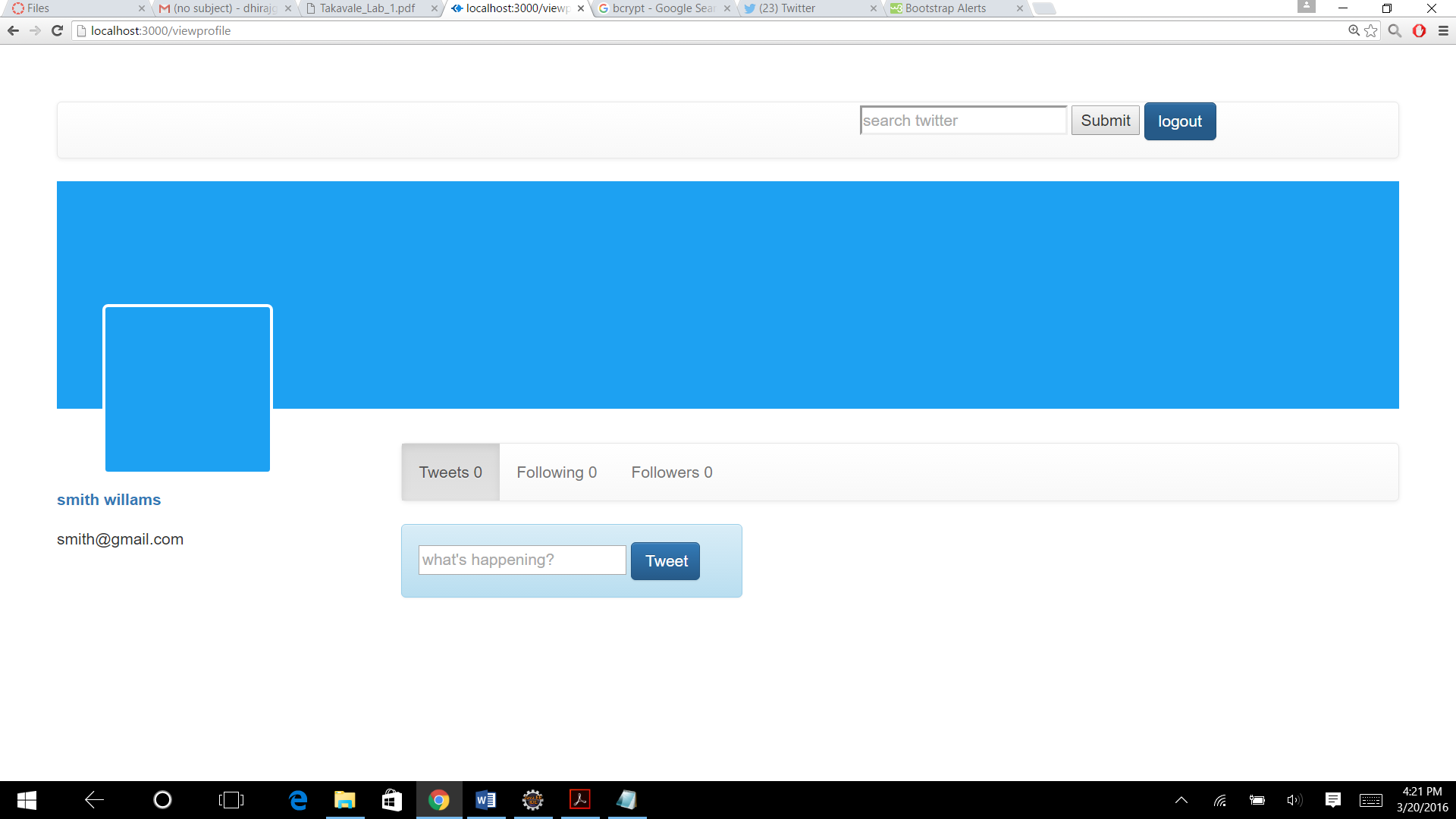
**Log in**



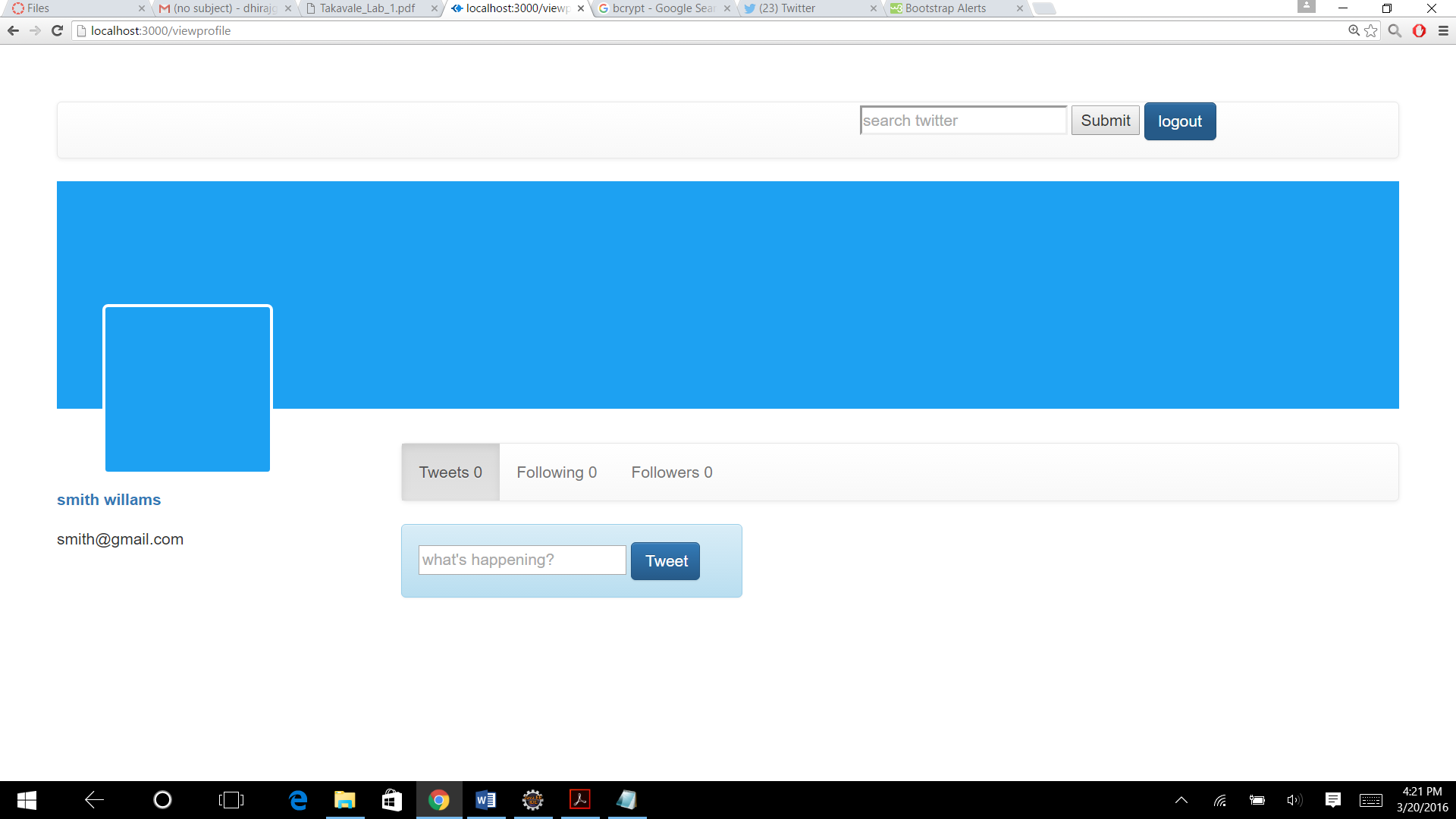


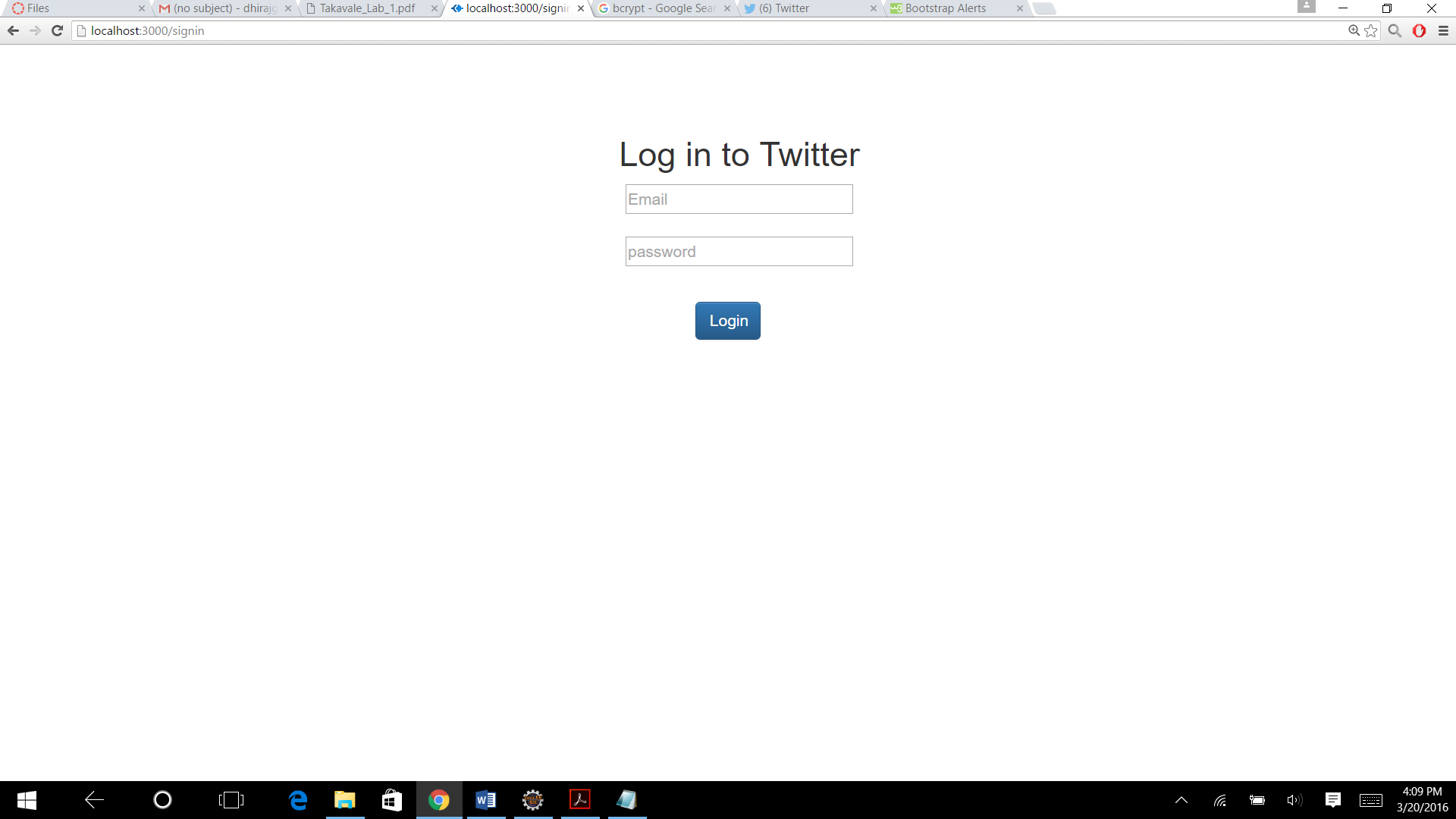
**Signin with an existing users**





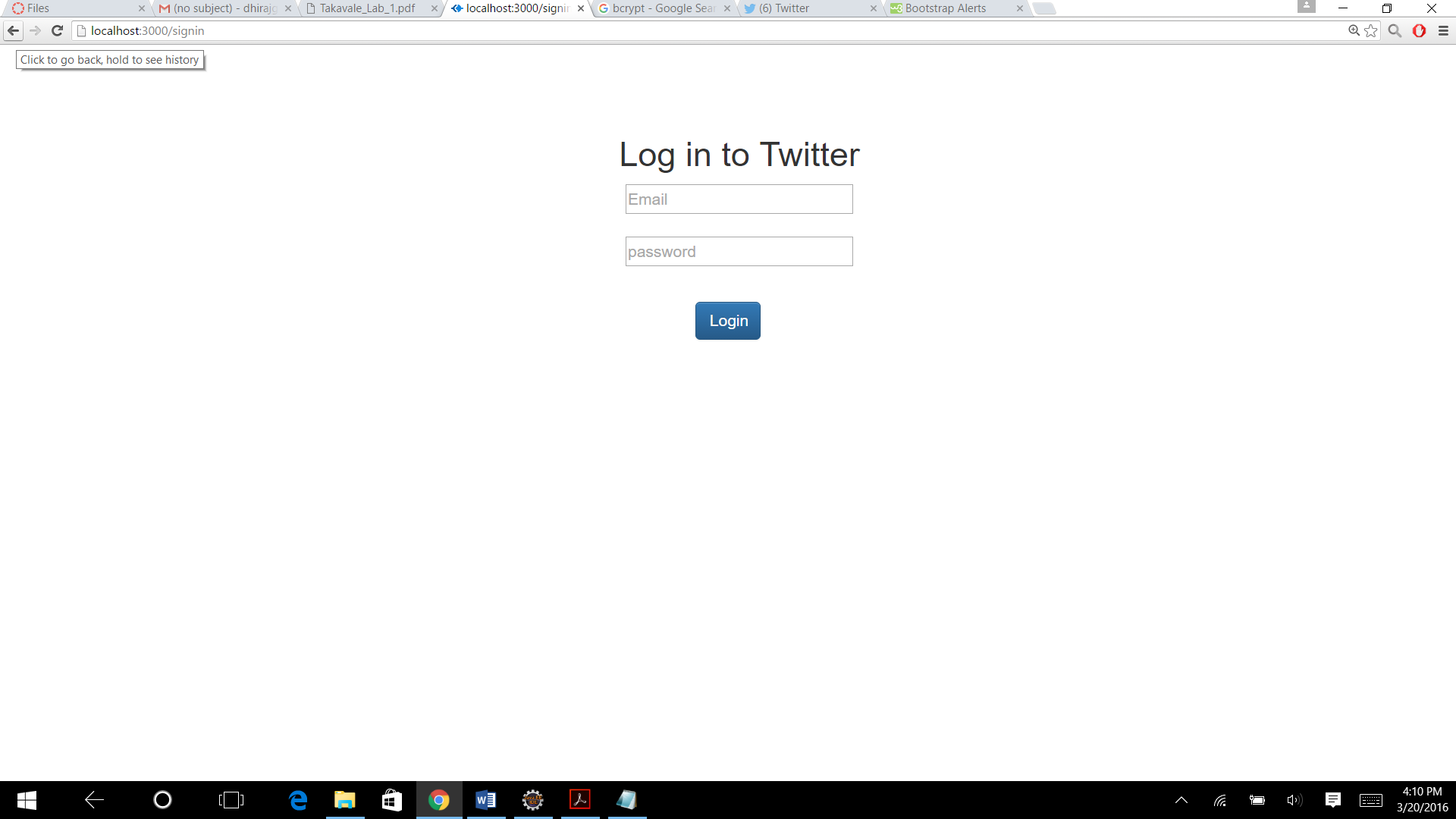
**Logout**

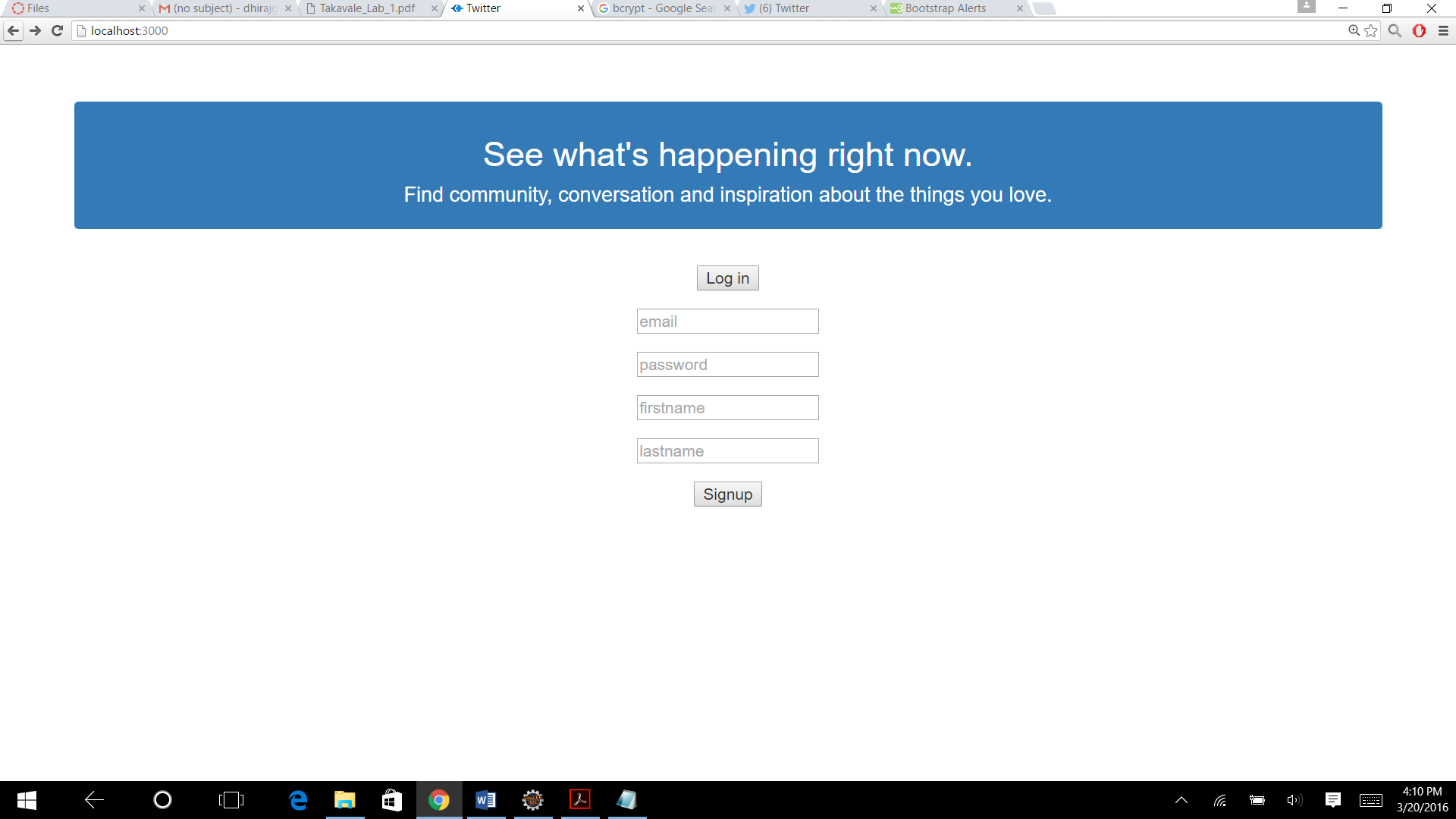




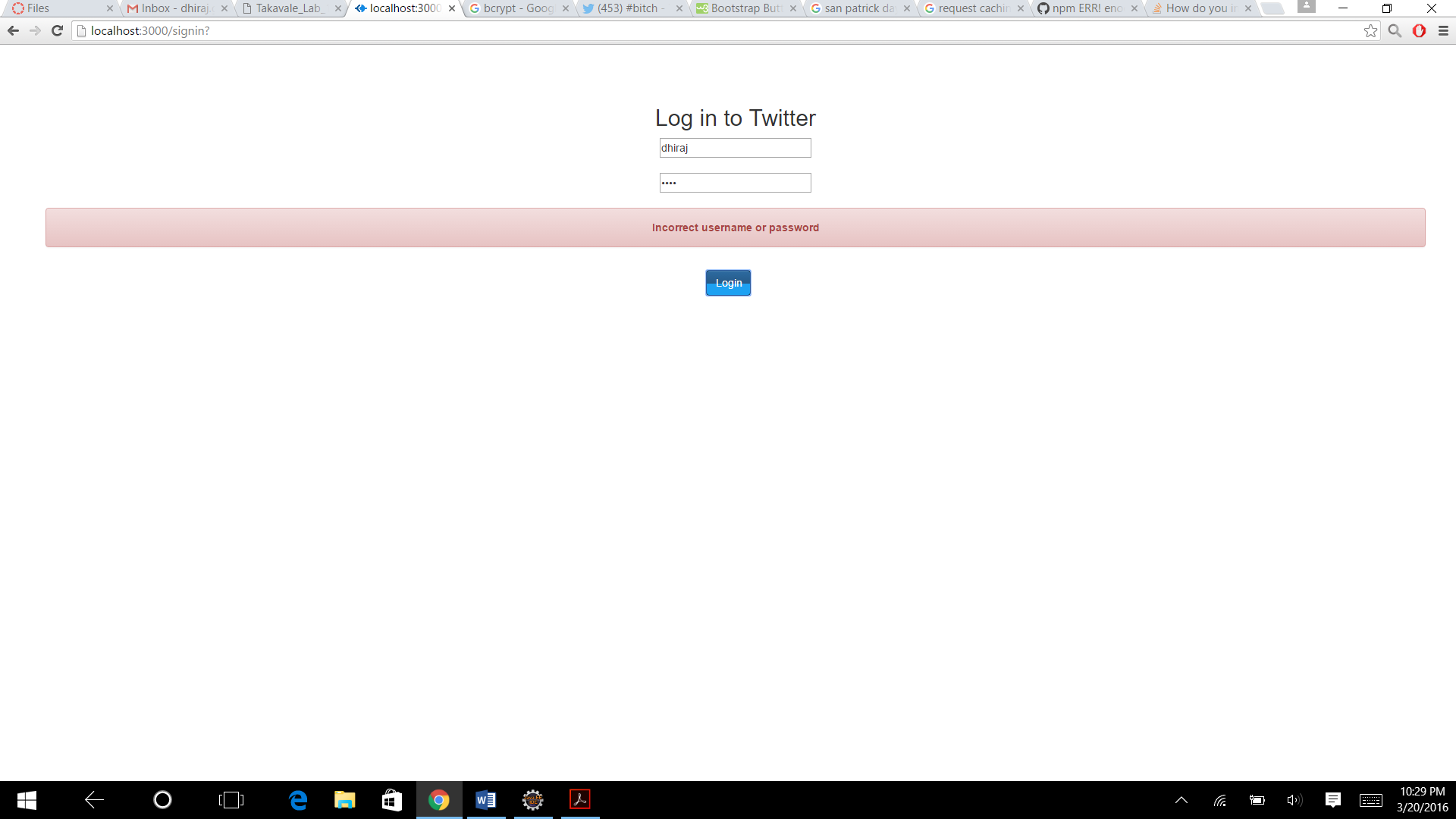
**Session mangment:**

**Pressing back button**

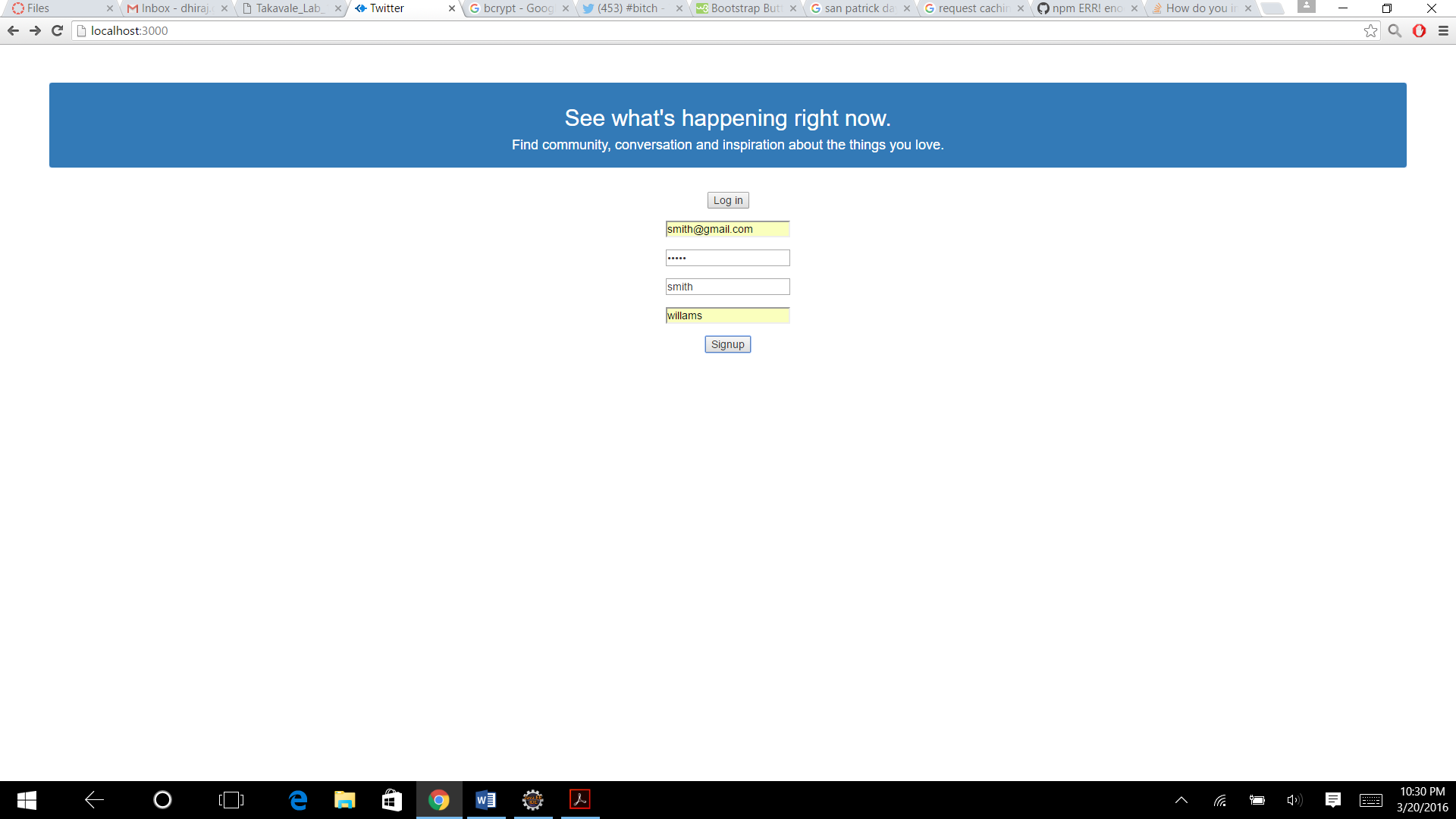




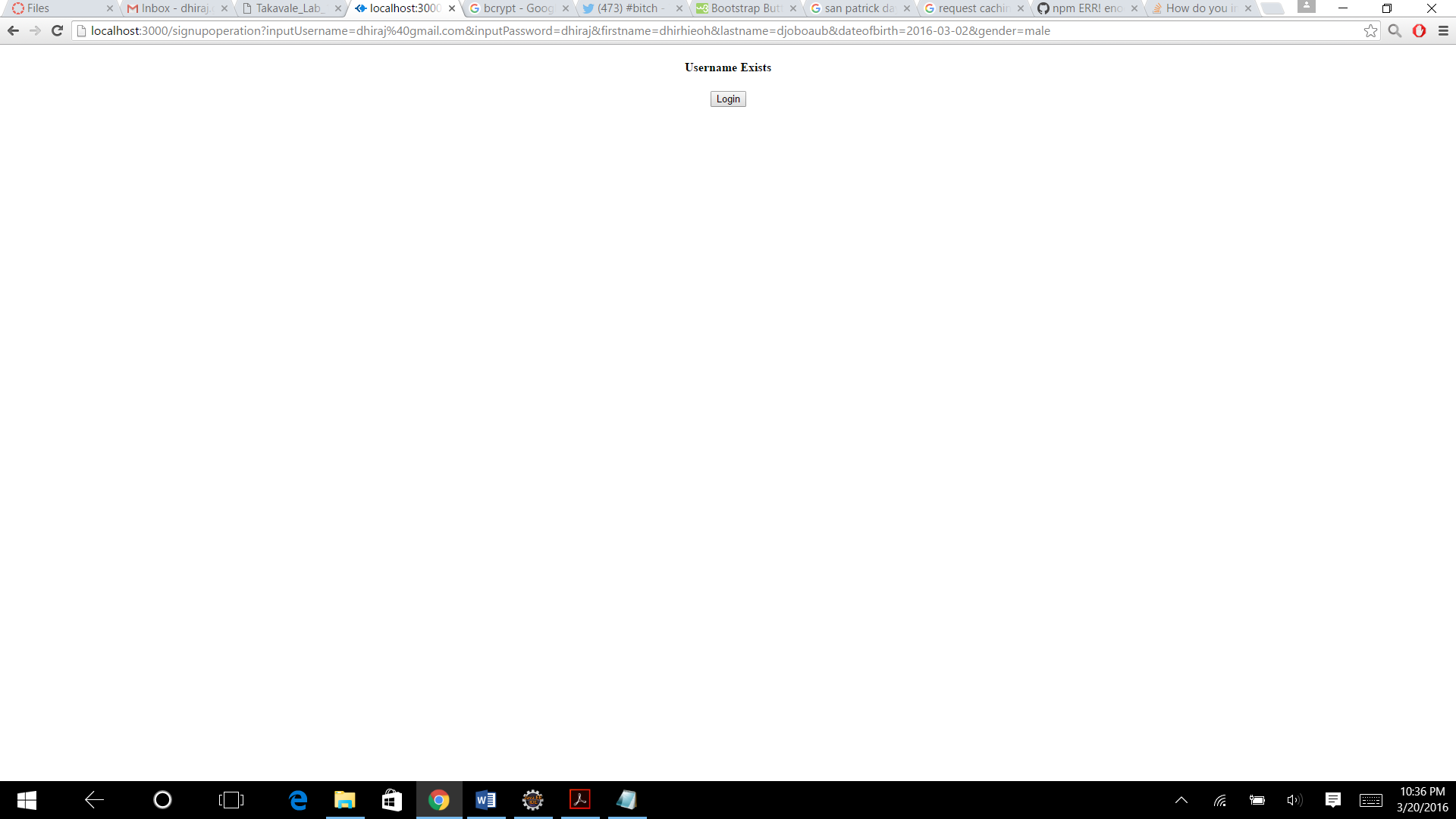
**If the username is not valid:**



**Signing up with the same username**

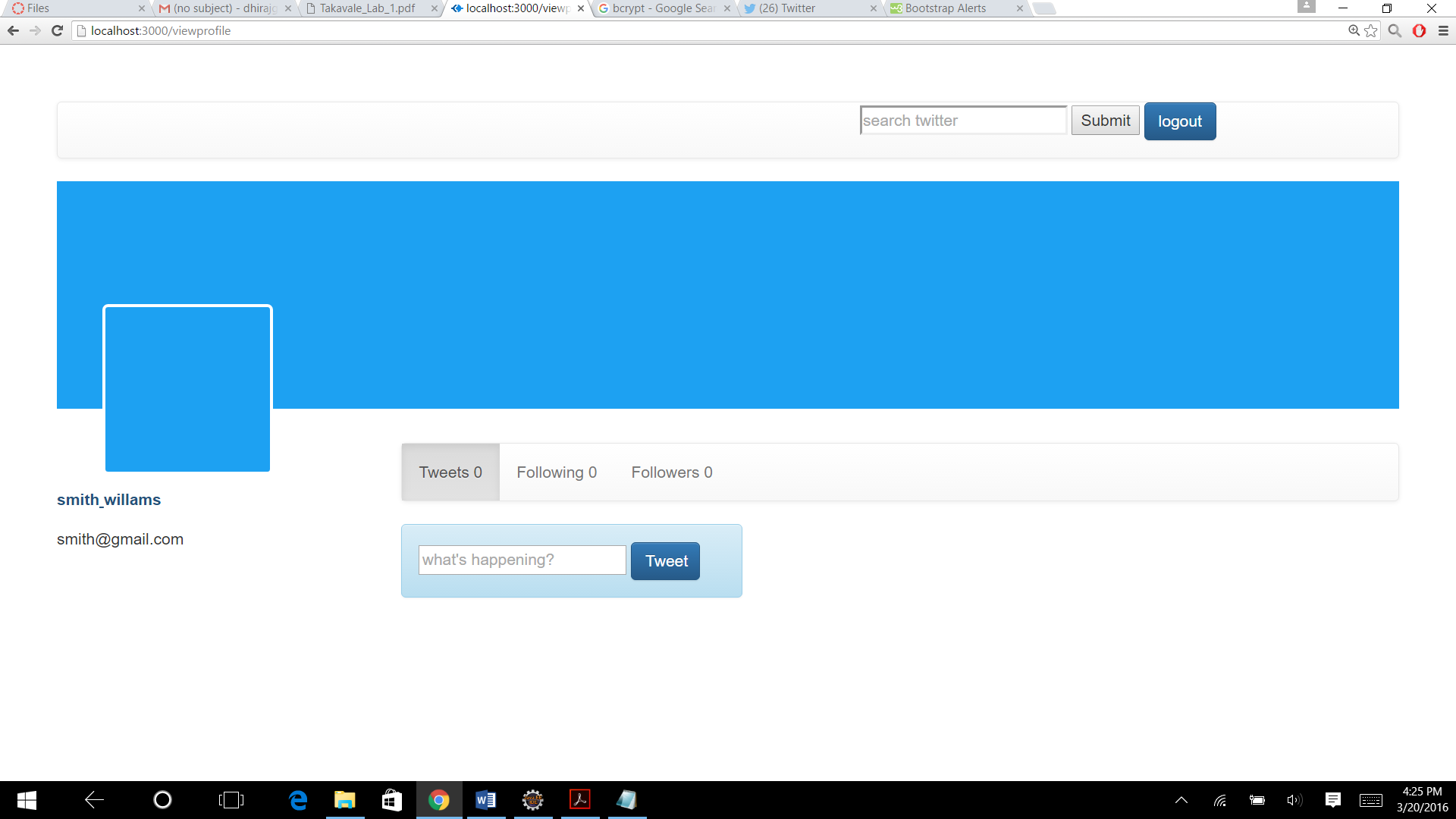


**Error displaying:**

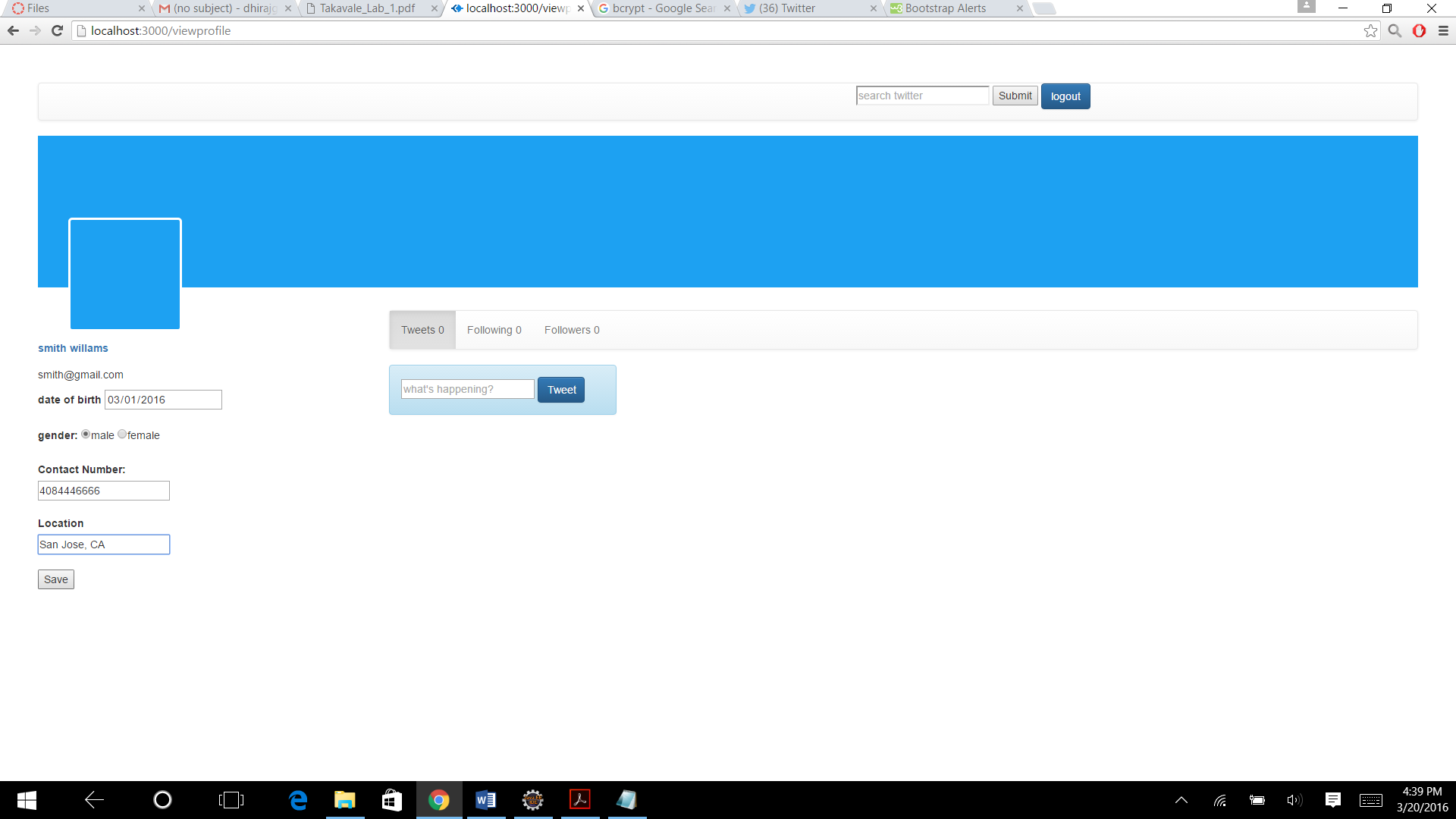


**2.profile**

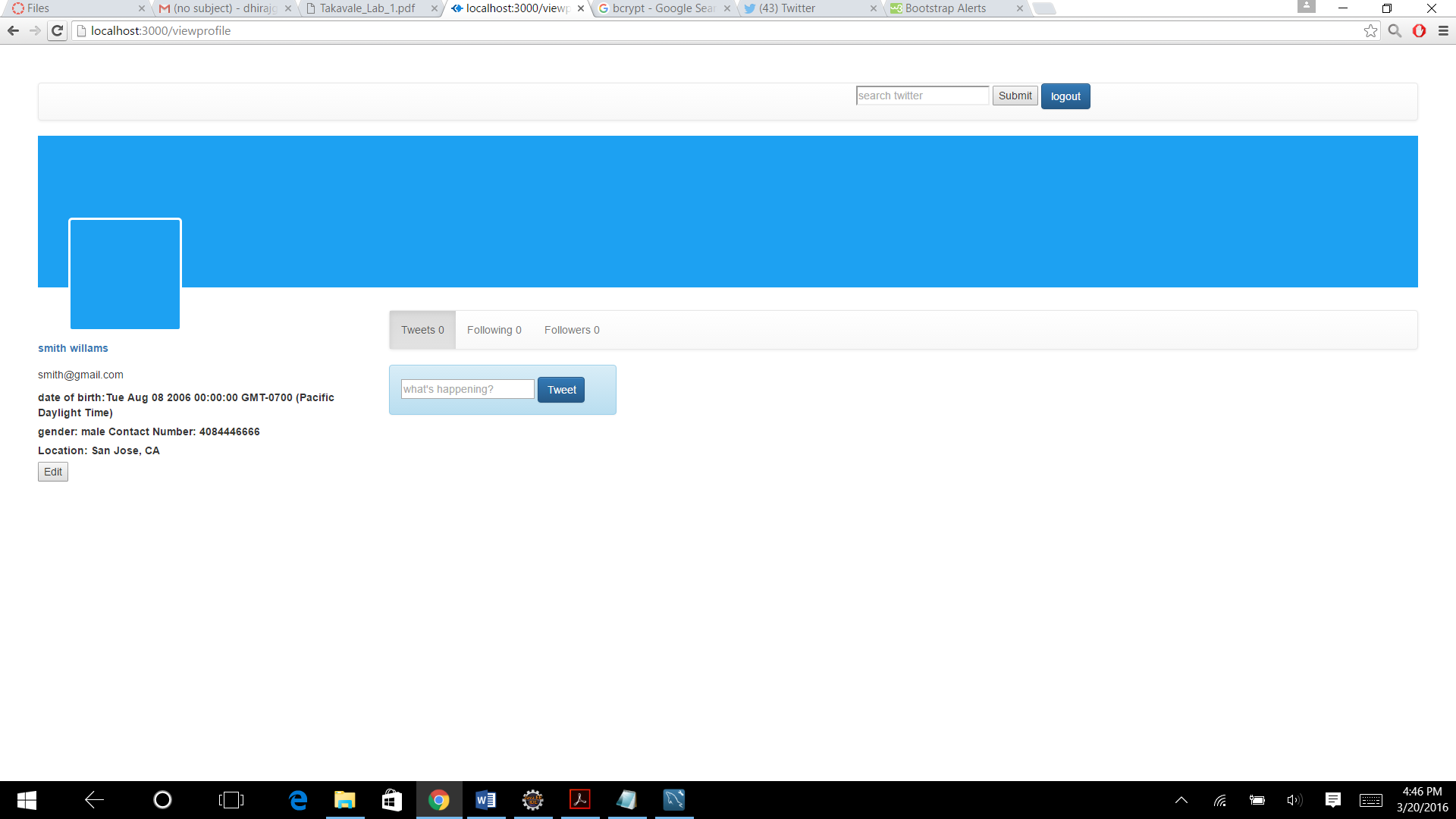
**About user**



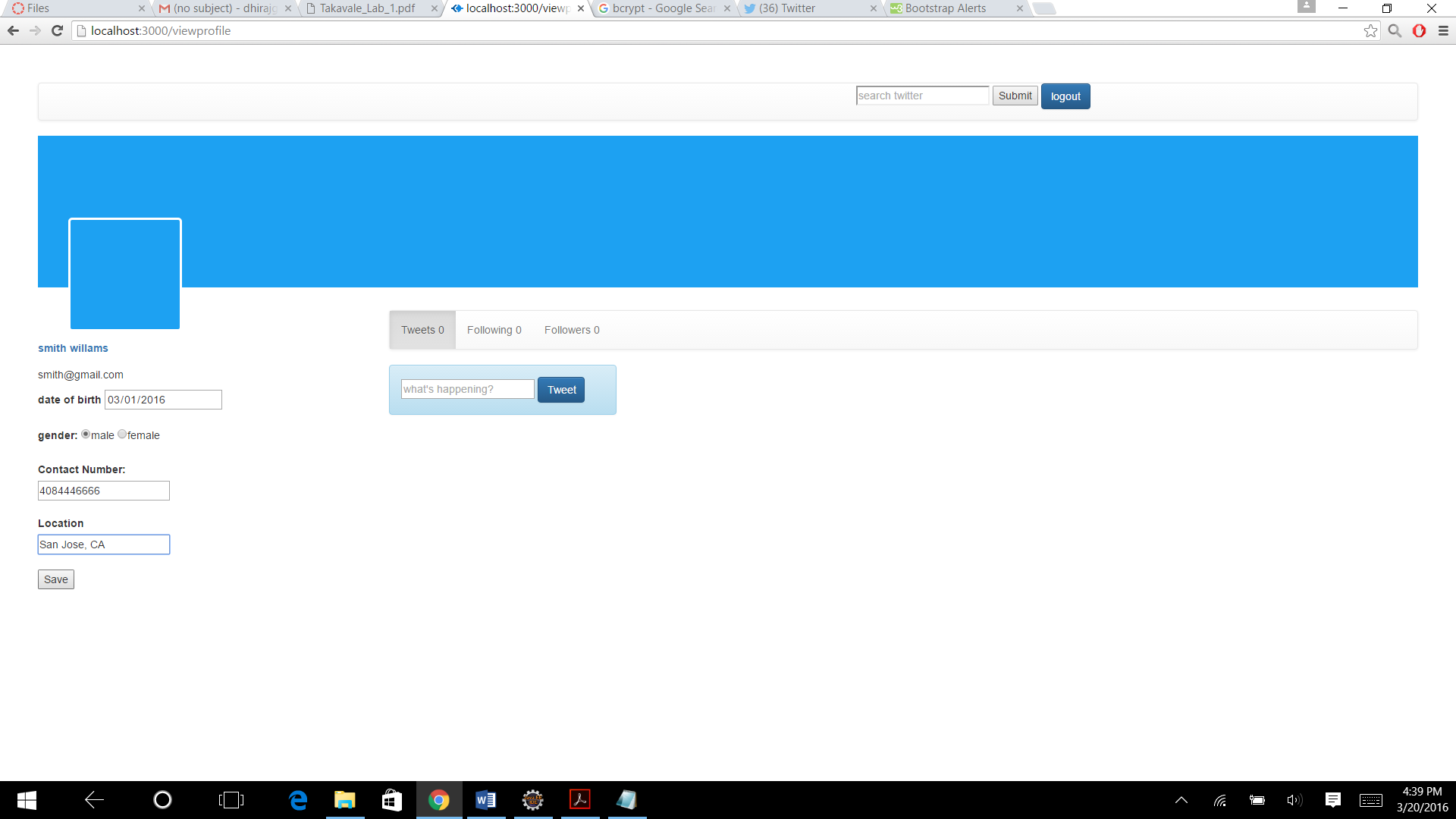
**Saving user info**



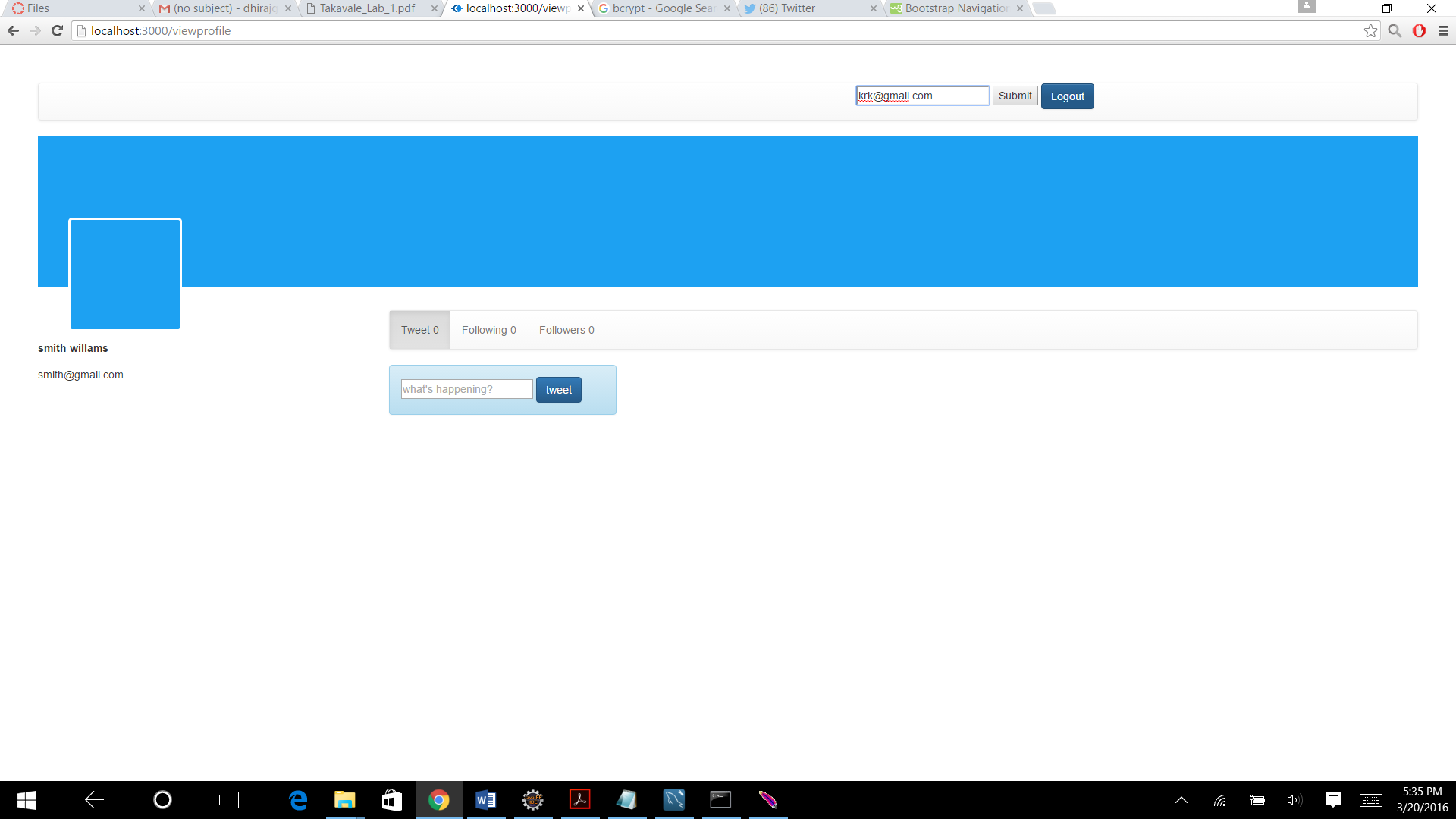
**Saved Data**

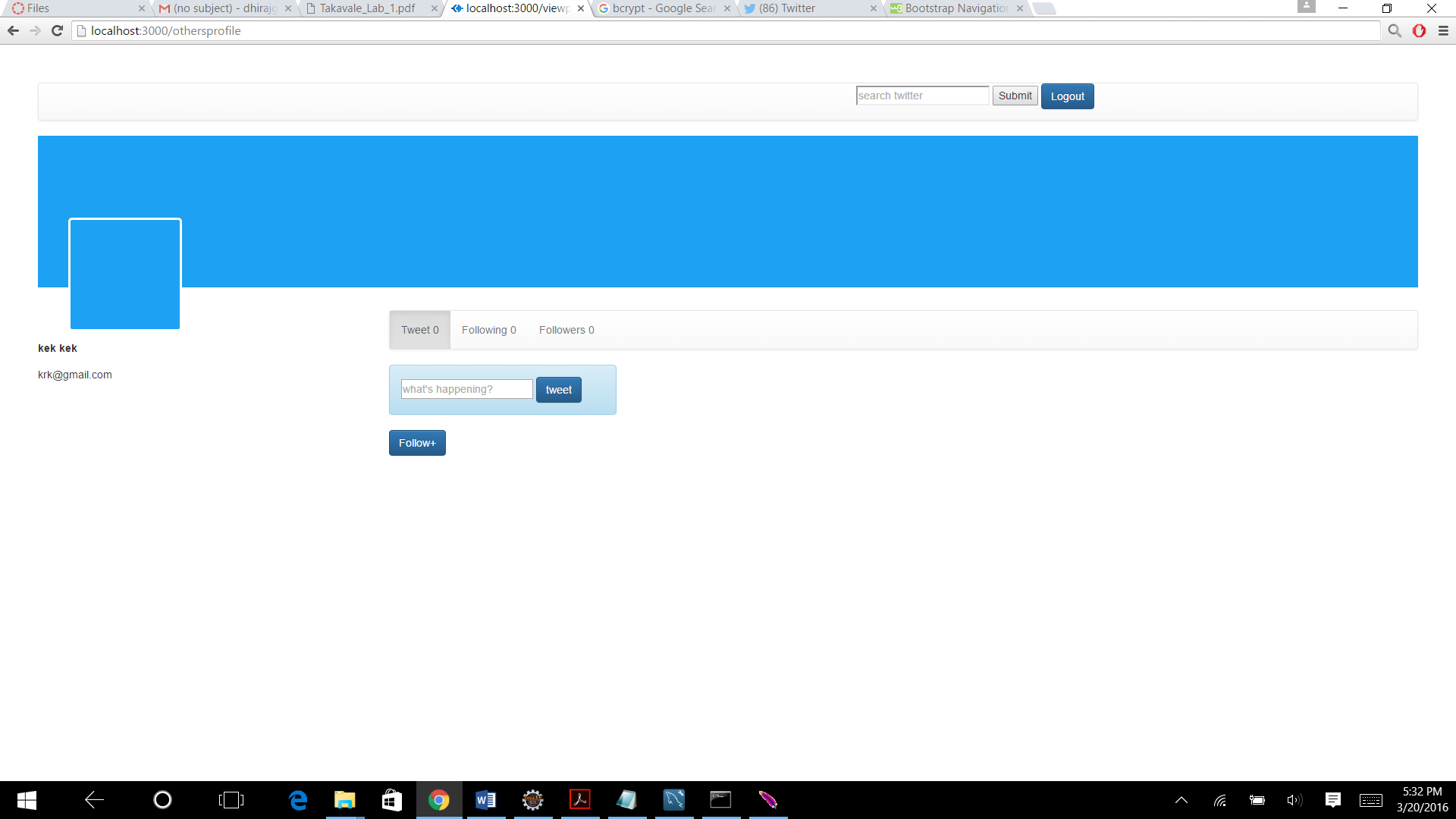


**Editing profile**

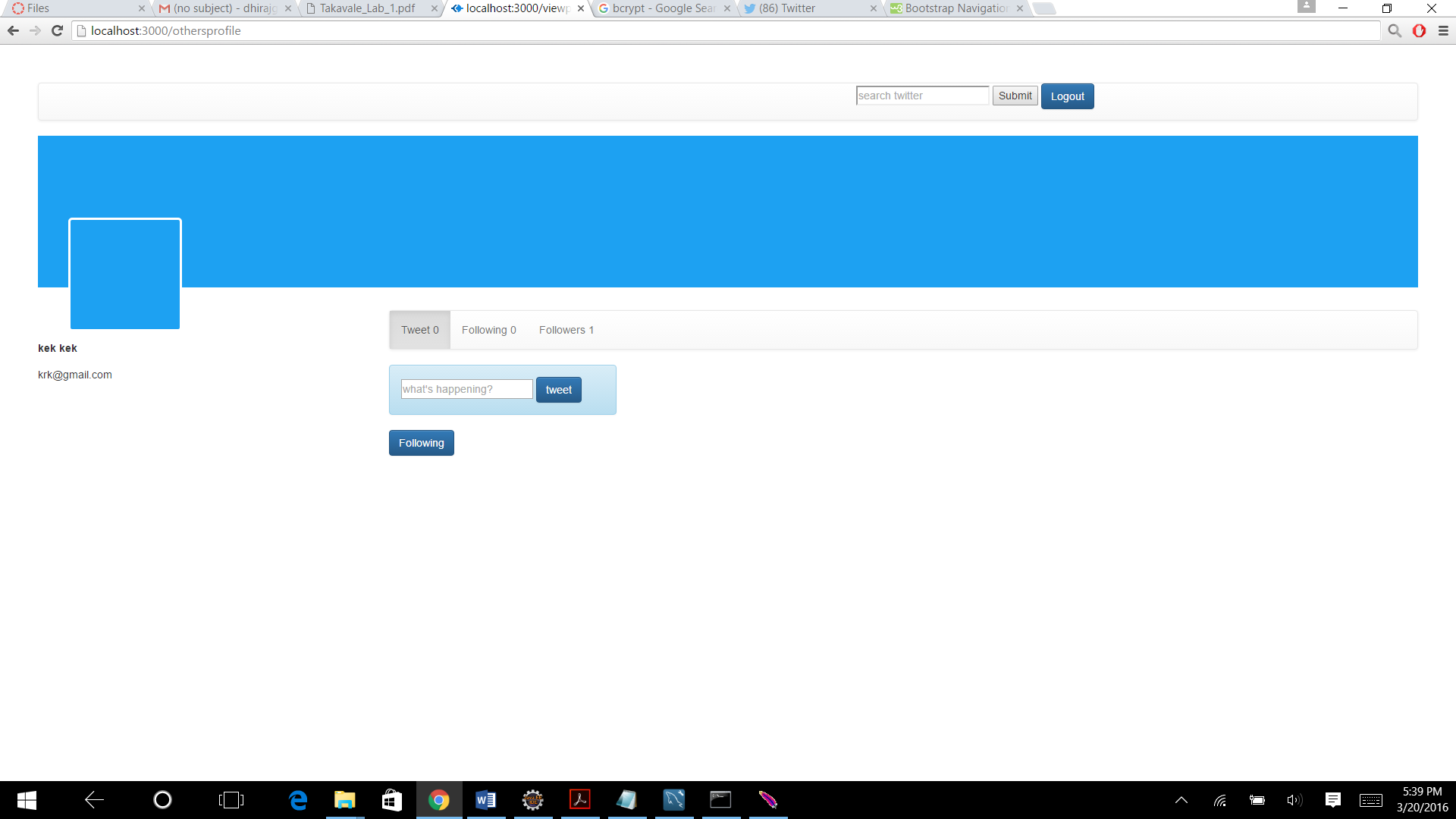


**Othersprofile**

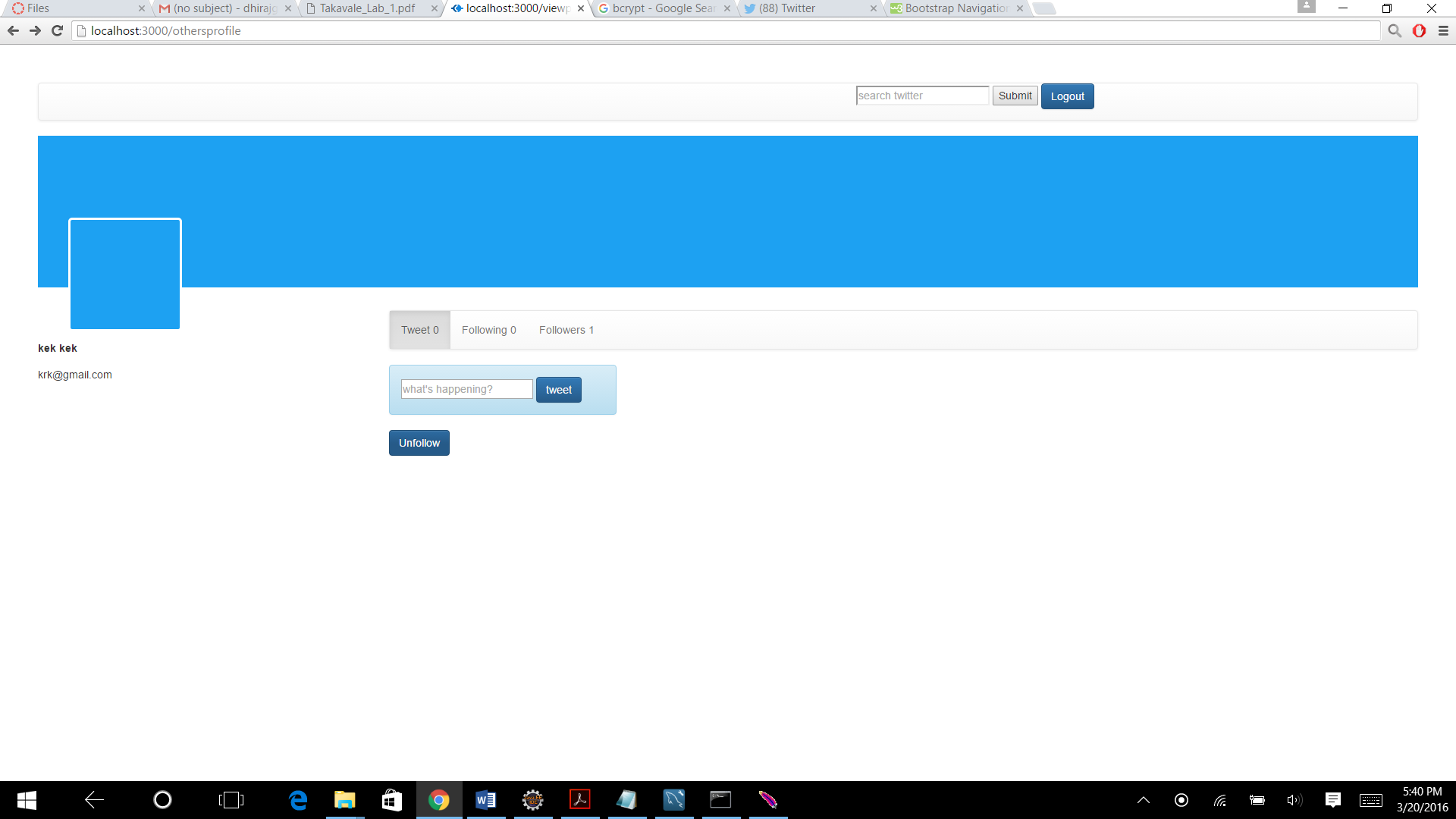




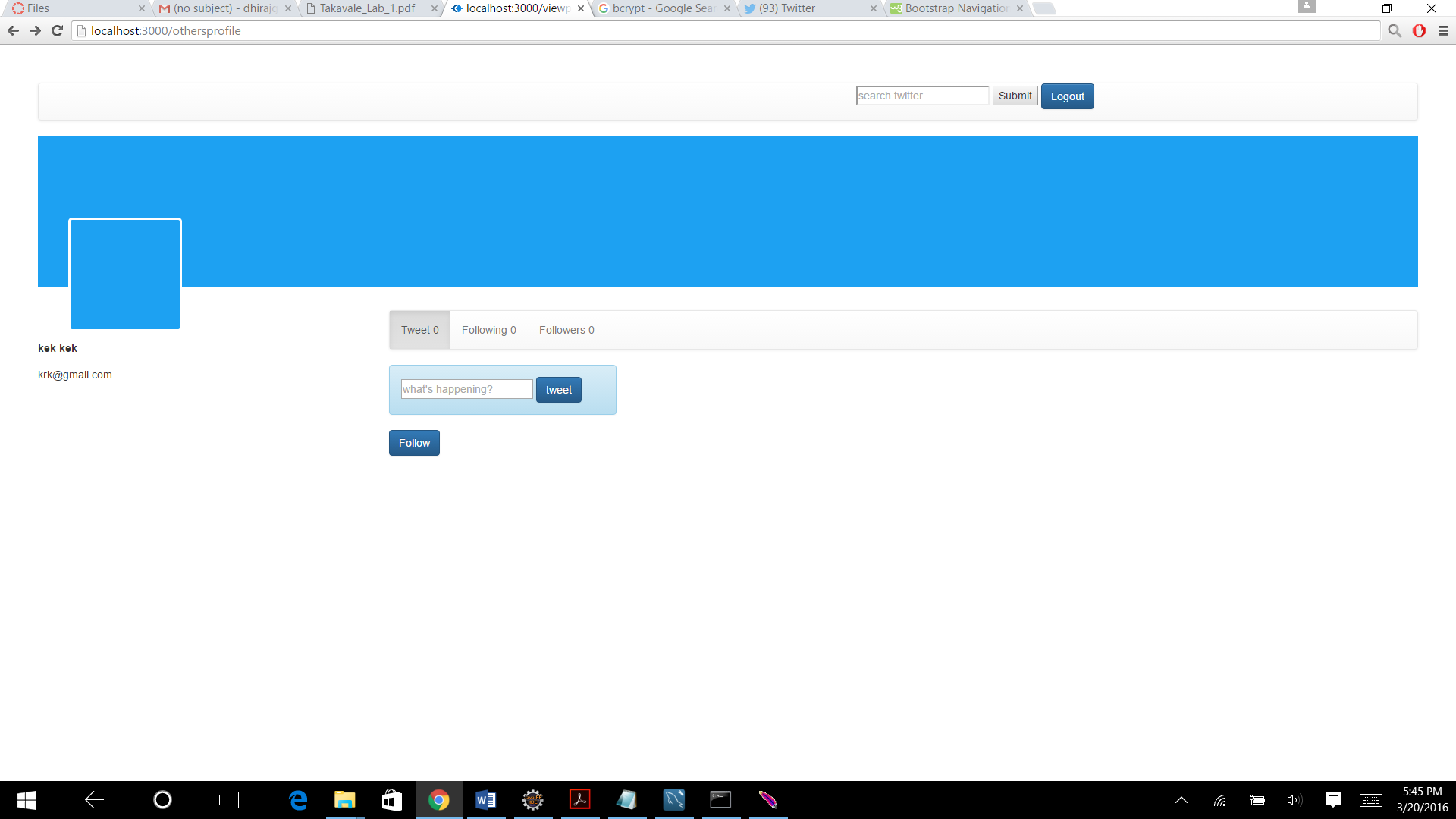
**Follow:**



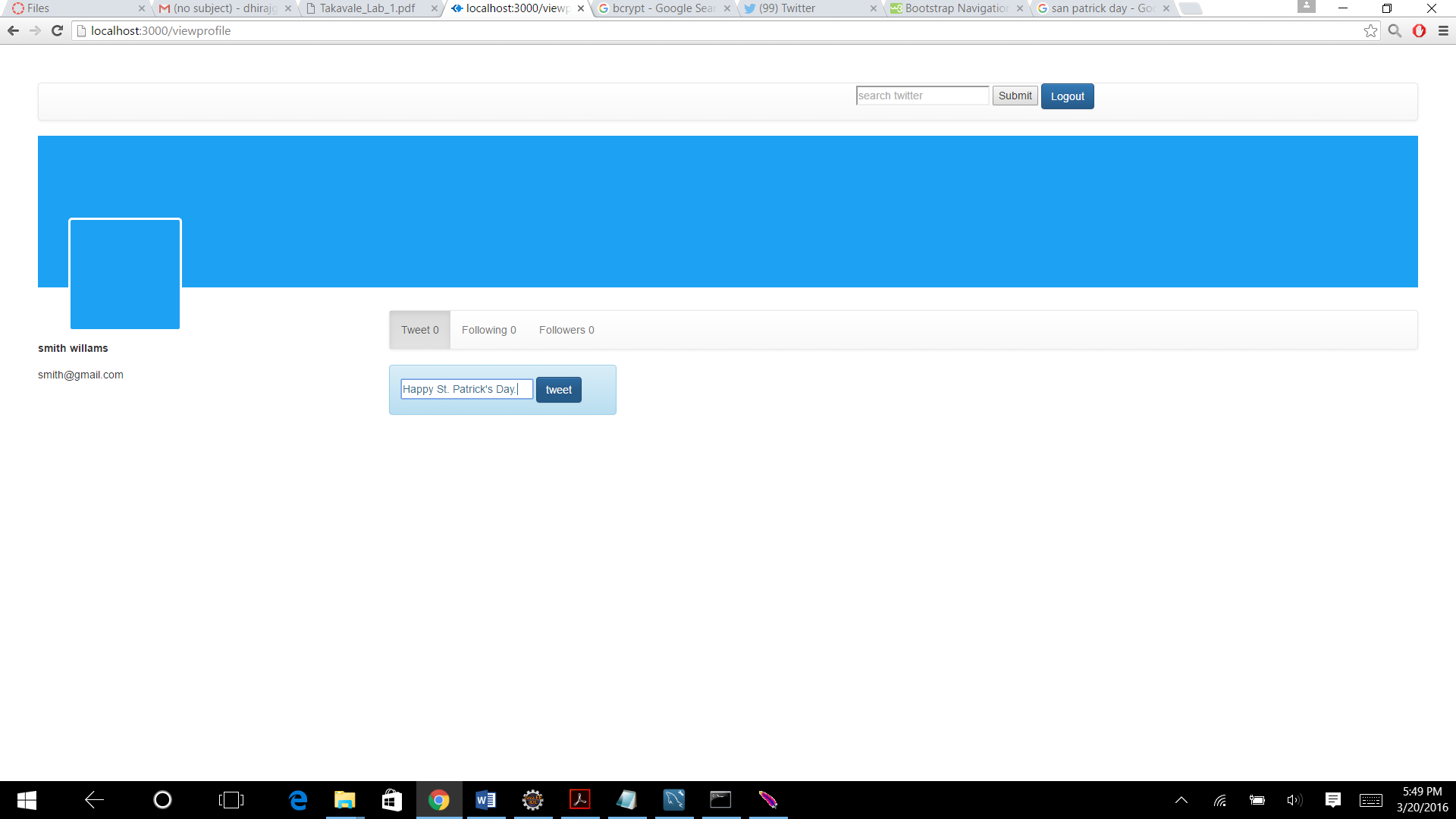
**Hover**

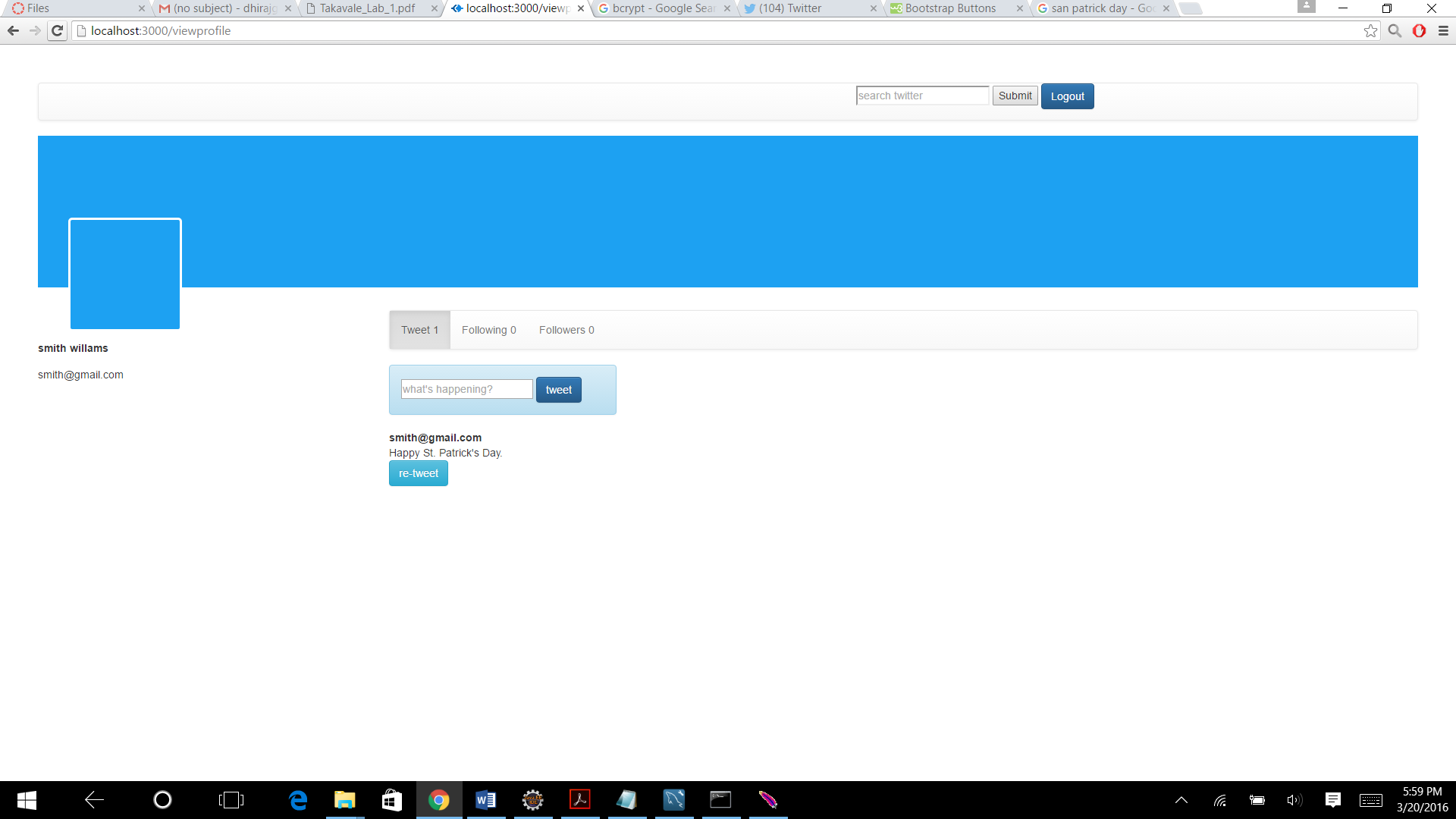


**Pressing Unfollow**

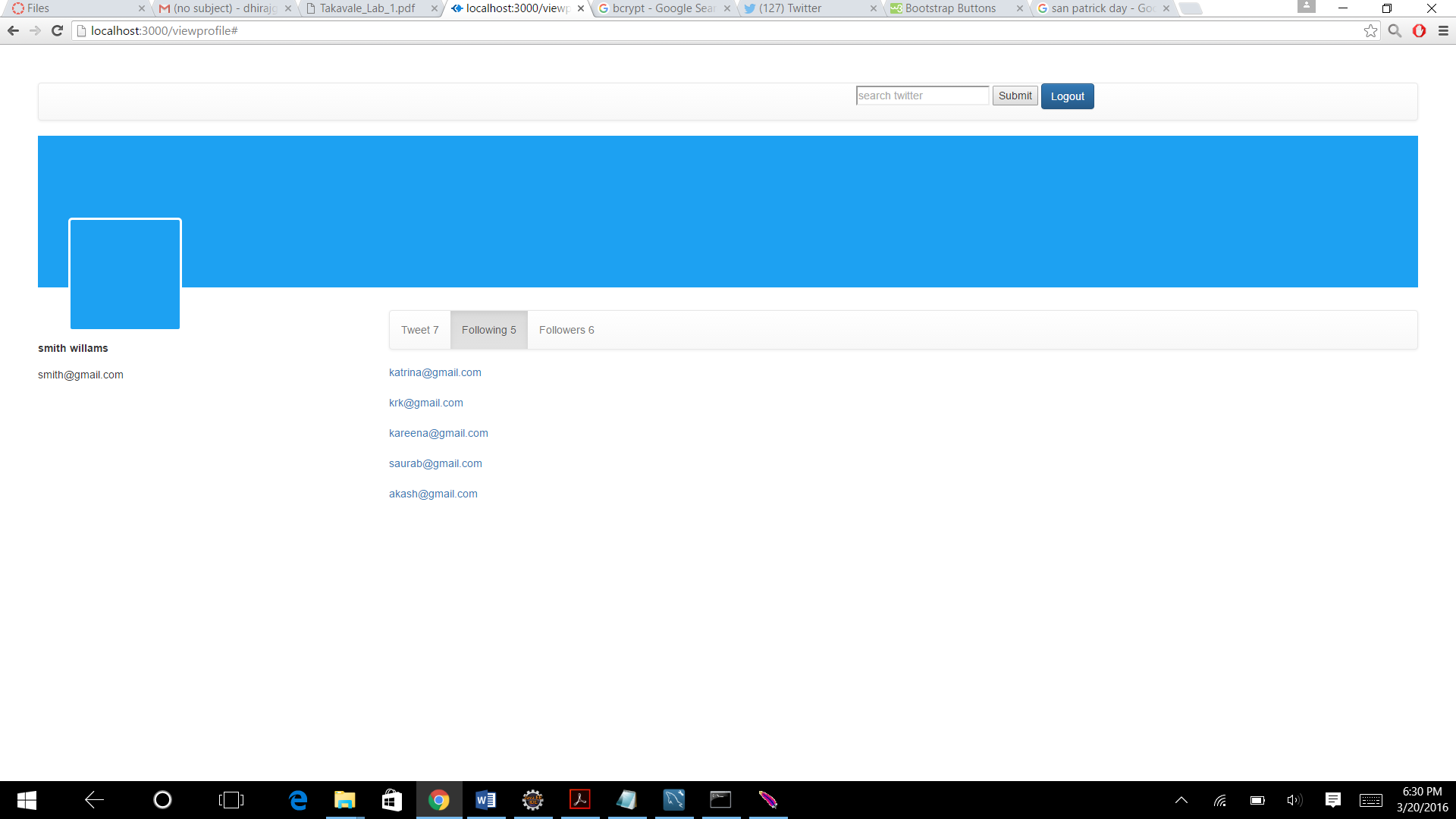


**Tweets:**

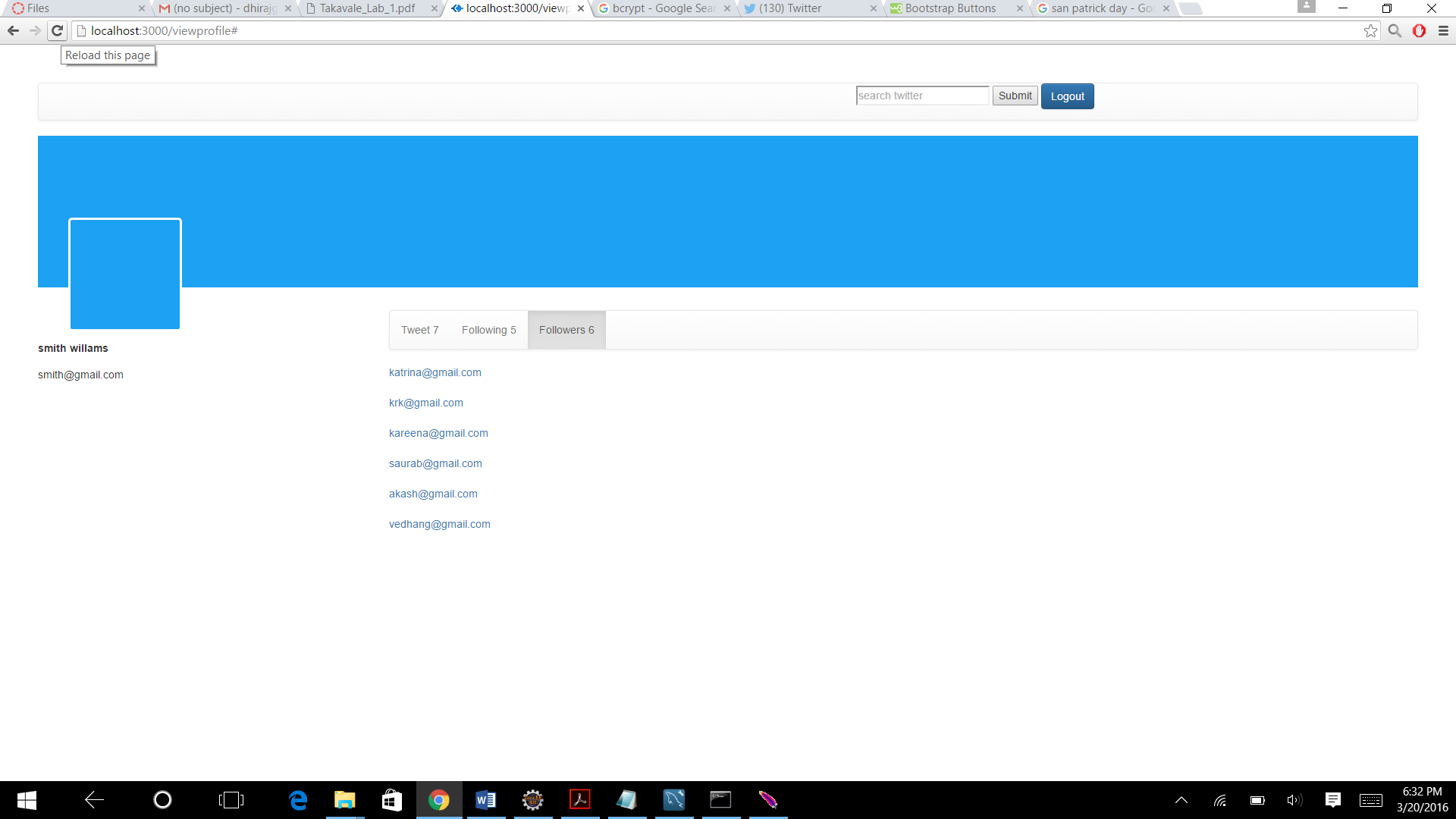




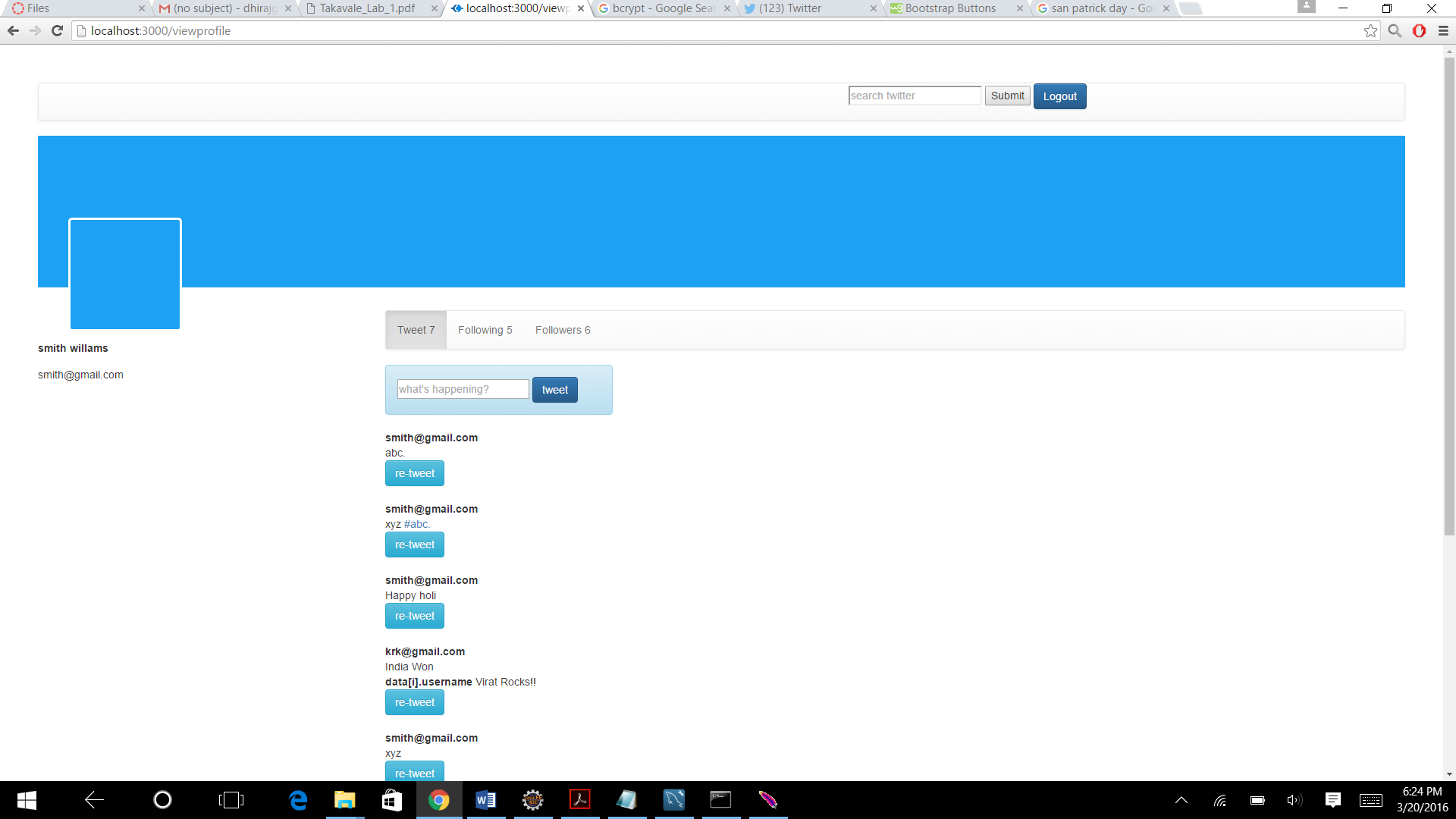
**Following List**

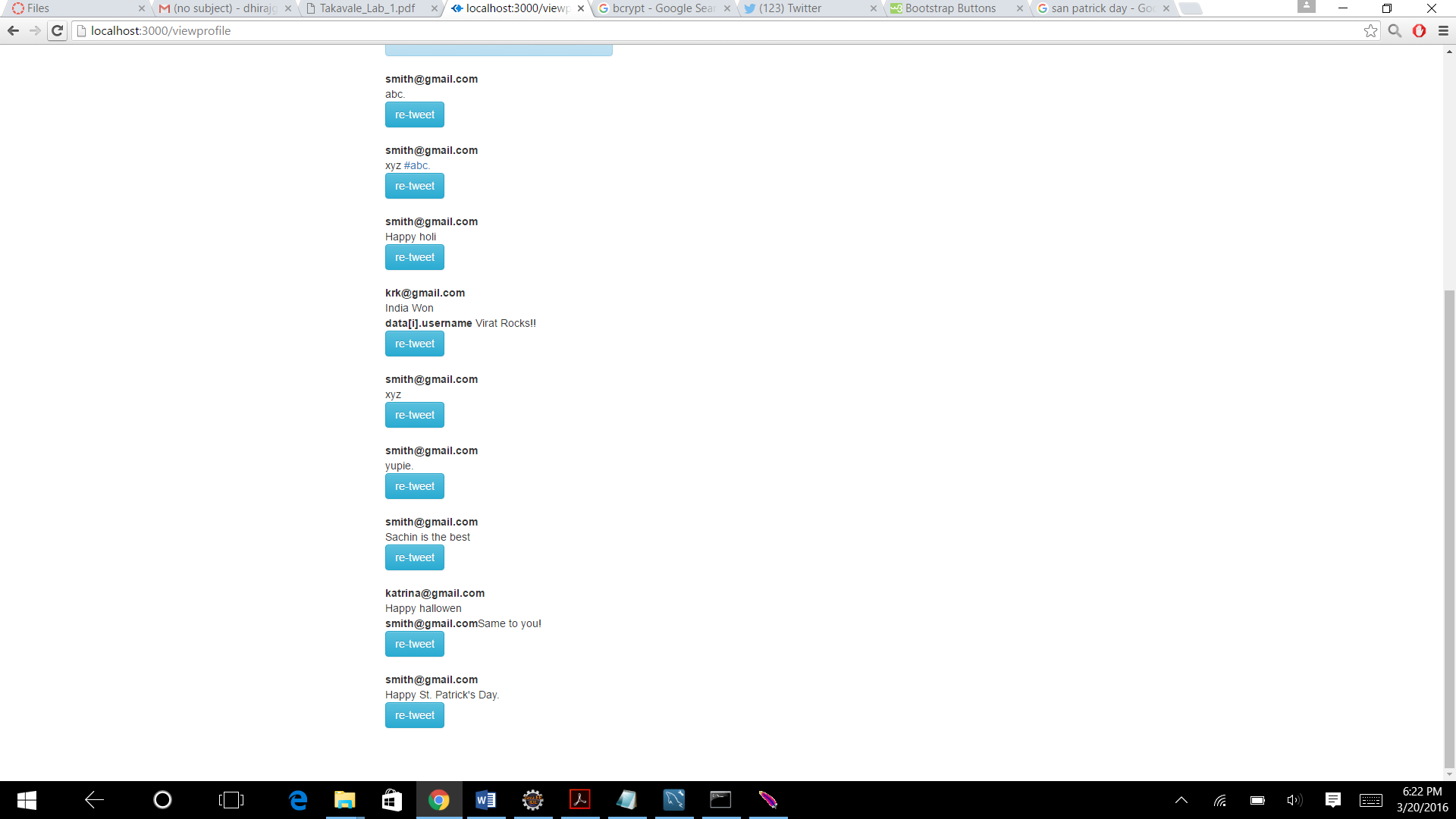


**Followers List**

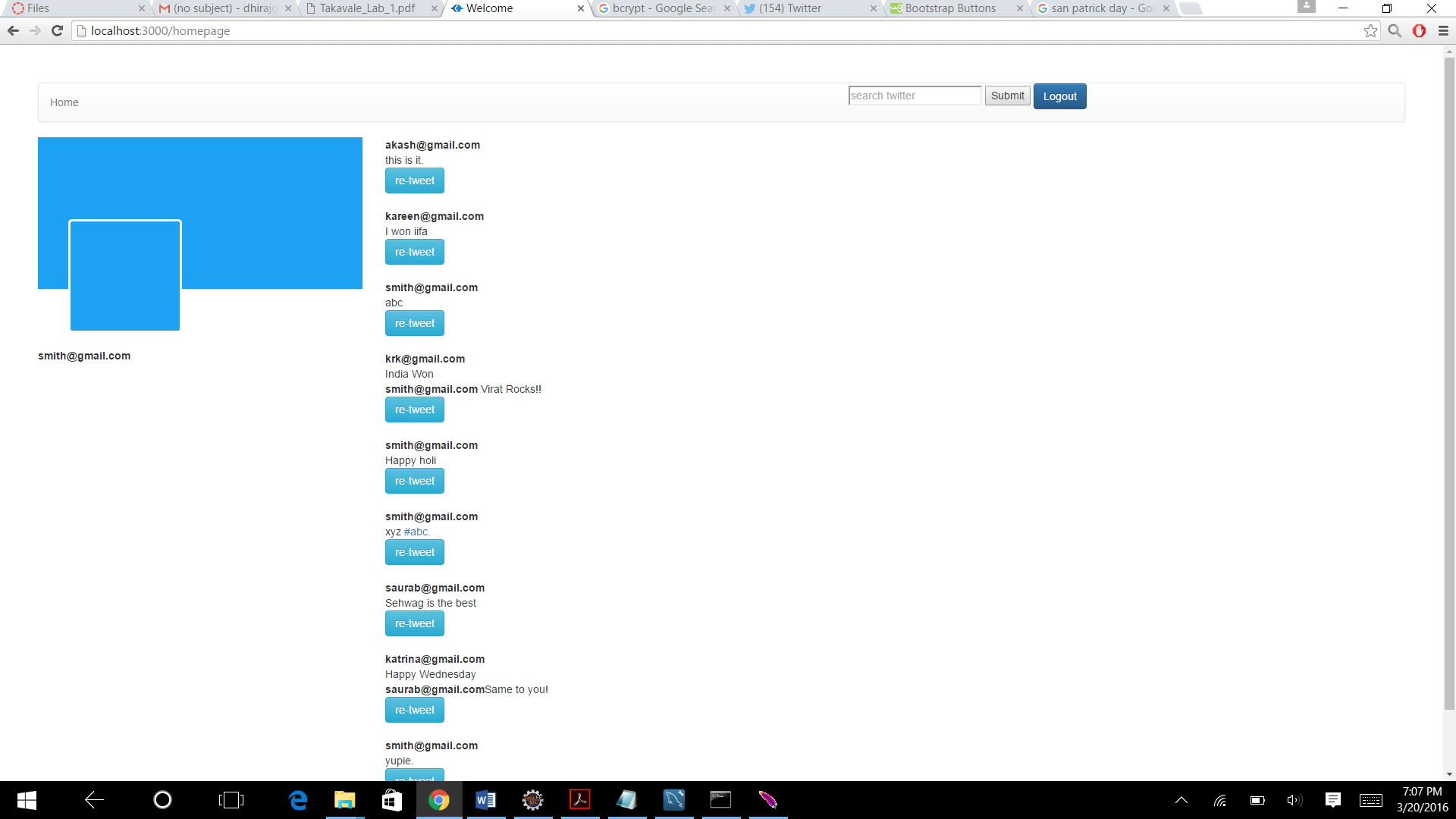


**Users Tweets and Retweets**

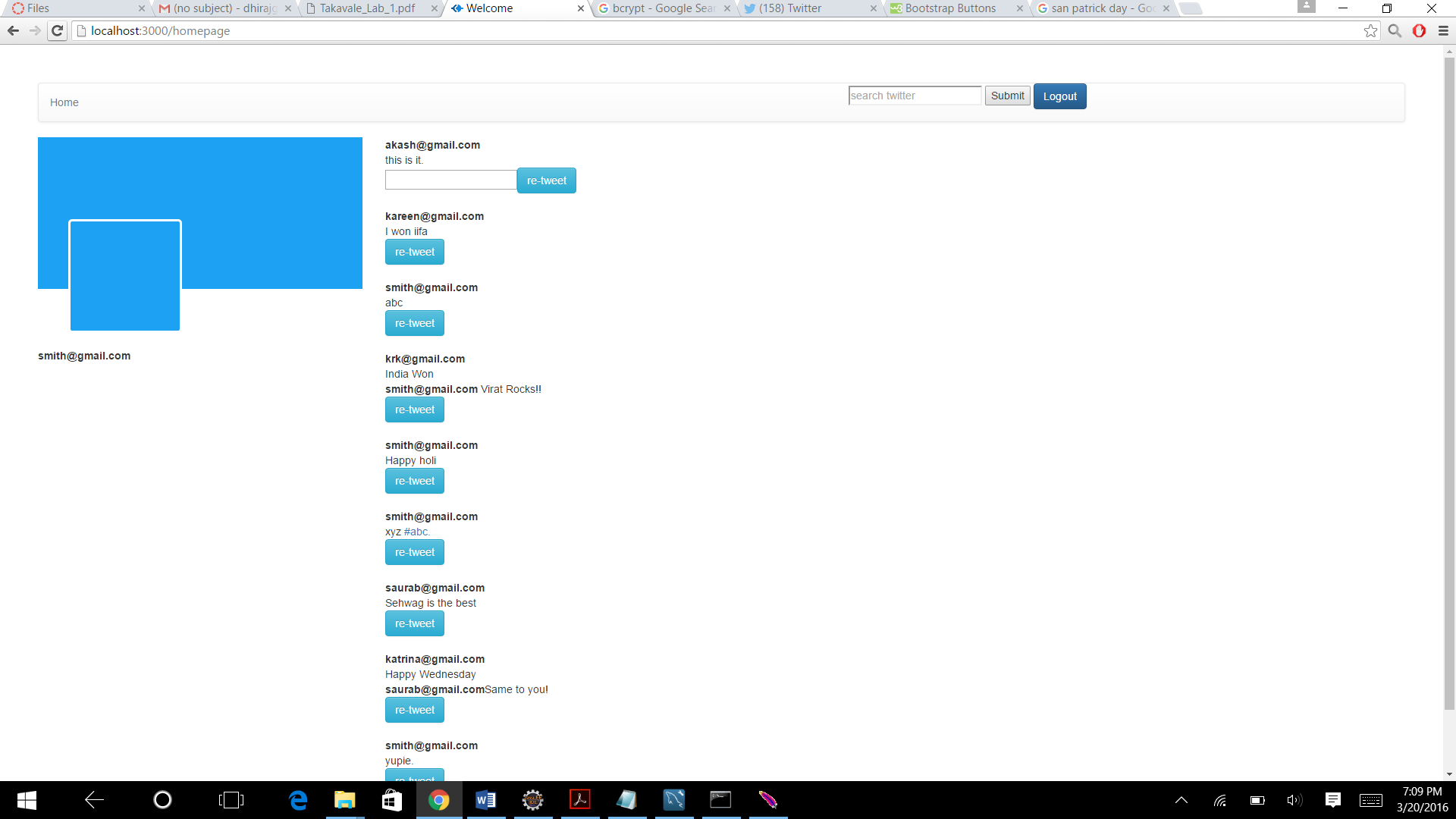


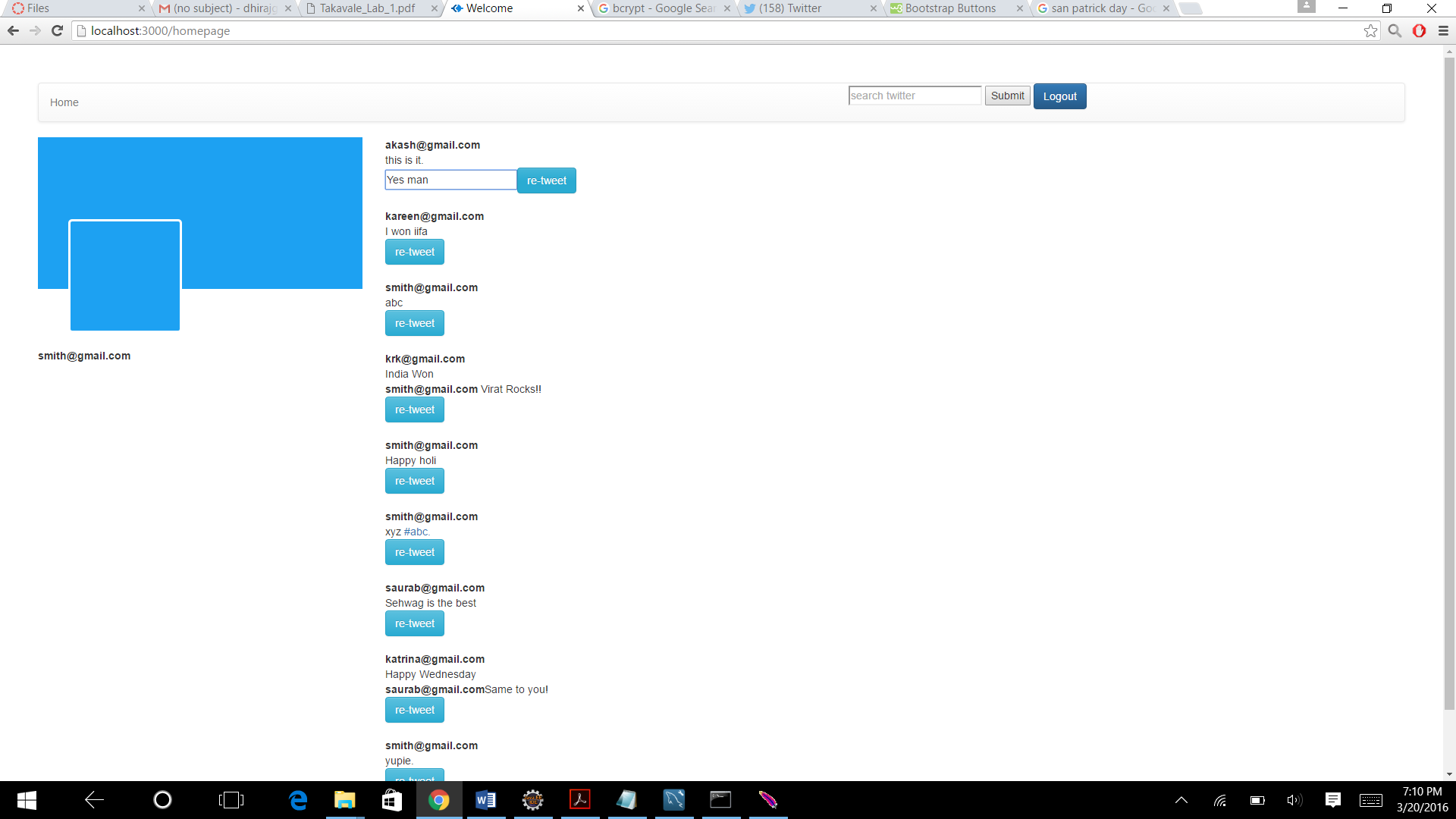


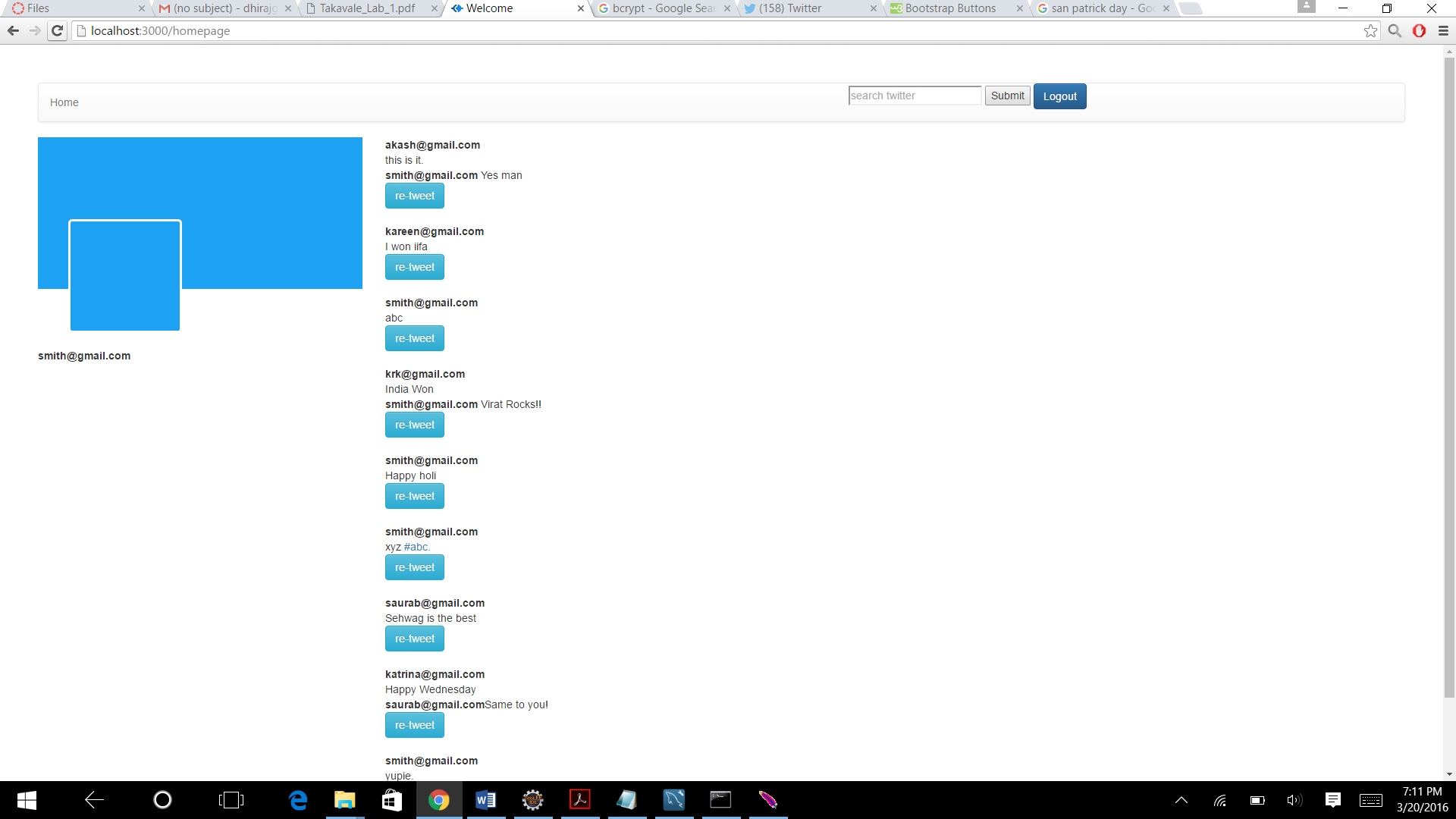
**Showing tweets of people you are following**



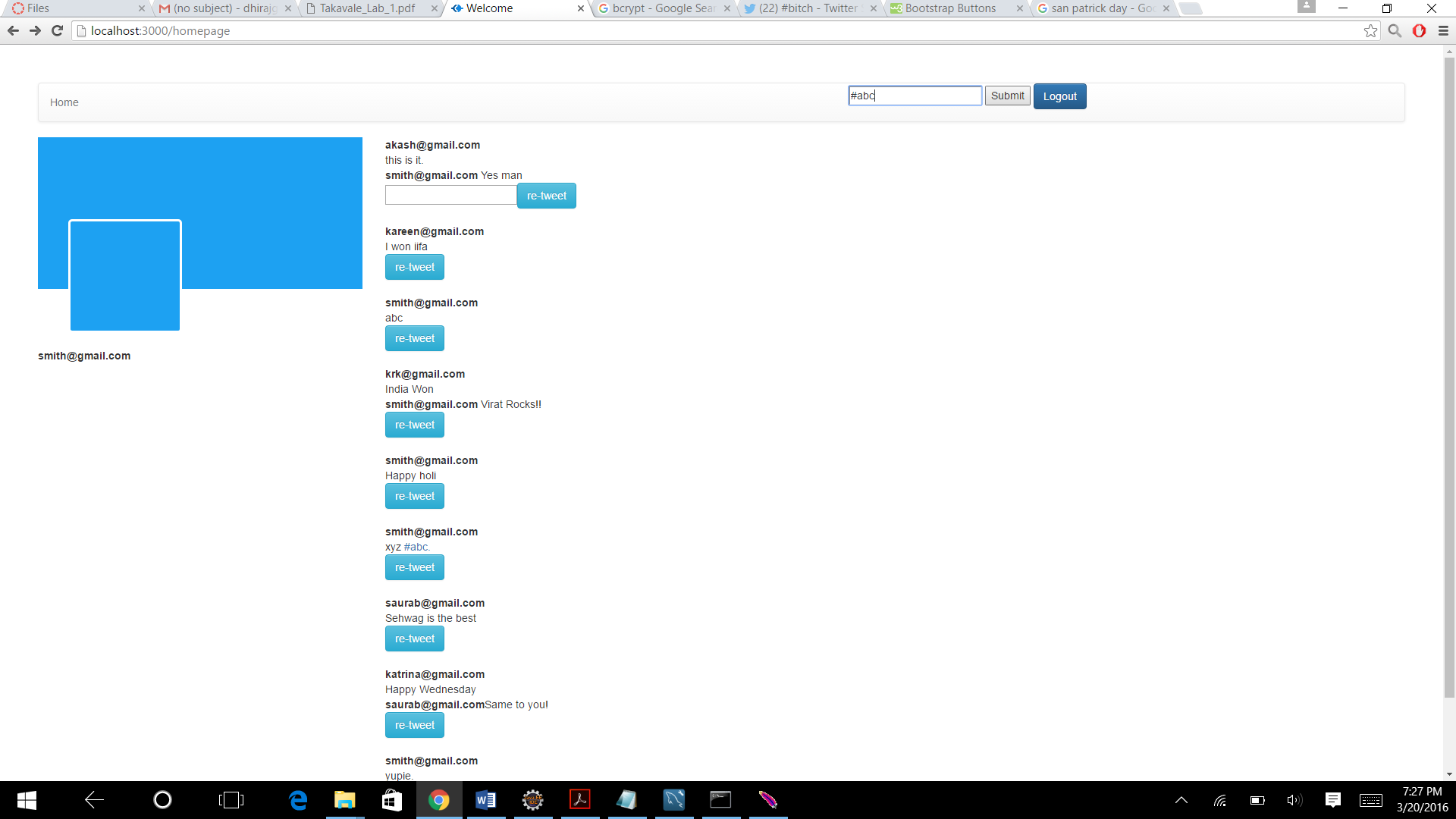
**Re tweet**

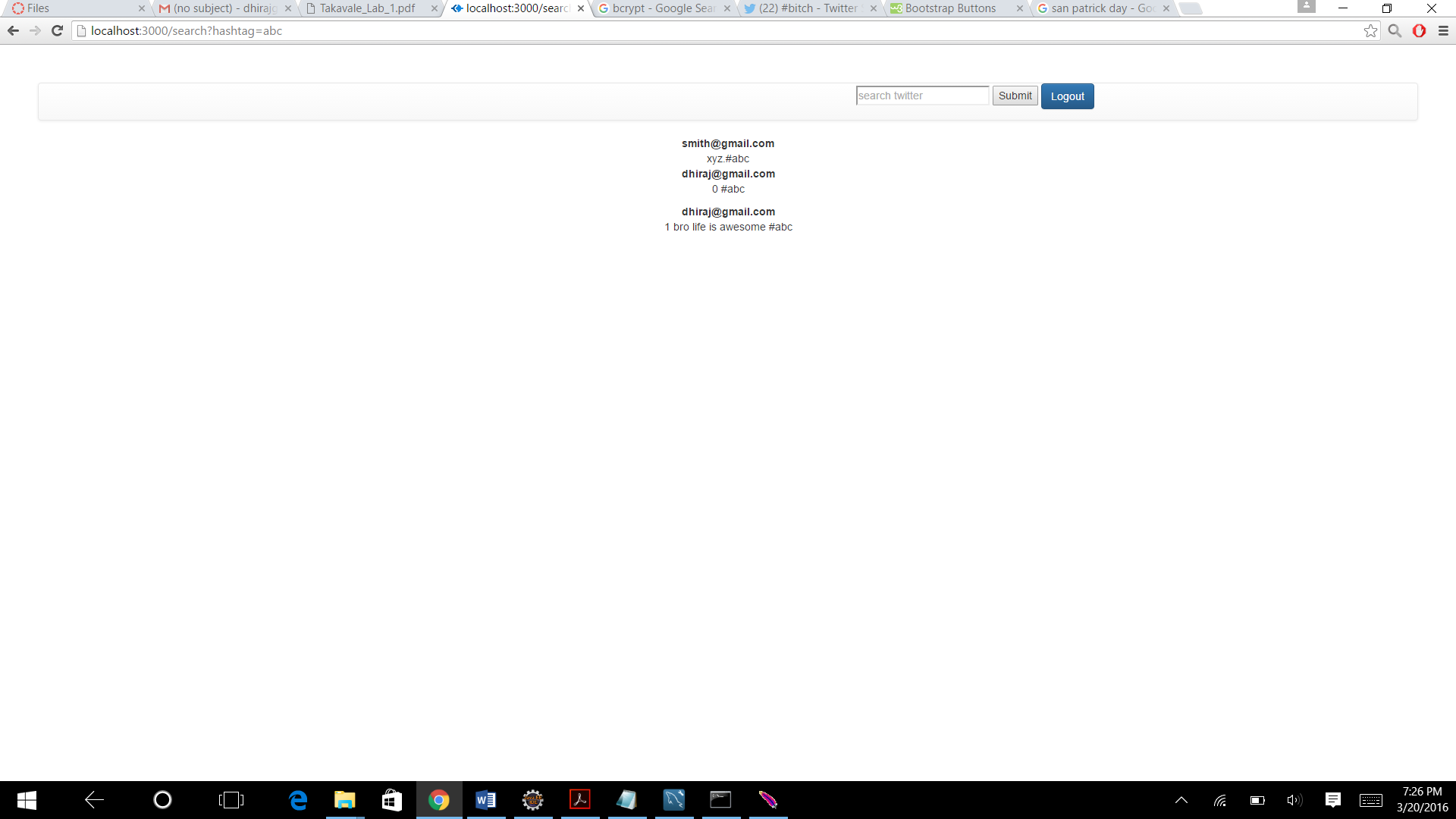




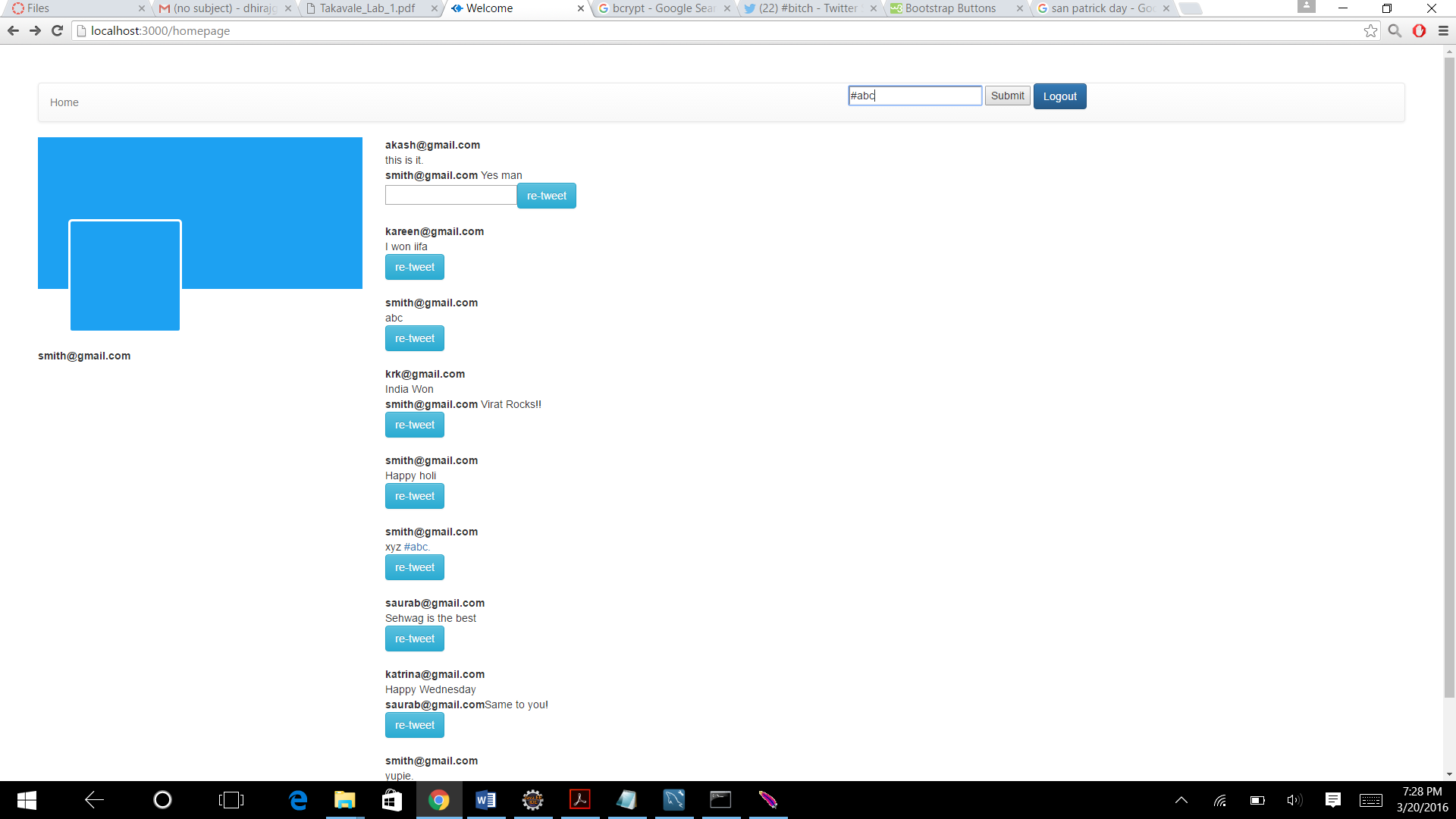


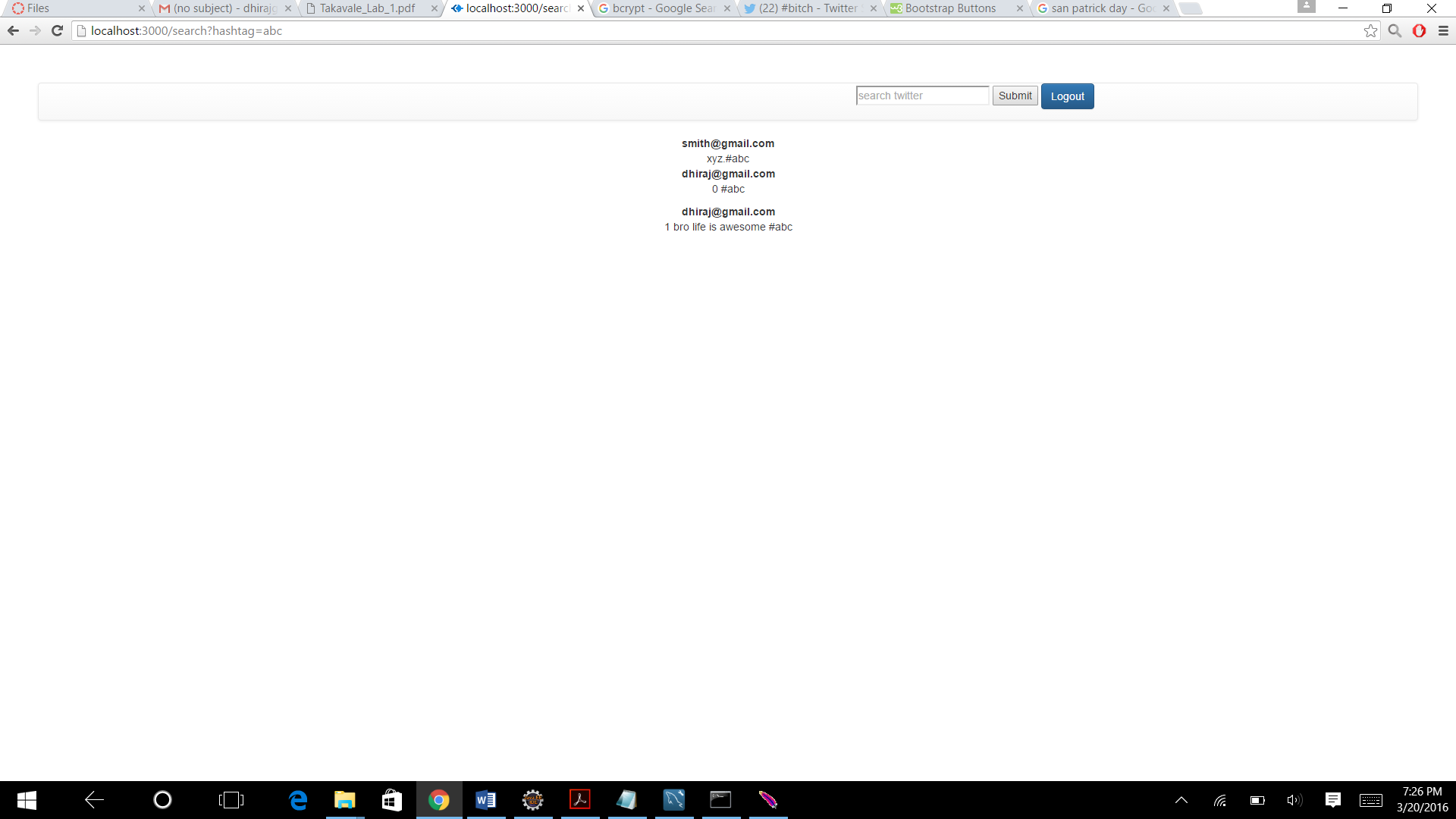
**#hashtag using search**



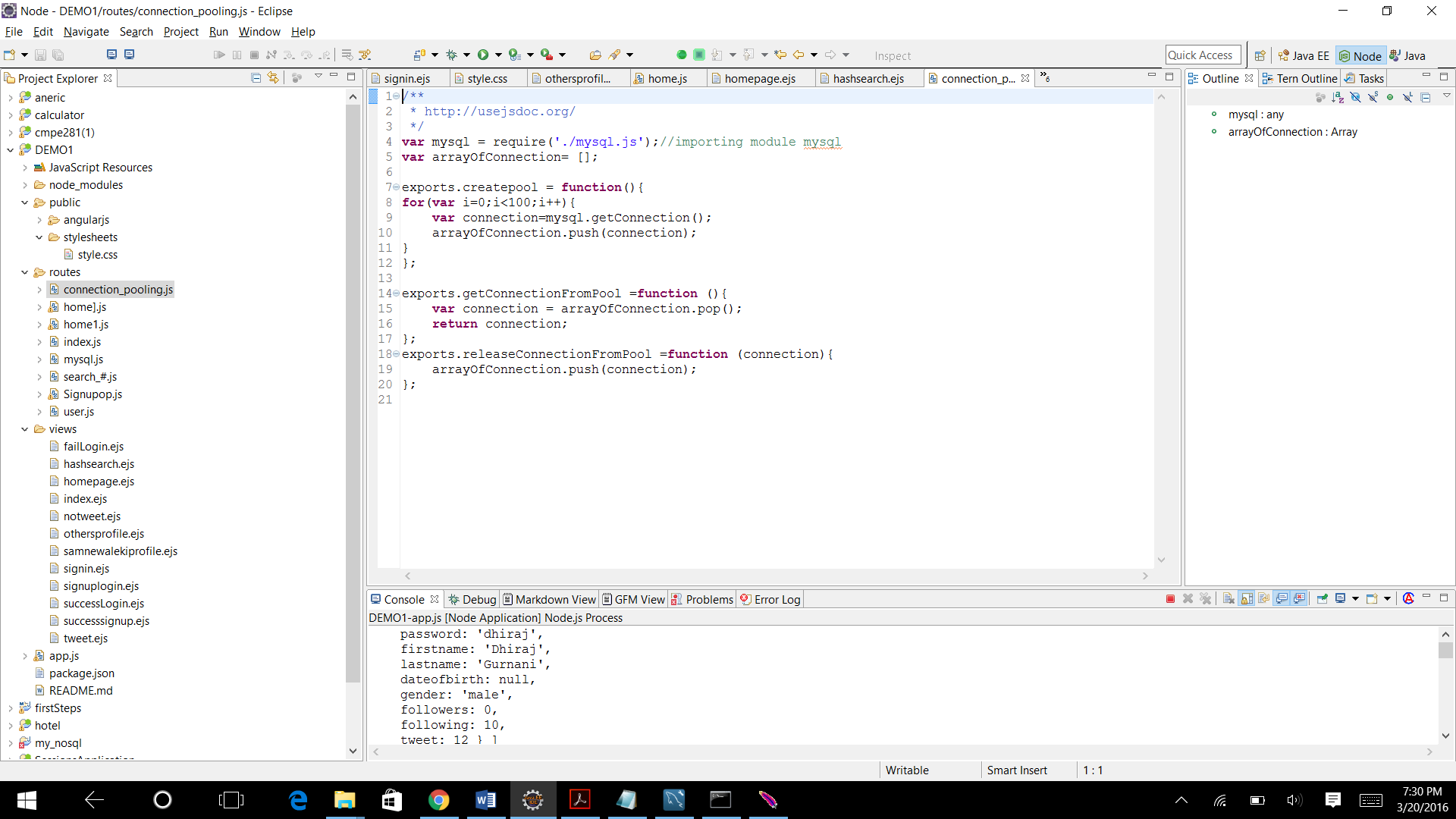


**#hashtag search in tweets**





**Connection pooling:**



**Jmeter**

**Explaination of the graph.**

As we can see the graph plotted for the User calls using the connection pooling and without connection pooling, there is change in the behaviour.

|  |  |  |
| --- | --- | --- |
|  | With connection pooling | Without connection pooling |
| 100 users | 10 | 12 |
| 200 users | 8 | 9 |
| 300 users | 11 | 12 |
| 400 users | 10 | 11 |
| 500 users | 8 | 10 |

There is approximately 13% difference in the between both the connection pooling and without connection pooling.

This difference is because of pools which are present which will save time for connection toget established and get destroyed. These are not present without connection pooling so it takes more time.

**3.Questions.**

**A1**

Using Bcrypt module for encryption in the twitter project.

It works on Blowfish cipher, which comes up with different passwords.

Bcrypt gives portability and cross platform support.

Bcrypt in gets the program gets the password and generates another password with its use after that bcrypt stores the password in the database and in the sign in function bcrypt gets the password from the user input and will compare the password with its own compare function and this will give the result and we will be able to sign in.It operates on64-bit of data,using a 56-bit. It is a ‘private key’ system.

HASH

It is used of a fixed length input. It is also known as a 'message digest', or a 'fingerprint'..

MD5

It is a 128 bit message digest function which is not useful for information more than it.

HMAC

HMAC is a hashing method that such as MD5 or SHA-1.

**A2.**

Here above is the chart of the data got by jmeter test with using connection pooling and without using connection pooling

Without connection pooling we have to create a connection every time and we have to remove it every time this takes a lot of time to establish a connection everytime and delete it everytime.

In connection pooling we have a pool established so we just have to use a connection from the pool so this will help us to save some time and our code will be implemented faster

Algorithm

1 We have first initialized a pool of array

2 There are 100 connections in the pool.

3 Now every time we need to connect to the database we just have to get the connection from the pool which is ready to use

4 after we are done with the usage we give the connection back to the pool.

**A3.**

Least Recently Used alogorithm has been used all the places recently used fields that are full while the time of search. All the frequently used data has been saved in the memory. Data is used, how much more data used and at what time. If the data is costly and which conforms when the usage is completed the cached is empitied of the expensive data. So, basically recently used data will be saved and one can get the often used data.

In request caching data is saved in the server with the help of key-value pair. Everytime a request is made the data is got from the server. Of all the caches available this is mostly used because cache is managed beyond the application by the browser. This makes the application to fetch data faster.

When we open a page and send a request the cookies would be check and if the last check was the same as this it will not send the whole file or else it will send so this will save the time and make the application run faster.