

Part 2: Ebay Marketplace Application:

INTRODUCTION:

I have implemented Ebay marketplace to demonstrate REST web services. I have tried to implement Ebay prototype and other functionalities very similar to ebay.

My application consists of following features.

1. Basic User functionalities:

- Sign up the new users (First name, Last name, Email, Password)
- Sign in with existing users
- Sign out

2. Profile:

- User will be able to edit their profile details which will include birthdate, address, user_id, email, name etc.
- All user will be able to see profile of any other user, Which I have implemented using ebay handle.
- Homepage will consist of advertisements and navigation bar to navigate through complete website.
- User will be able to take part in bidding which will last for four days after that user with highest bid will get the item and all details of transaction will be reflected in their accounts.
- User's cart will be maintained in database. User will be able to get cart details when he log in to the account.
- User will be able to see last logged in time and their bought and sold items in profile section.

3. other features:

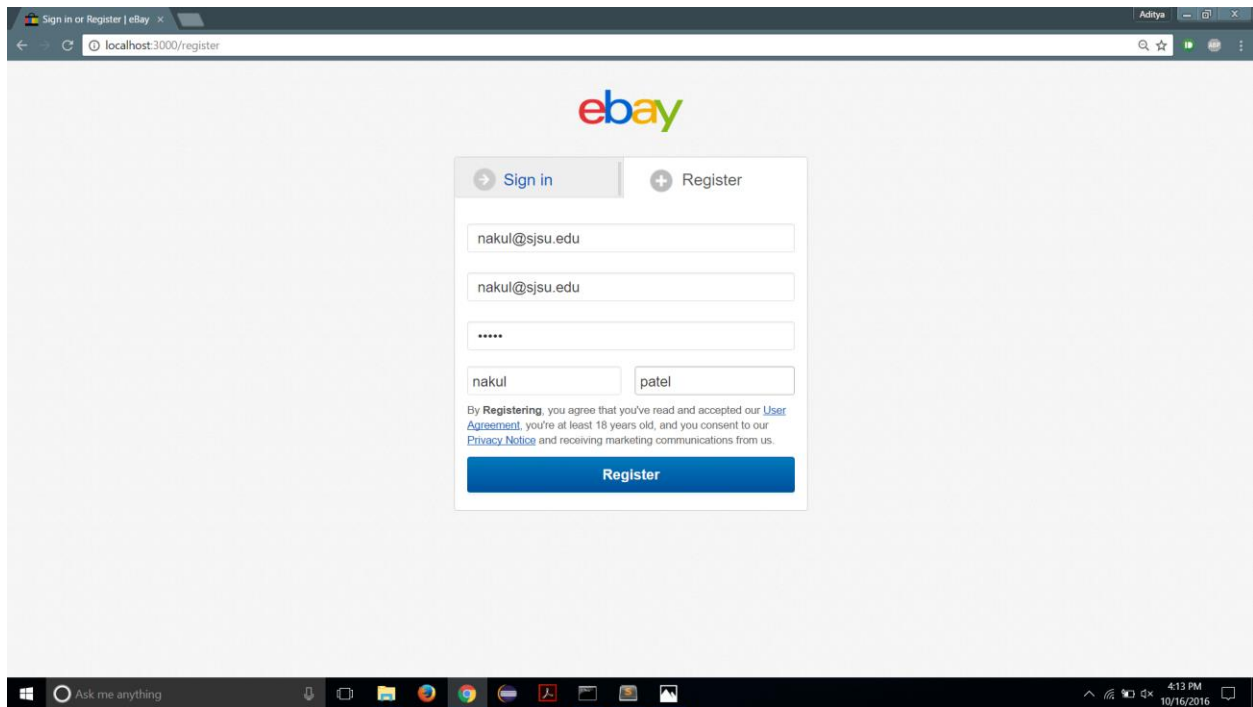
- Server will perform card validations at payment page.
- Server will maintain last logged in time of every user. In addition to that server will maintain event logs and bidding logs for all users.

System design:

I have used HTML5, CSS, Bootstrap, Javascript, Angular js, Node.js and Express.js to build this application.

- I have used Node.js and Express js in backend whereas Angular.js, HTML5, CSS are used in front end.
- There are 13 pages in my application.
- Homepage, sign in page, register page, Listing pages, Item pages for auction items and buy now items, profile pages, cart page and checkout page.
- For the storage purposes I have used session storage and MYSQL database.
- When user tries to sign up server checks whether user exists in database or not, same validation will be applied for sign up page.
- Connection pool is created using collection data structure whenever user requests for resource user will be assigned connection from pool.
- Bidding is implemented using cron npm package which schedules task in time intervals

Sign up page:



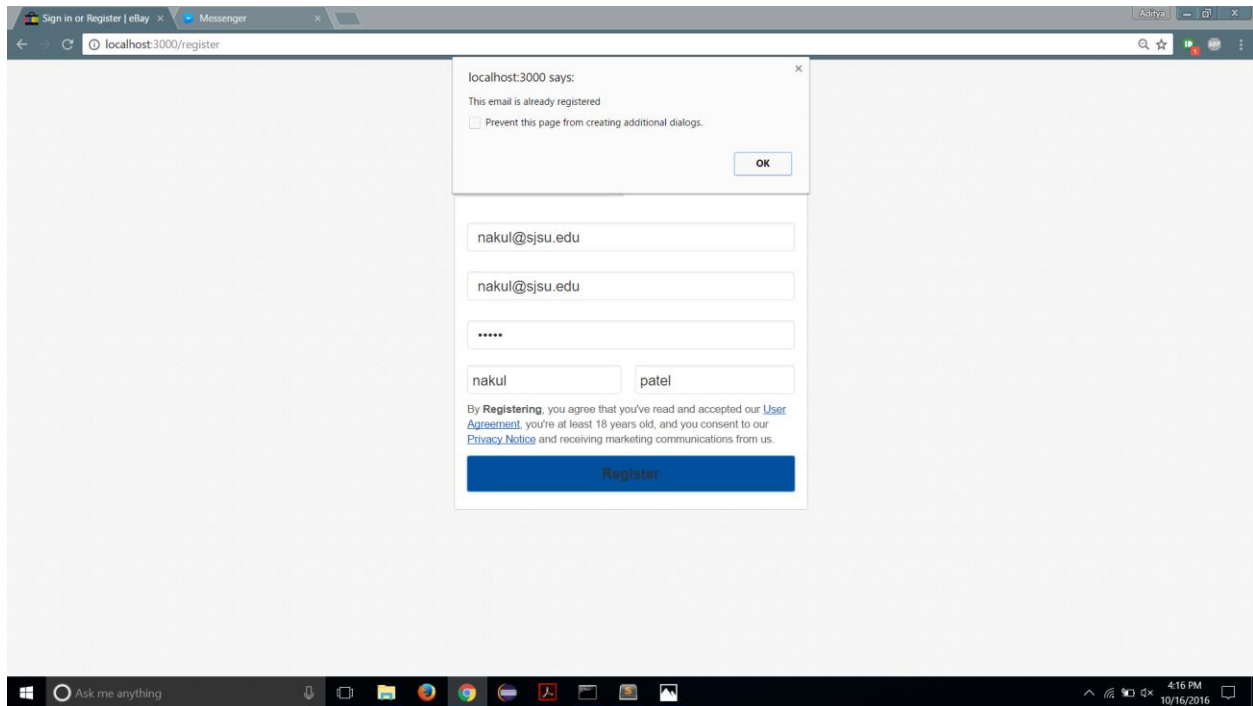
The screenshot shows a web browser window with the address bar displaying "localhost:3000/register". The page features the eBay logo at the top center. Below the logo is a registration form with two tabs: "Sign in" and "Register". The "Register" tab is active. The form contains the following fields:

- Email: "nakul@sjsu.edu"
- Confirm Email: "nakul@sjsu.edu"
- Password: "*****"
- First Name: "nakul"
- Last Name: "patel"

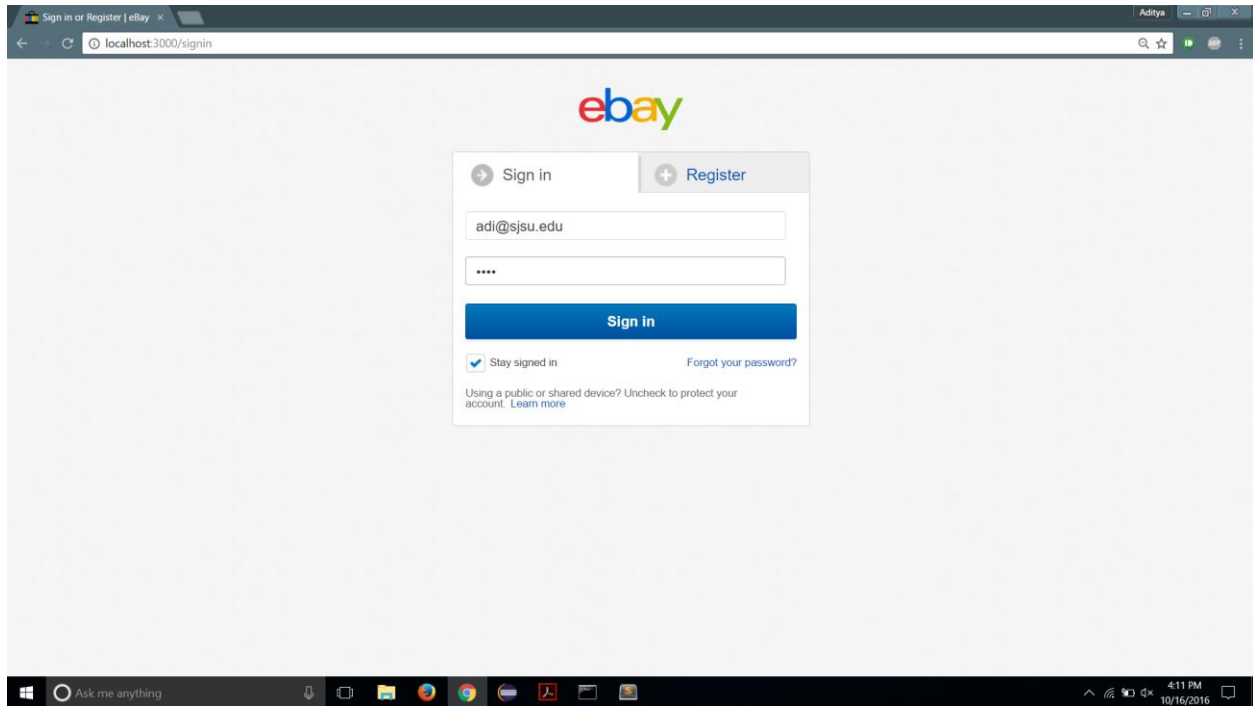
Below the fields, there is a line of text: "By **Registering**, you agree that you've read and accepted our [User Agreement](#), you're at least 18 years old, and you consent to our [Privacy Notice](#) and receiving marketing communications from us." At the bottom of the form is a blue button labeled "Register".

The Windows taskbar at the bottom shows the system clock as 4:13 PM on 10/16/2016.

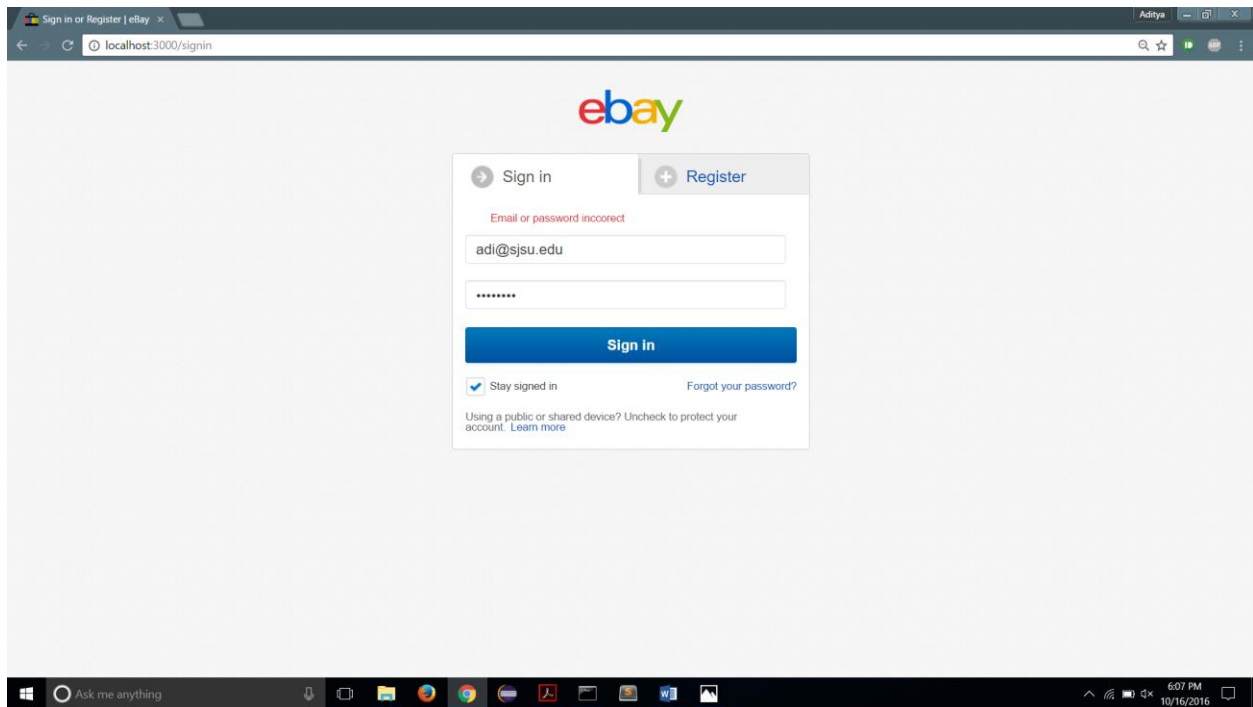
Sign up page with validations:



Sign in page:

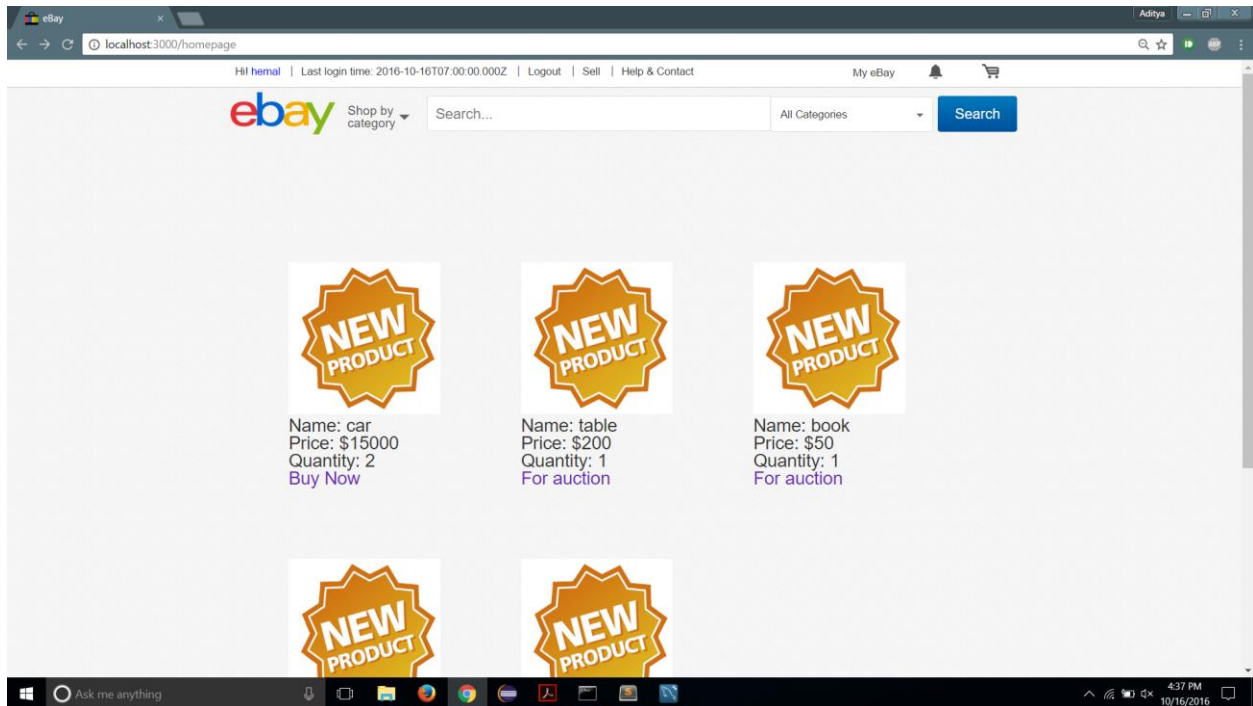


Sign in page with validations:

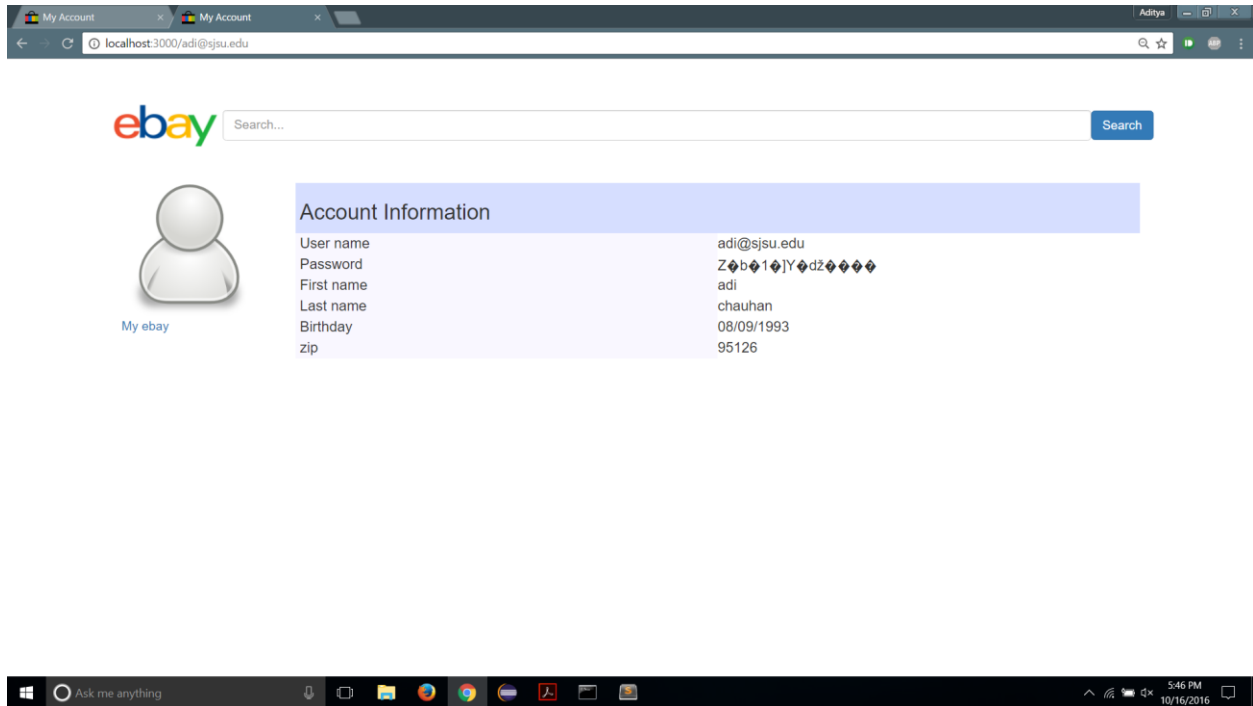


Homepage :

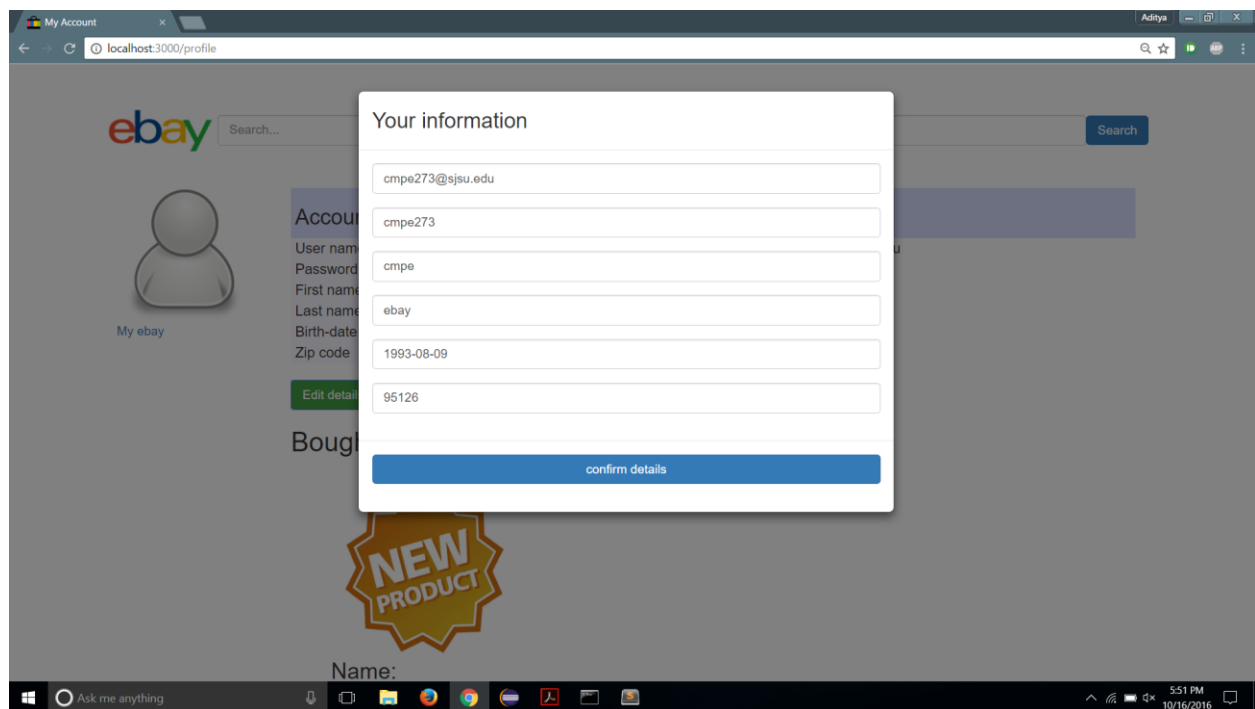
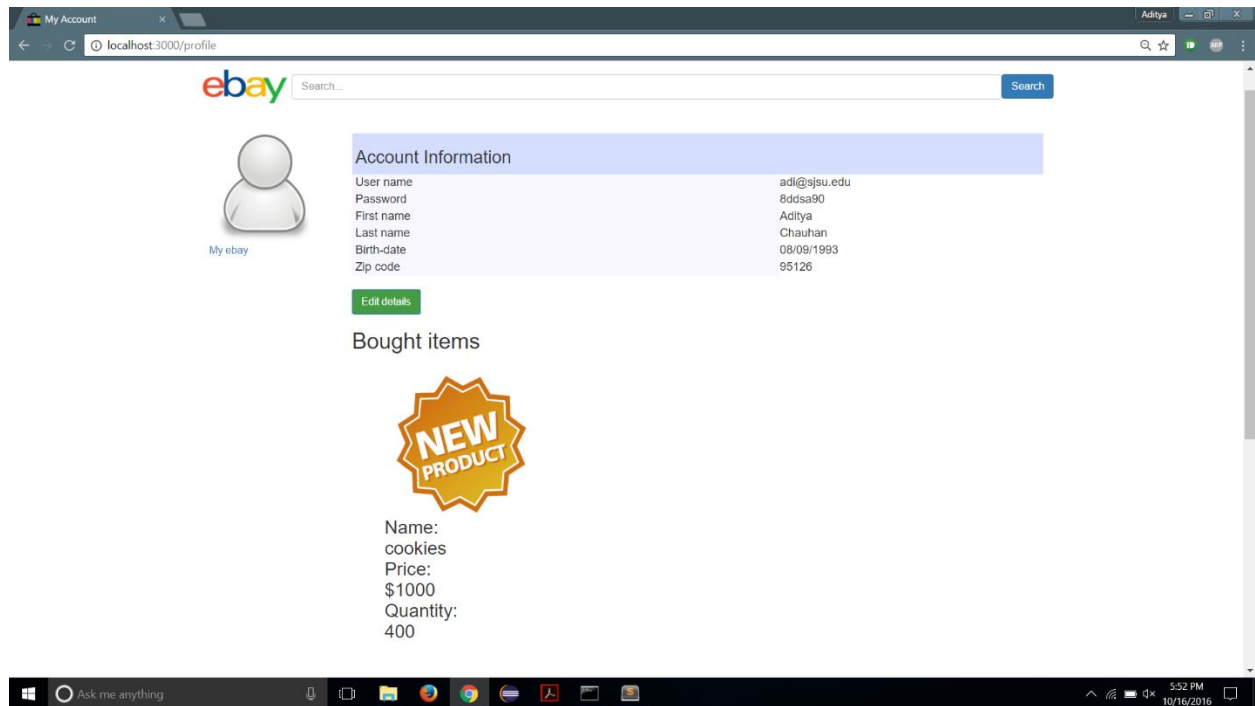
Contains navigation bar and products displayed for auction and buy now.



Ebay handle:



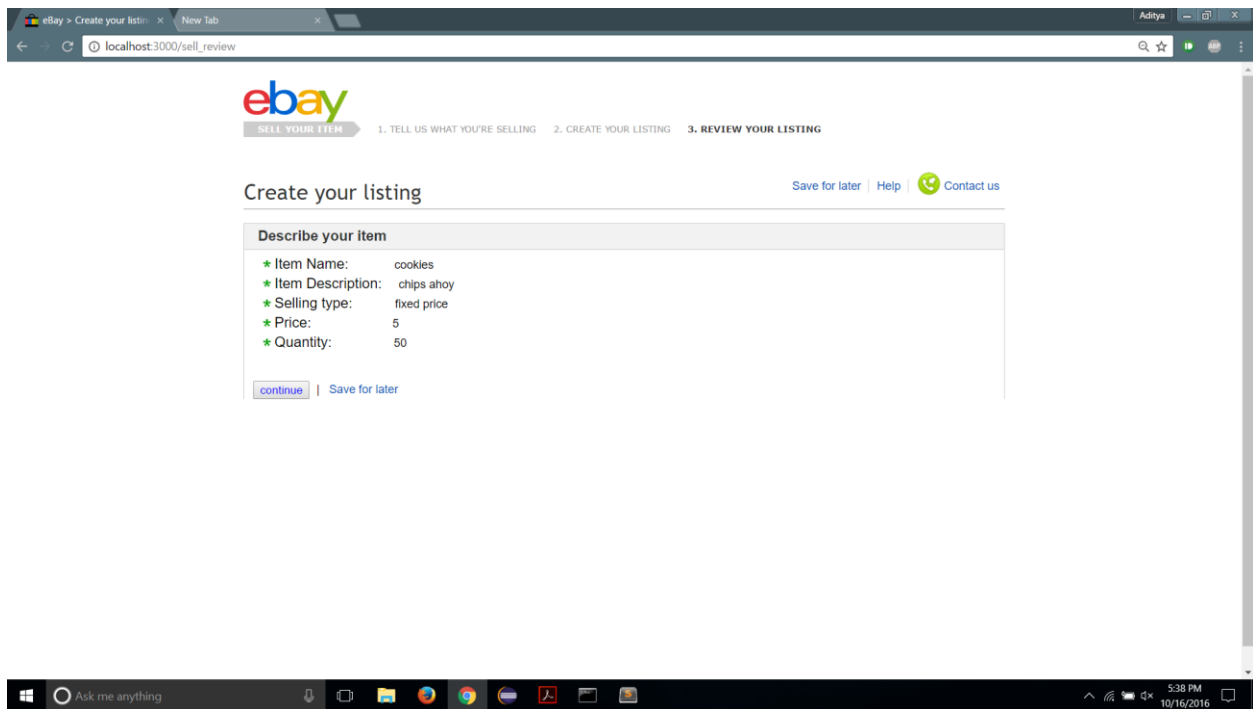
Users who is logged in will be able to edit their details:



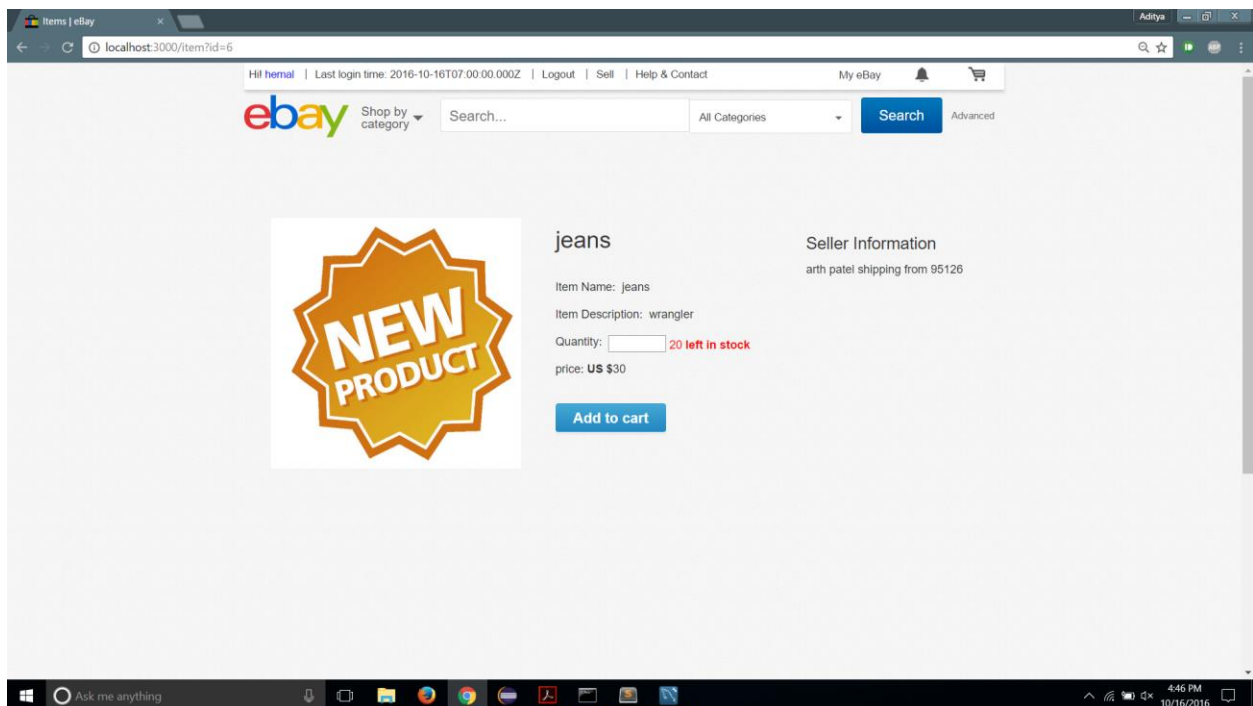
Post listings:

The screenshot shows a web browser window with the URL `localhost:3000/sell`. The page features the eBay logo and a progress bar with three steps: 1. TELL US WHAT YOU'RE SELLING (active), 2. CREATE YOUR LISTING, and 3. REVIEW YOUR LISTING. Below the progress bar, the heading 'Tell us what you're selling' is followed by 'Help' and 'Contact us' links. A 'Start a new listing' section contains a text input field with the placeholder 'Give us a title for your listing (mobile phone)' and a 'Get started' button. A hint below the field reads 'For example: Mobile, Car, Cookies, Camera'.

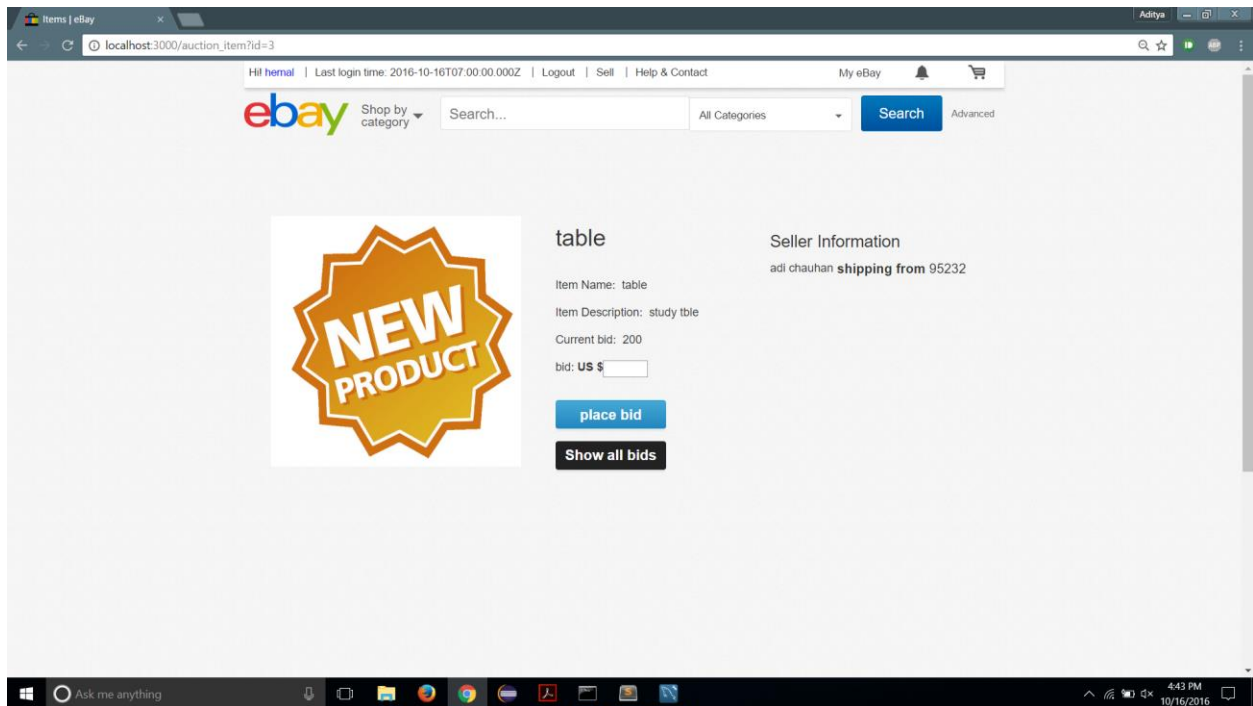
The screenshot shows a web browser window with the URL `localhost:3000/sell_info?keywords=mobile+phone`. The page features the eBay logo and a progress bar with three steps: 1. TELL US WHAT YOU'RE SELLING, 2. CREATE YOUR LISTING (active), and 3. REVIEW YOUR LISTING. Below the progress bar, the heading 'Create your listing' is followed by 'Save for later', 'Help', and 'Contact us' links. A 'Describe your item' section contains several form fields: 'Item Name' with the value 'cookies', 'Item Description' with the value 'chips ahoy', 'Choose a Format and Price' with 'Fixed price' selected, 'Price' with the value '5', and 'Quantity' with the value '50'. A legend indicates that an asterisk (*) denotes a required field. At the bottom of the form, there are 'enter' and 'Save for later' buttons.



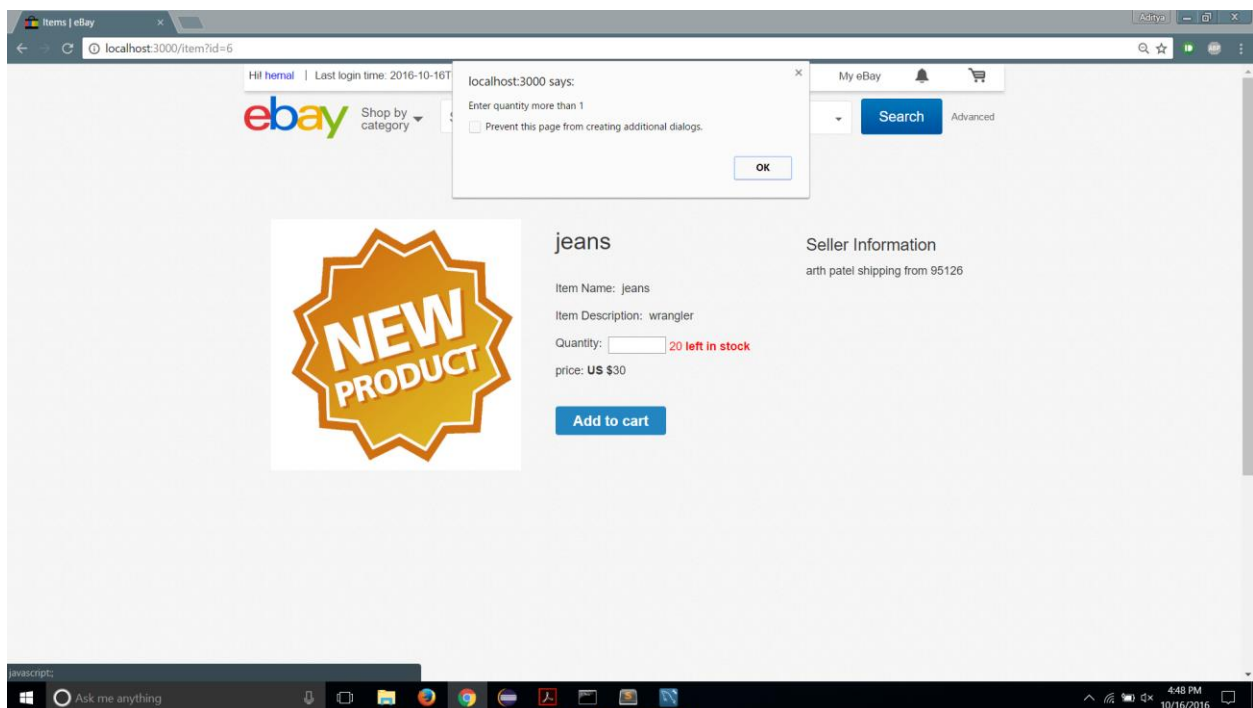
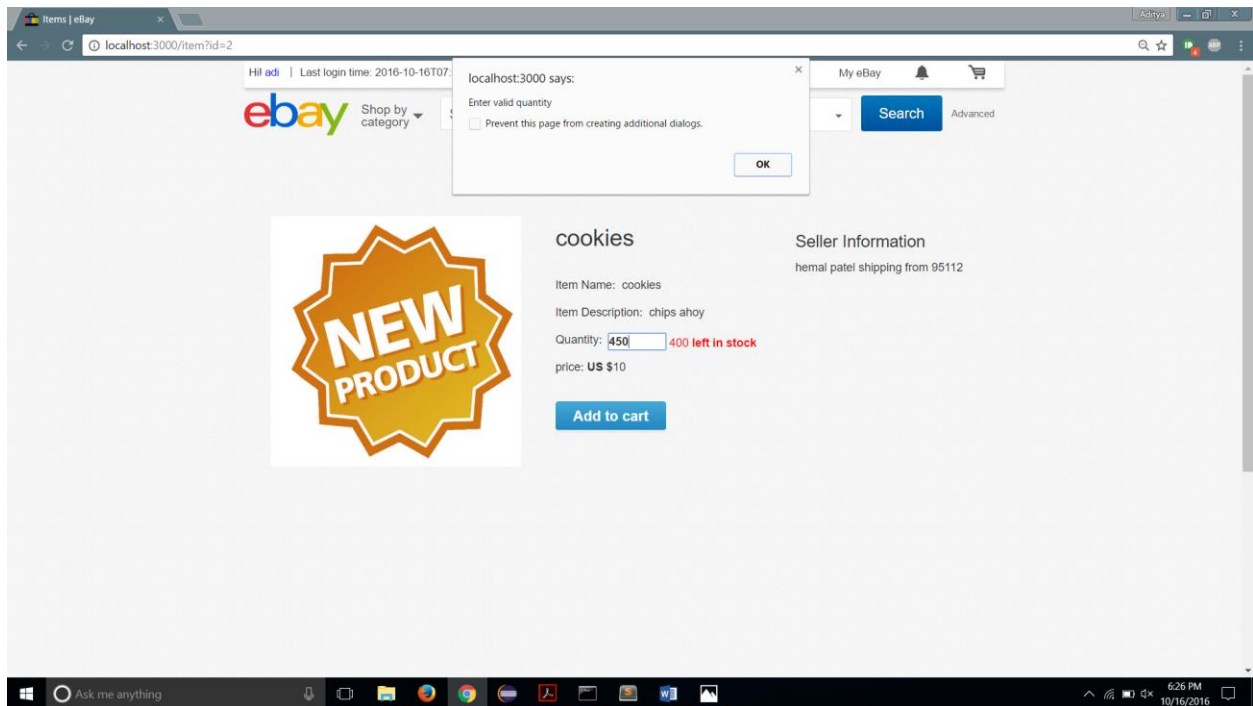
Buy now products:



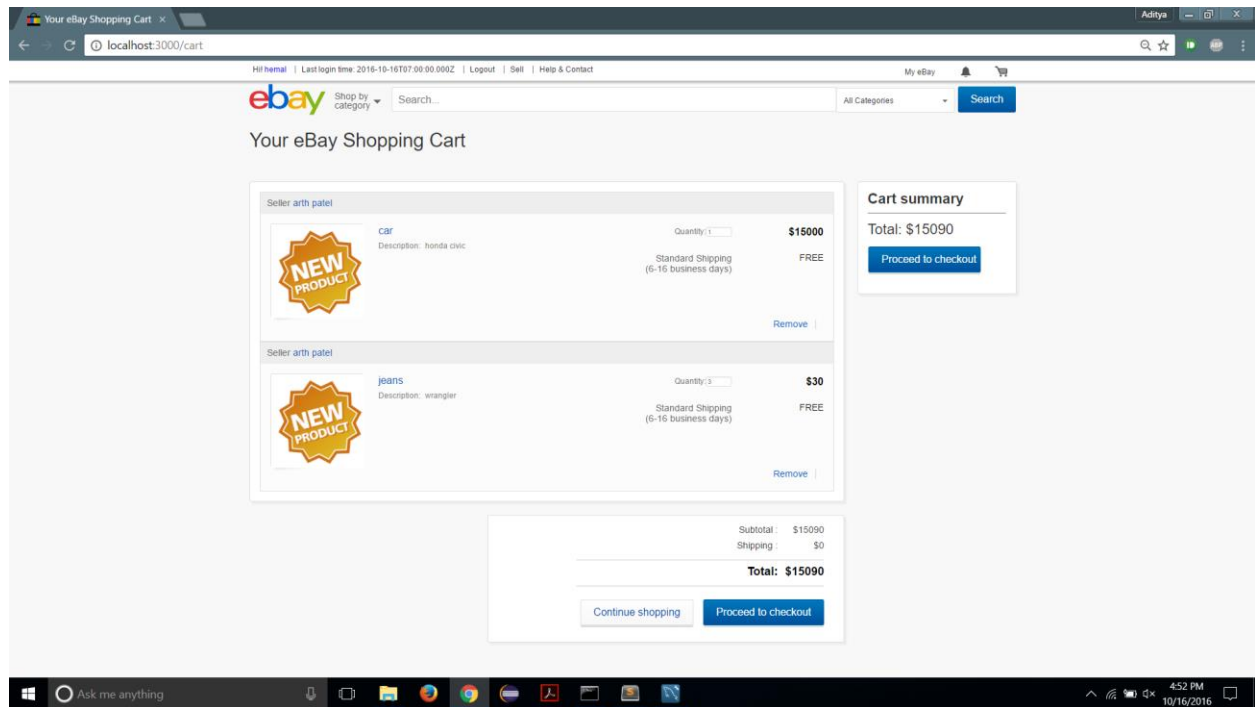
Products for auction:



Products with validations:



Cart page:



Checkout page:

The screenshot shows the eBay Checkout page in a web browser. The page is titled "eBay Checkout" and has a URL of "localhost:3000/checkout". It is divided into several sections:

- Pay with:** A section for entering card details. It includes a radio button for "Enter card details", a card number field (5645879856234512), an expiration date field (11/2018), a CVV field (***), and a "Submit" button.
- Items:** A summary of the items being purchased, showing "Items" for \$15090 and "Shipping" for Free.
- Order total:** A box showing the "Order total" as \$15090, with a "Confirm and pay" button.
- Ship to:** A section for the shipping address, showing "Aditya Chauhan, Krishna-2, Ahmedabad India, 380005" and a "Change" link.
- Review items and shipping:** A section for reviewing the items and shipping. It shows the seller "arth patel" and a table of items:

Item	Description	Price	Quantity	Shipping
car	honda civic	\$15000	1	Standard Shipping
Total:		\$15090		

The bottom of the page shows a Windows taskbar with the "Ask me anything" search bar and various application icons. The system clock indicates 5:17 PM on 10/16/2016.

Checkout page with card validations:

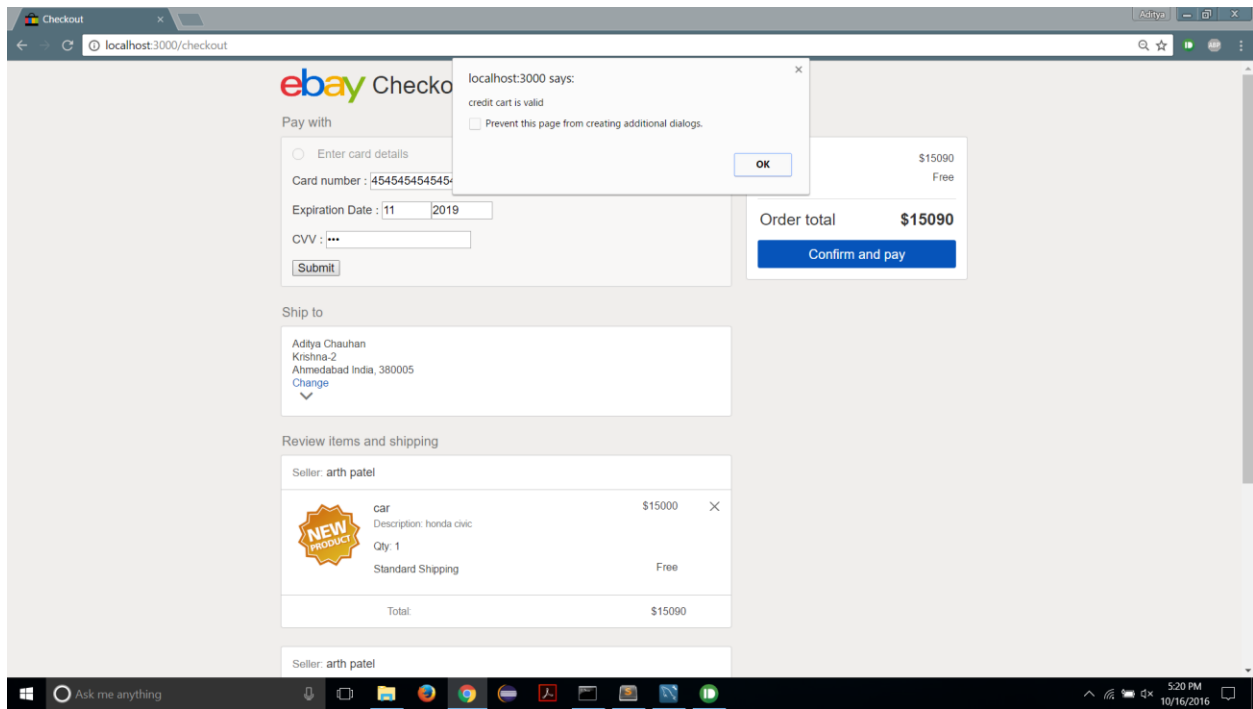
This screenshot shows the same eBay Checkout page as the previous one, but with a modal dialog box open for card validation. The dialog box is titled "localhost:3000 says:" and contains the text "Enter valid card details" and a checkbox for "Prevent this page from creating additional dialogs." with an "OK" button.

The background page shows the same "Pay with" section, but the card number field now contains "89898989898". The "Expiration Date" field is set to "15/2015". The "CVV" field is still "***". The "Submit" button is visible.

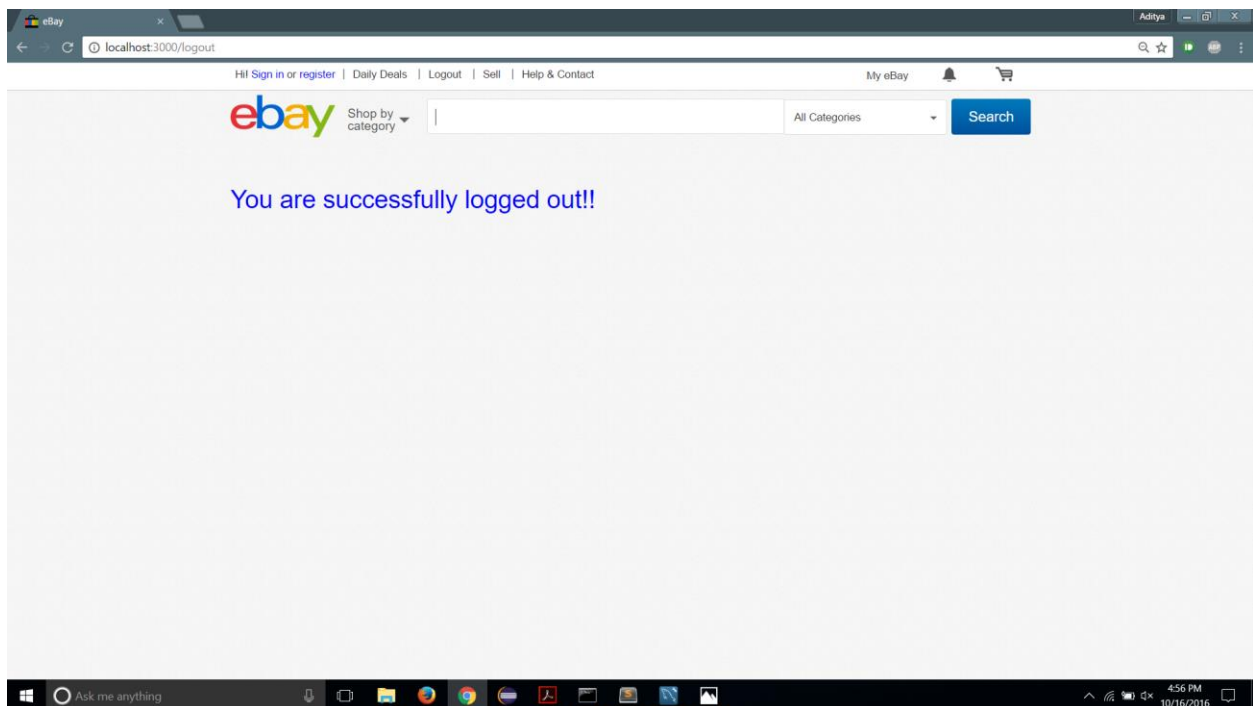
The "Items" and "Order total" sections remain the same, showing \$15090 for items and free shipping, with a total of \$15090.

The "Ship to" and "Review items and shipping" sections are also visible, showing the same shipping address and item details as the previous screenshot.

The bottom of the page shows the same Windows taskbar with the "Ask me anything" search bar and various application icons. The system clock indicates 5:19 PM on 10/16/2016.



Logout message:



MOCHA tests for REST web service API calls

```
C:\Users\sunny\ebayk\test>mocha mocha_test_aditya
```

```
http tests
```

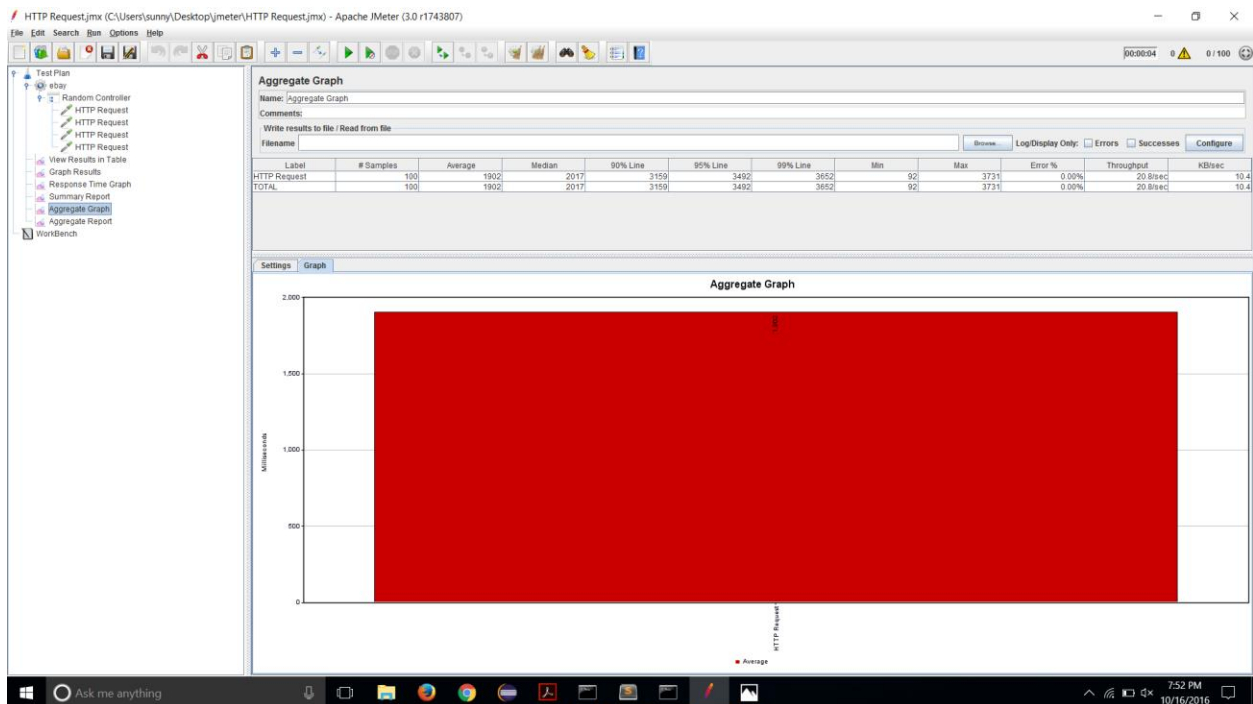
- ✓ should return signin page for correct url
- ✓ should not return page for wrong url
- ✓ should return register page for correct url
- ✓ should login for correct credentials
- ✓ should not allow to login

```
5 passing (78ms)
```

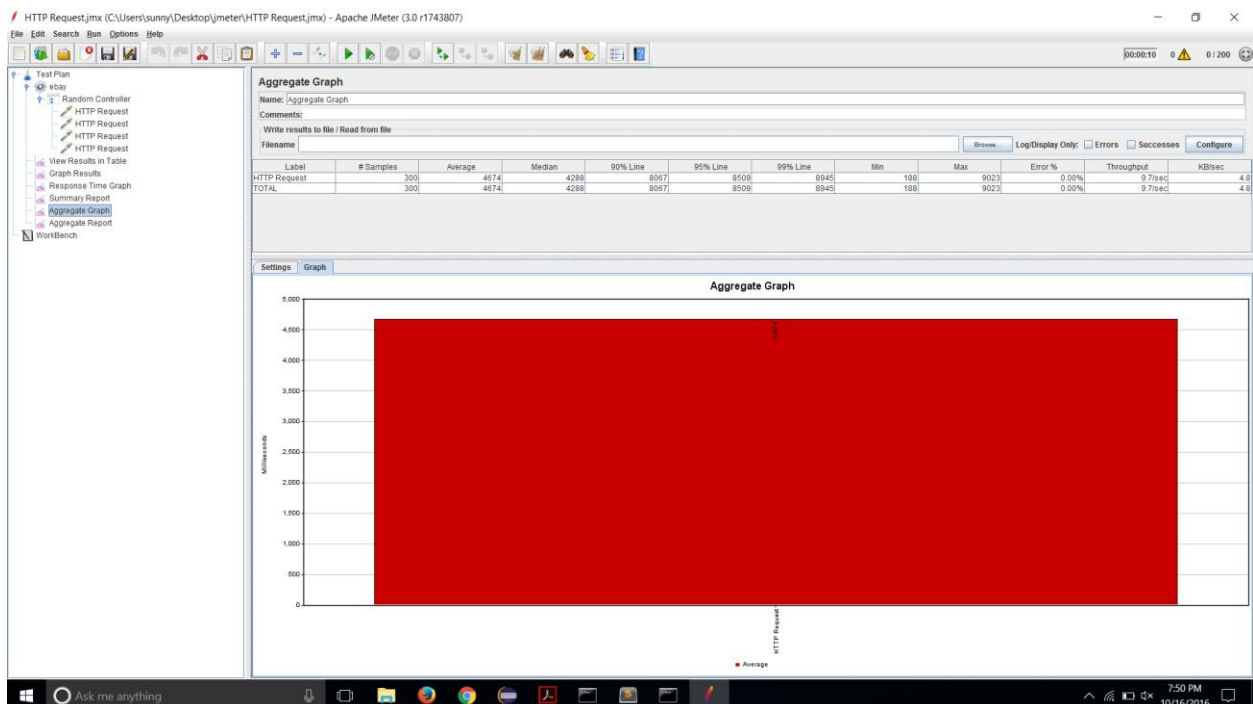
```
C:\Users\sunny\ebayk\test>
```

JMeter test with connection pooling:

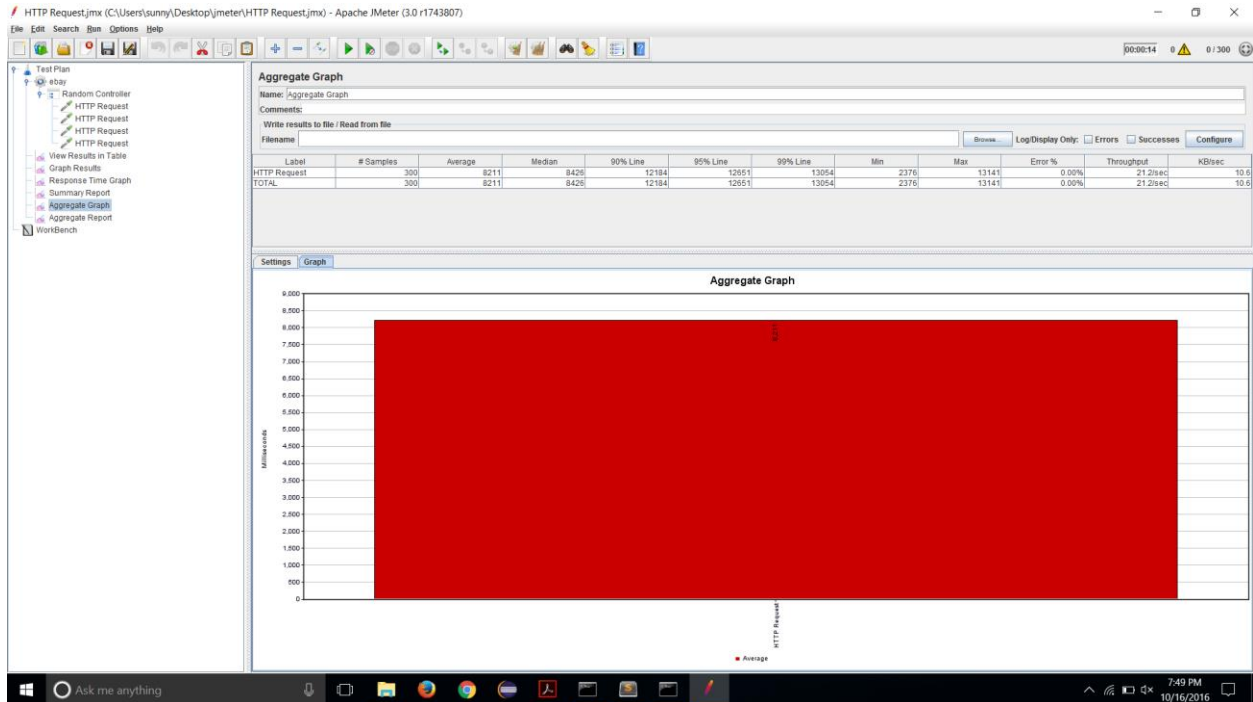
For 100 concurrent users:



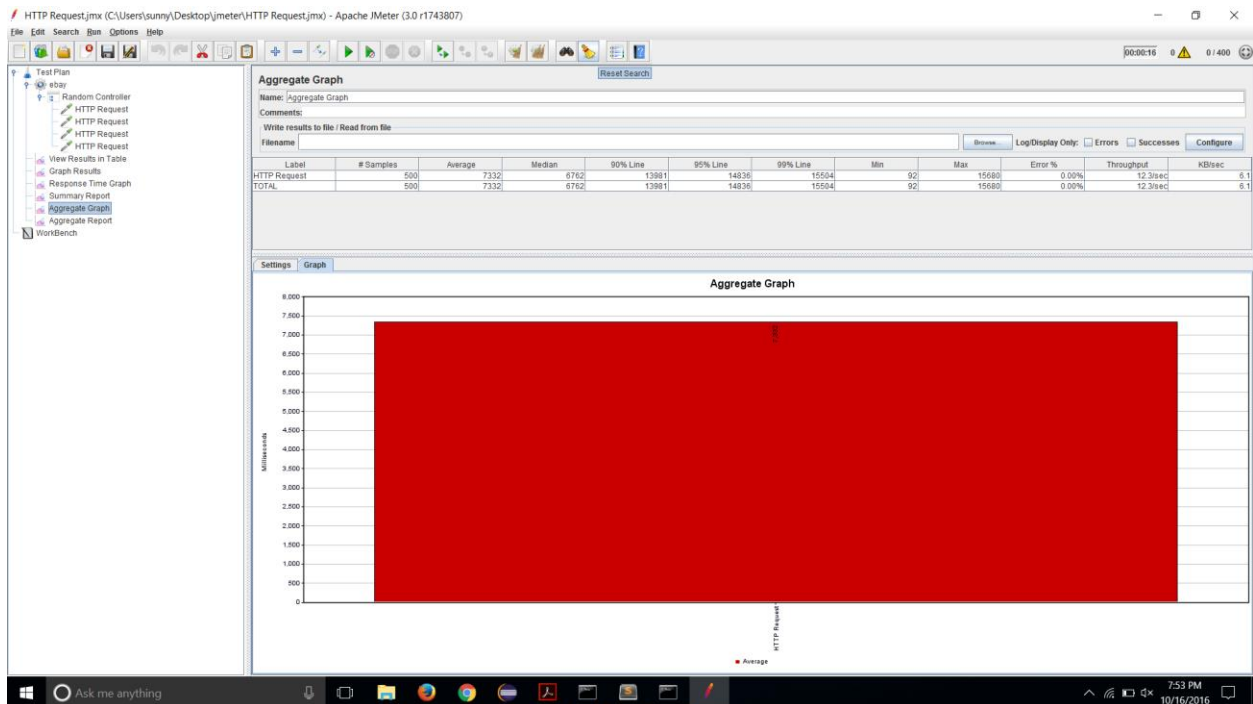
For 200 concurrent users:



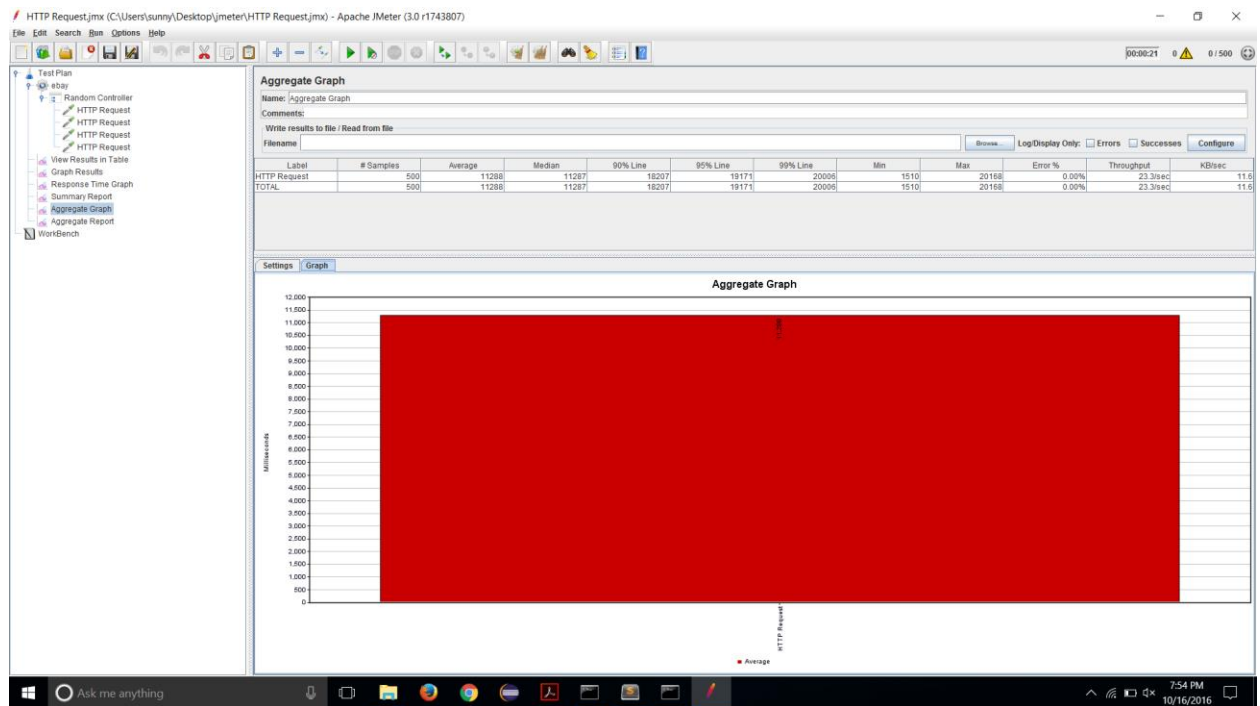
For 300 concurrent users:



For 400 concurrent users:

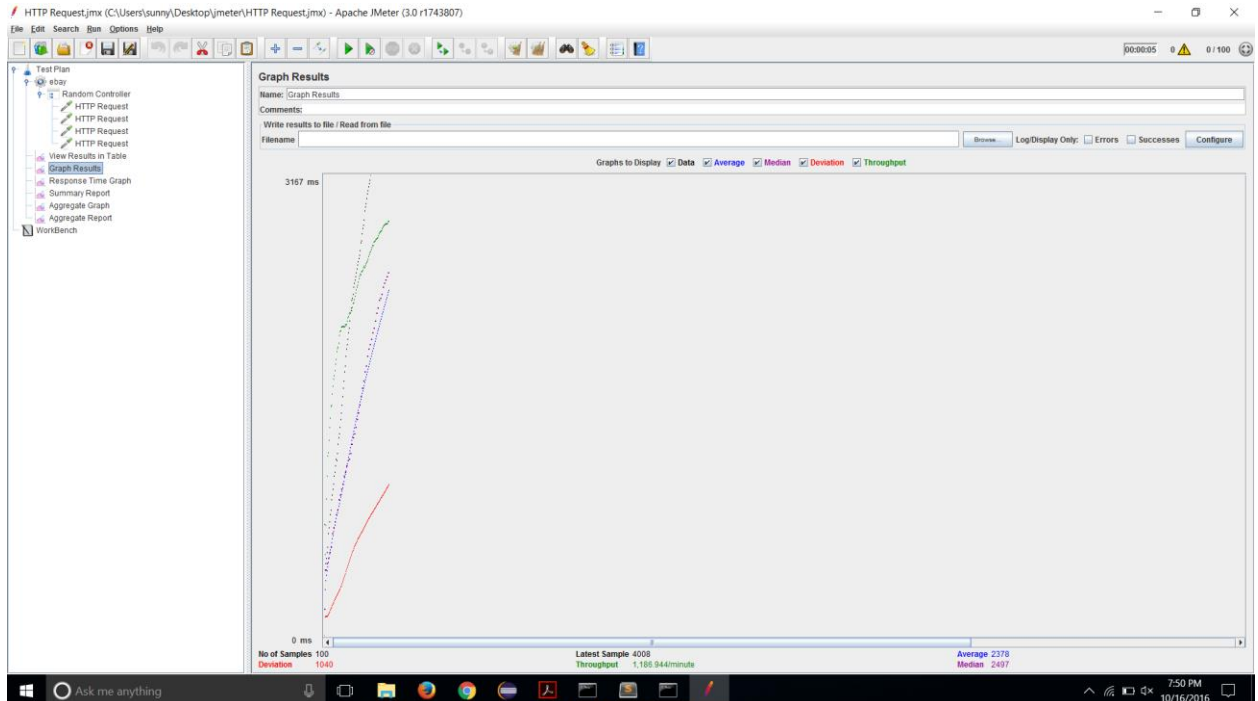
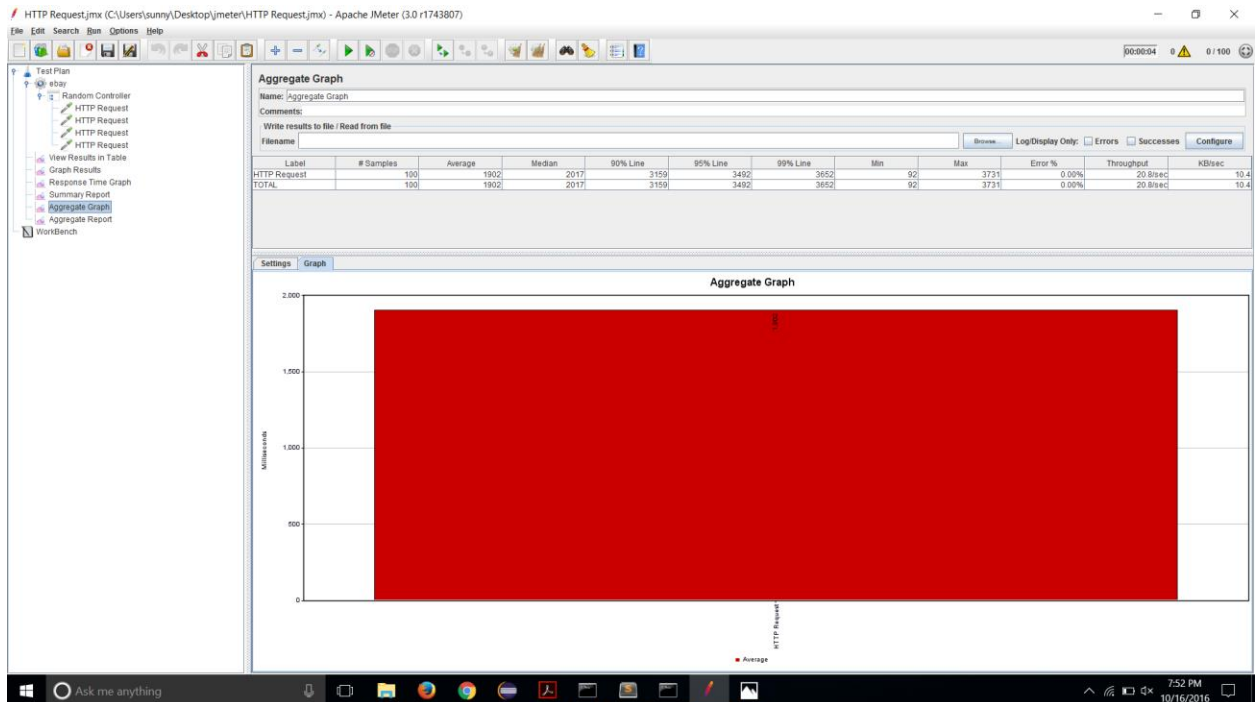


For 500 concurrent users:

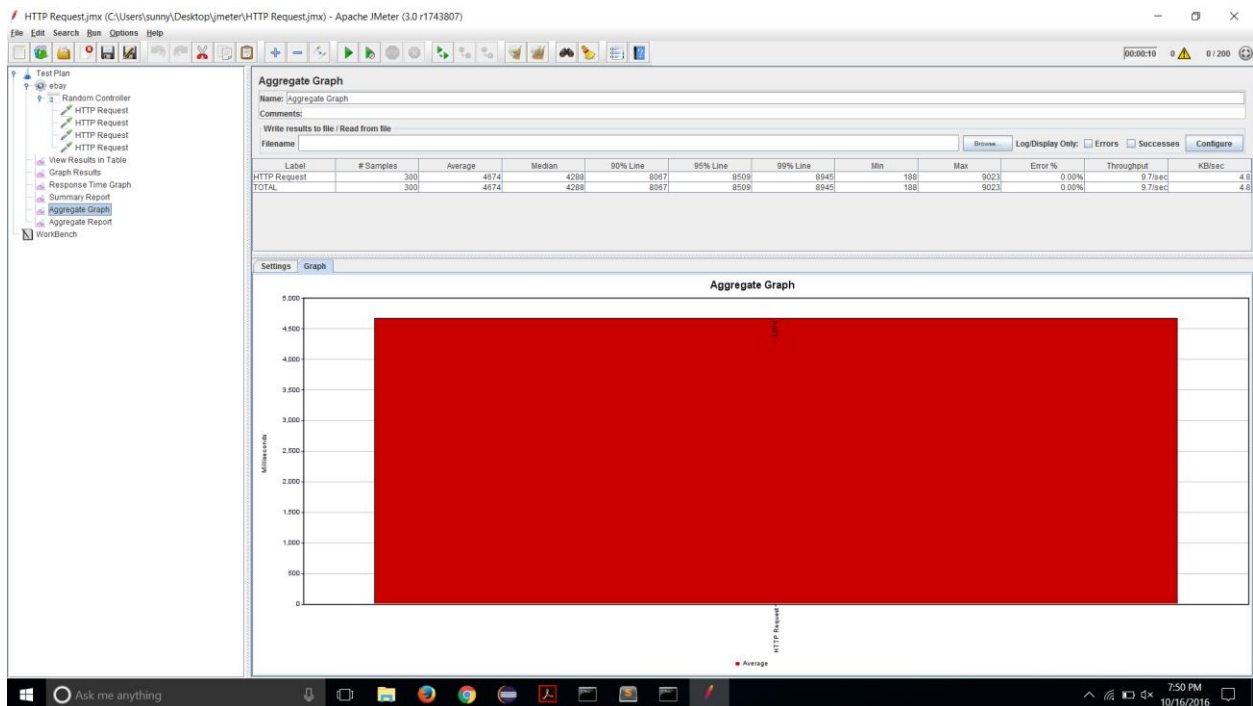
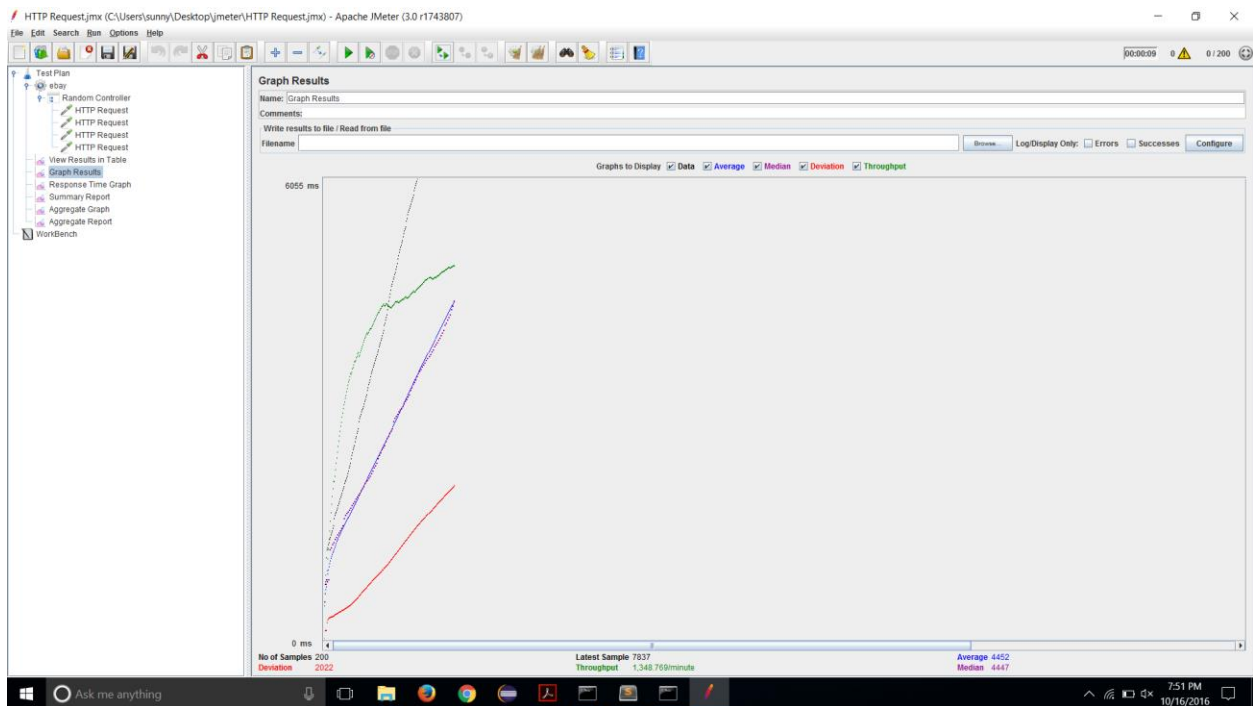


JMeter test without connection pooling

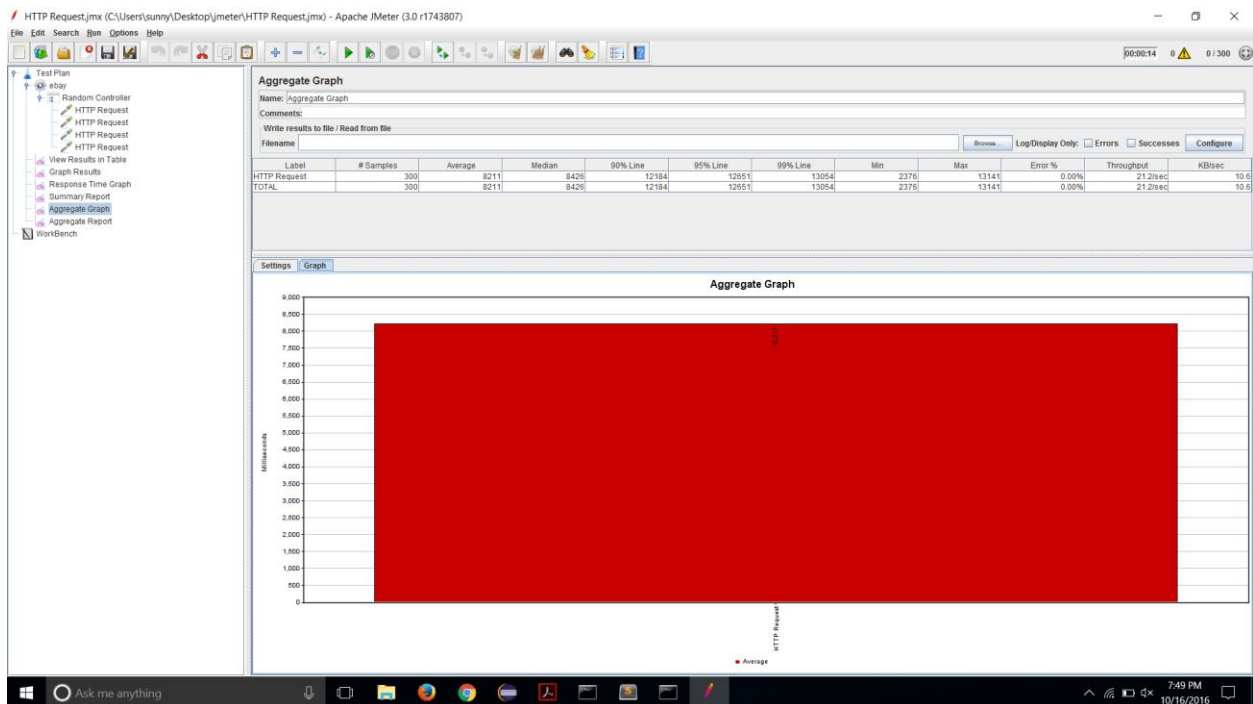
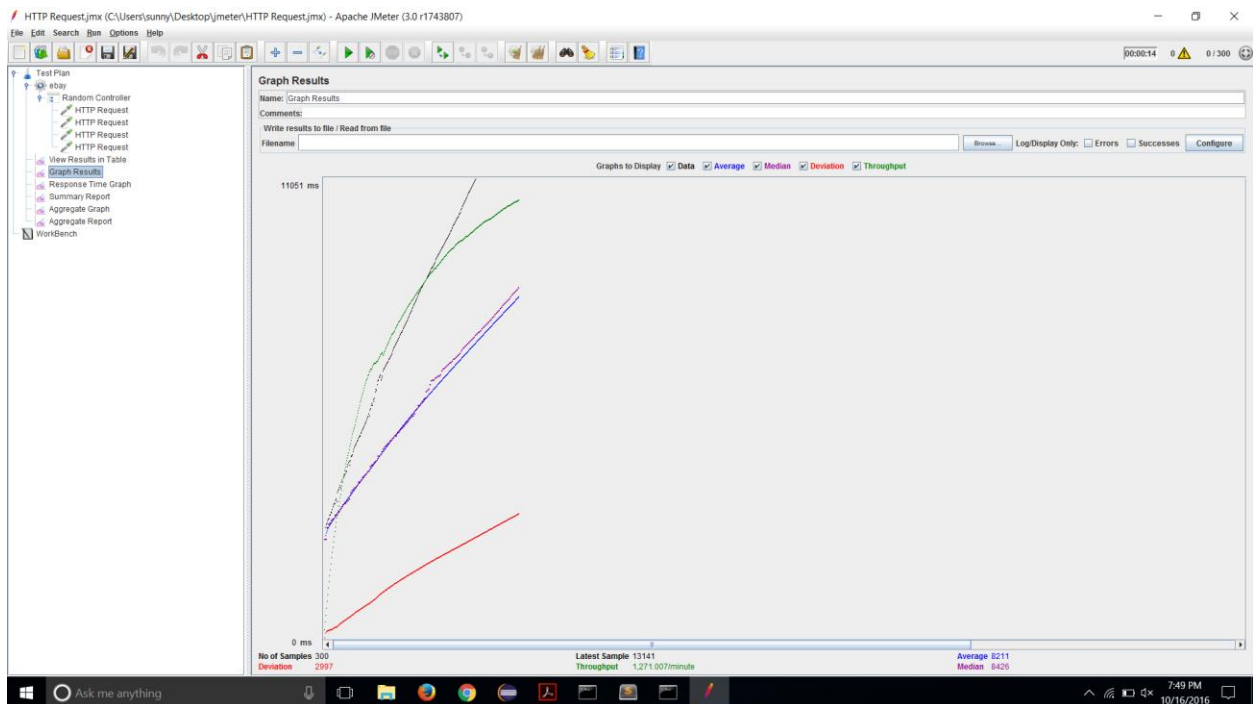
For 100 concurrent users:



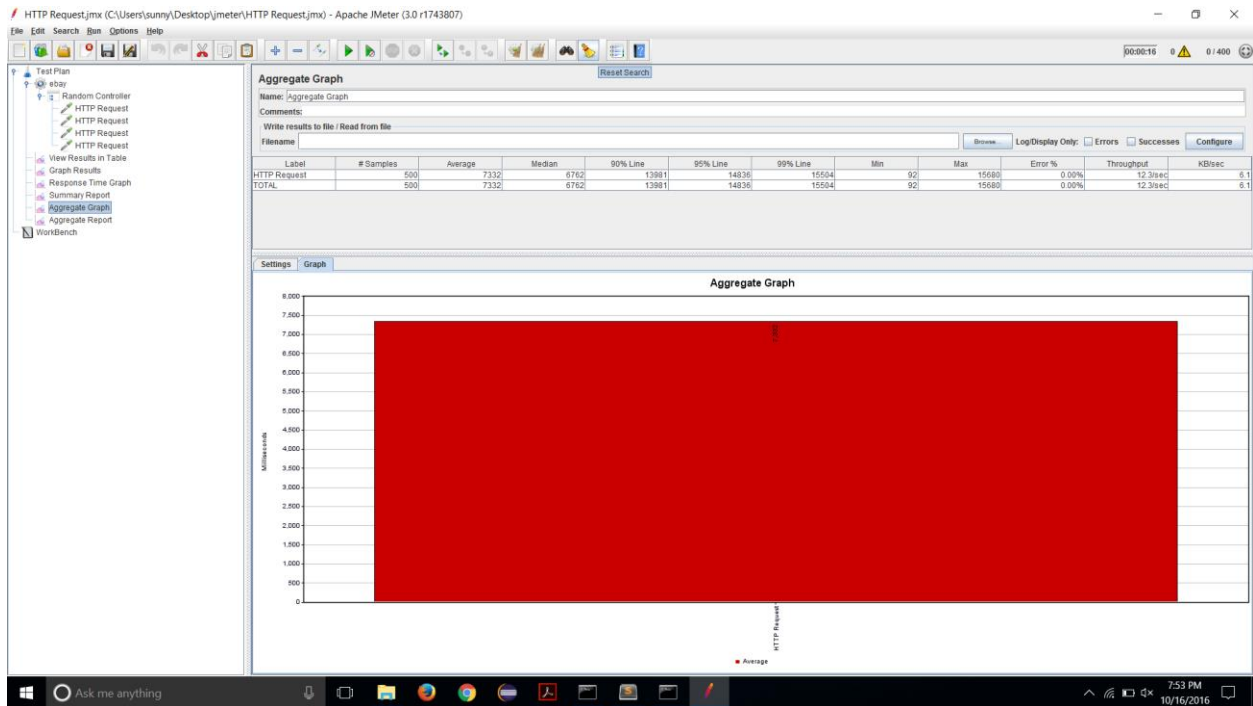
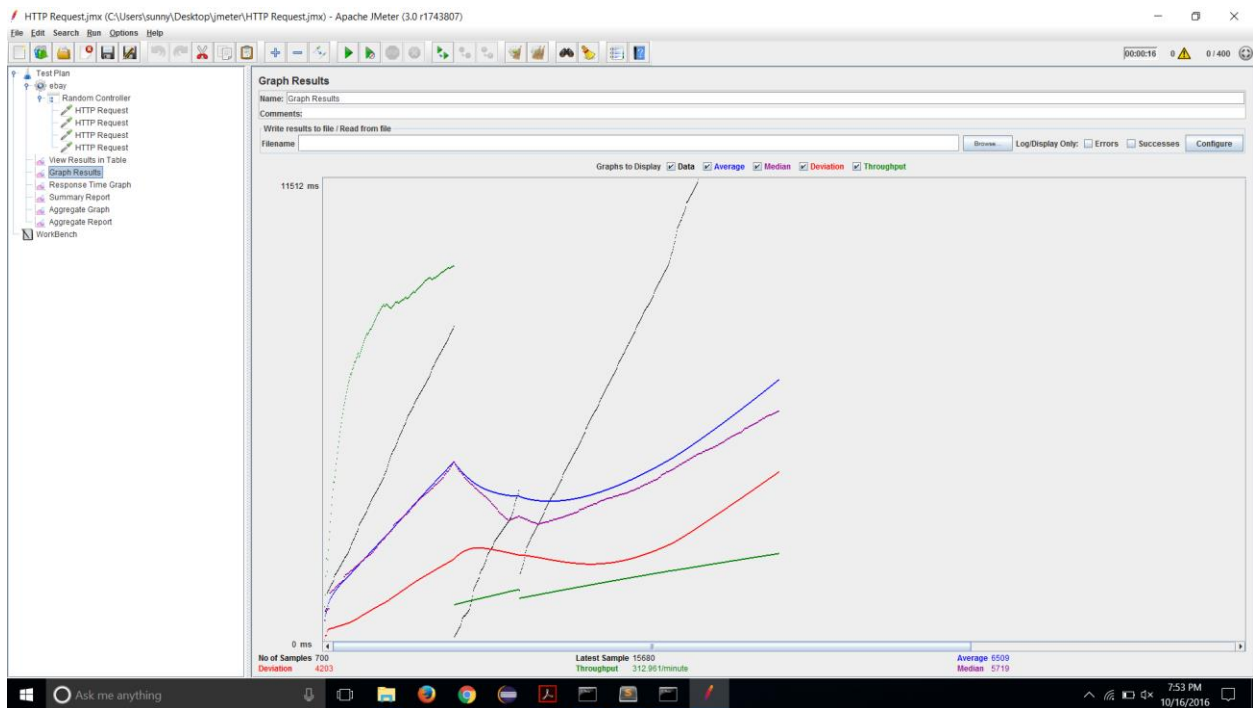
For 200 concurrent users:



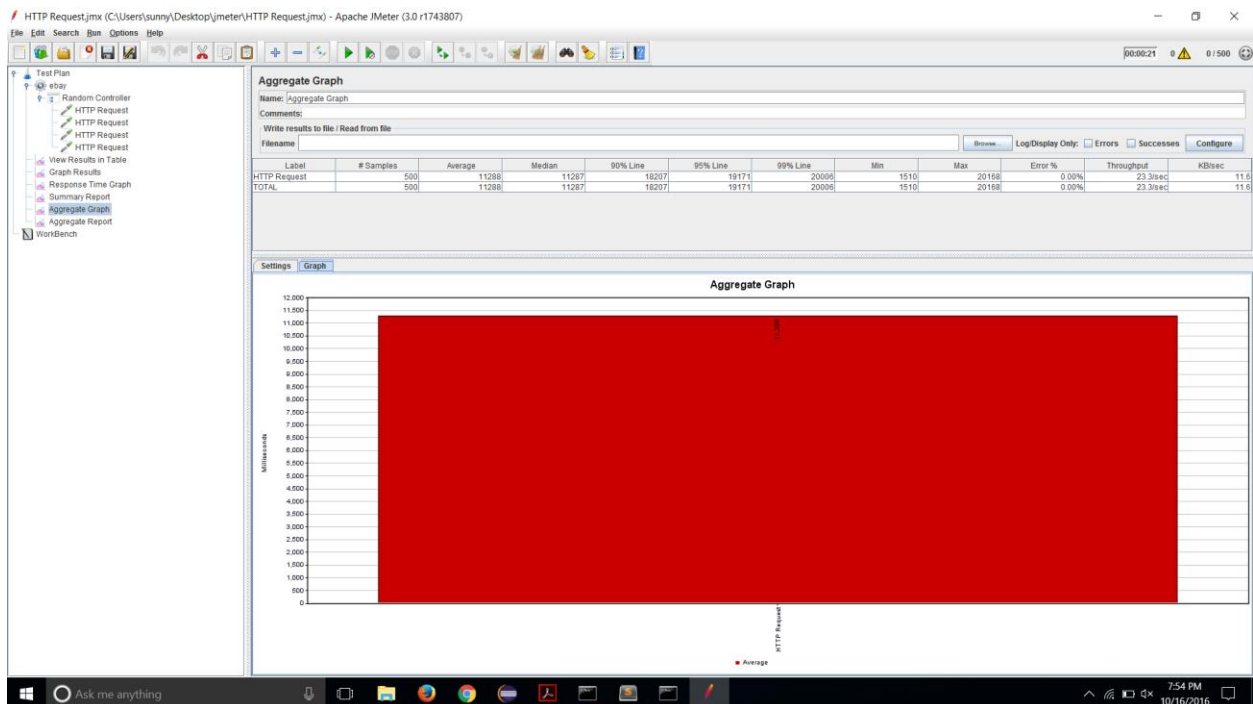
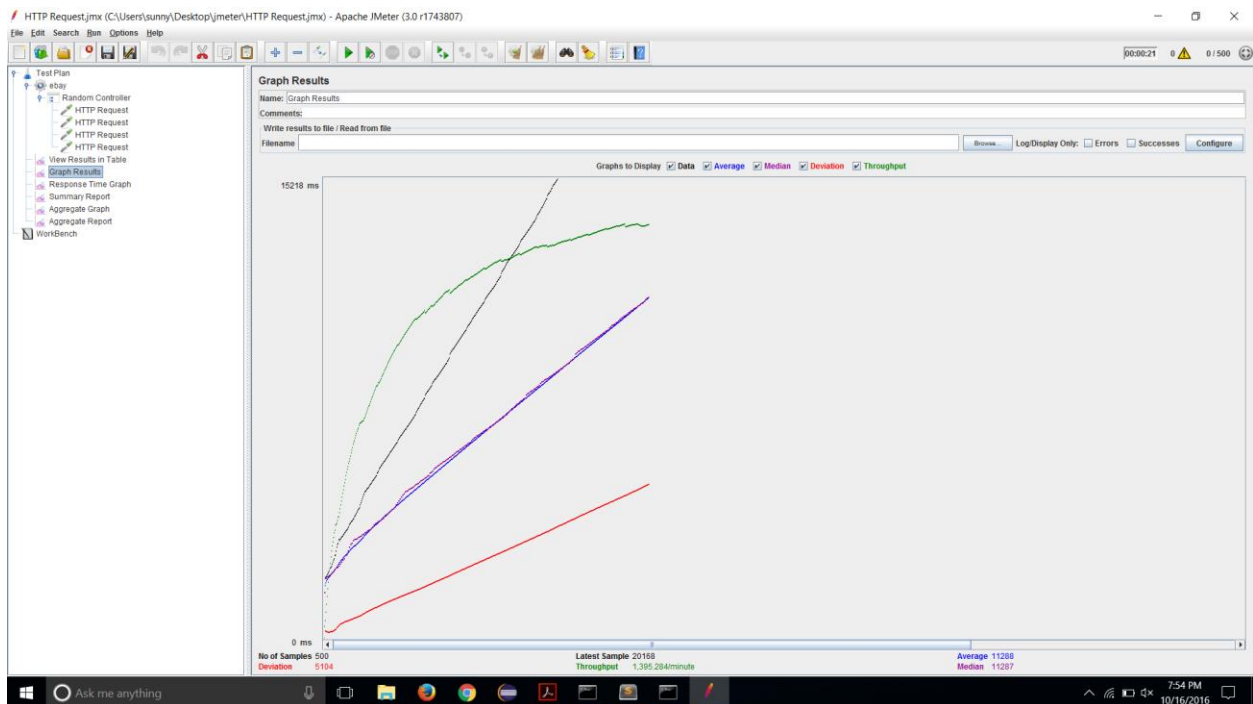
For 300 concurrent users:



For 400 concurrent users:



For 500 concurrent users:



Part-3 Questions:

1. Explain the encryption algorithm used in your application. Mention different encryption algorithms available and the reason for your selection of the algorithm used.

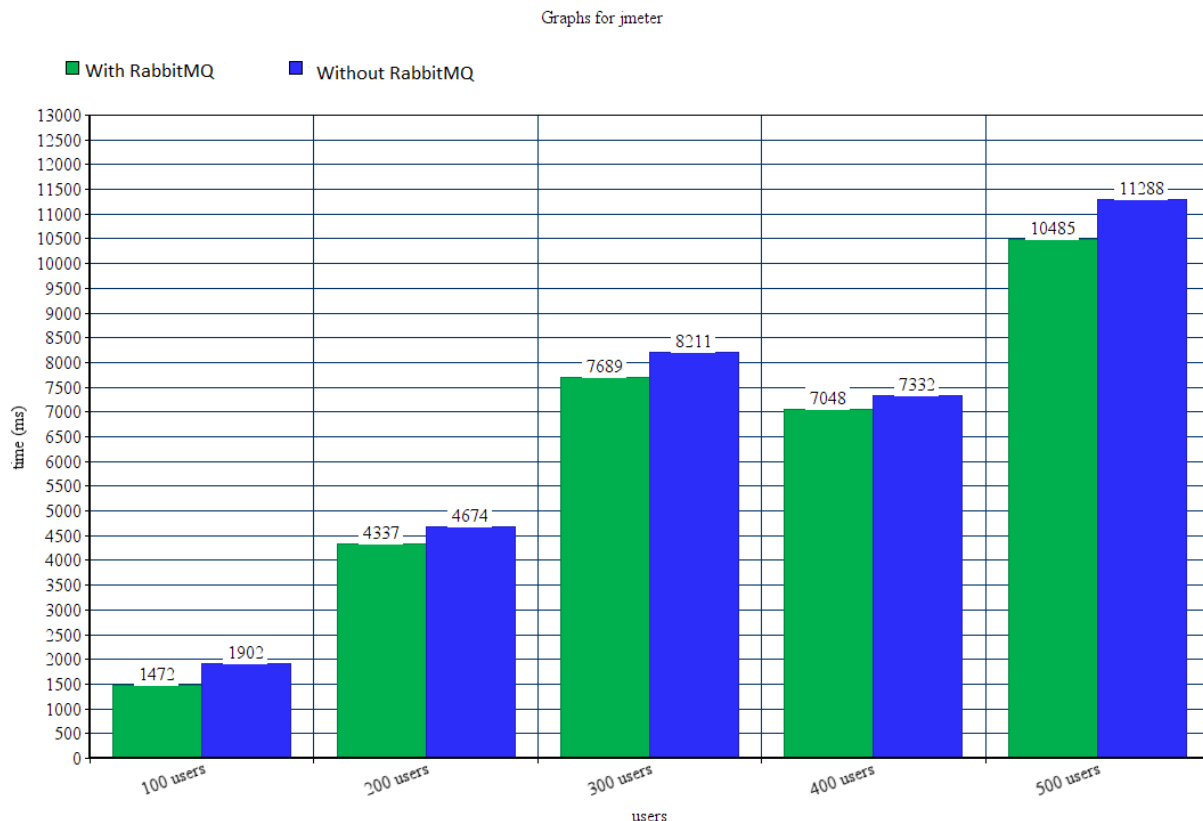
Answer:

Blowfish, Triple DES, RSA, Twofish, AES are different encryption algorithms used today. In my application I have used Advanced Encryption Standard(AES) encryption algorithm. The Advanced Encryption Standard (AES) is an encryption algorithm for securing information in commercial transactions in the private sector. AES algorithm works on the principle of Substitution Permutation network. AES is a symmetric key encryption standard adopted by the U.S. government. AES is more secure than 3DES means it is less susceptible to cryptanalysis. The Advanced Encryption Standard consists of three block ciphers. They are: AES-128, AES-192, and AES-256. Each of the above standard ciphers is 128-bit block size with key sizes of 128, and 192 & 256 bits respectively. AES is faster in both hardware and software.

I have used MySQL AES_ENCRYPT() function to encrypt passwords, AES_ENCRYPT(string, keystring) function takes two arguments one is string to be encrypted and other is key to encrypt password. It will encrypt password and then store it in database.

2. Compare the Results of the graphs with and without connection pooling. Explain the results in detail. Describe the algorithm of connection pooling used in your application.

Answer:



Comparison of throughputs

It is visible from screenshots that if we don't implement connection pooling average response and request time increases slightly. Connection pooling makes process of creating and assigning connections faster. We will not have to create connection object everytime when user requests for resource.

In my application I have used collections to implement connection pooling. From collections data structure I have used List to implement pooling. Firstly collections will be pushed to pool. When user requests for connection it will pop connection out from the pool. It will create pool of connections and assign it to database. I have set maxlength to 100 so maximum 100 connection will be

created. When a new connection is needed it will borrow connection from pool and release it when it's work is complete. Connection pool will reduce request and response time and improve performance of application.

3. How would you implement Request Caching? Explain in detail. No need to implement a function – use pseudo code or detailed explanation.

Answer:

Caching is implemented for faster retrieval of information. Caching would be useless if it did not significantly improve performance. The goal of caching in is to eliminate the need to send requests in many cases, and to eliminate the need to send full responses in many other cases.

When we cache new request we would only cache the URL and not the entire API call because if we do so it would overload caching and strategy to make performance better will not be fulfilled. If the response's headers tell the cache not to keep it, it won't. If a representation is stale, the origin server will be asked to validate it, or tell the cache whether the copy that it has is still good. When URL is requested for the first time new entry will be generated for that URL. When URL is requested again it will first check headers for cache and after that it will go to server for resources.