

**SAN JOSE STATE UNIVERSITY  
SOFTWARE ENGINEERING DEPARTMENT  
CMPE 273 – ENTERPRISE DISTRIBUTED SYSTEMS  
CLASS PROJECT**



**VIDEO LIBRARY MANAGEMENT SYSTEM**

**PROJECT REPORT**

**Team 3**

**Abhi Shah(008424052)**

**Meet Mehta(008648939)**

**Snehal D'mello(008647639)**

**Sweta Patel(008671754)**

**Vidhi Shah(008141068)**

## **Individual member contribution in project**

### **Abhi Shah**

Client GUI, client-side servlets, client-side validation, Database designing, Jmeter Performance measurement and server side validation.

### **Meet Mehta**

Designing of web services and its Server side methods implementation, Database designing, Project Report and deployment on Google Cloud.

### **Snehal D'mello**

Designing of web services and its Server side methods implementation, Database designing and Project Report.

### **Sweta Patel**

Designing of web services and its Server side methods implementation, Connection Pooling, Database designing and Project Report.

### **Vidhi Shah**

Client GUI, client-side servlets, Junit testing and Project Report.

## Object Management Policy

### Abstract:

In this project, we have designed a 3-tier system that implements the functions of Video Library Management System. Our system keeps track of all the movies in a database as well as members and rents out the movies to the members as needed. Depending upon the type of the member, we have limitations on the number of movies issued. Fine is imposed on the members if the movie is not returned back. It manages the transactions done by the members very well and depending on the type of member, he is charged in a different way. Developing a website is not a challenge but making it accessible to all the customers at the same time without delay is a real big challenge. Thus a website has to be scalable and should manage the increase in customer traffic very well without delays. In such cases, system's performance is put to test. We have tried overcoming the challenge by making the system efficient and have designed to make it more scalable and reliable.

### Introduction:

In this electronic world, millions of websites are being created but the ultimate winner is the one with great performance and the one that is able to keep data secure. Major challenges faced in website creation are:

#### 1. Accessibility:

Our website has to be accessible to people with different backgrounds, abilities and disabilities. The website should be designed in such a way that all the people have equal access to information and features of the website.

#### 2. Compatibility:

One of the most common aspect of the website that is overlooked is the browser compatibility. A website that is developed has to be compatible with wide variety of browsers so that different users with different browsers have access to the website without facing any browser compatibility issues. Thus, a website has to be tested on different browsers.

#### 3. Navigability:

The navigation done on the website has to be really simple so that any layman customer can access it too without facing any problem. If customers find it hard to navigate on your website, they will leave quickly and would never want to come back to the website again. Thus navigation should have an effective structure.

#### 4. Readability:

The text and the fonts used on the website should be such that it enhances readability. Thus the website users focus on making it more readable for users regardless of age groups and background.

Thus we have designed our website keeping in mind the above points and security concerns and worked more on improving the performance so that it can handle more number of customers.

**Implementation:**

Video Library has vast collection of movies. When a customer wants to buy movies on a Video Library Management System, he has to SignUp first. Customer will be having two types of memberships to choose from: 1. Simple Members 2. Premium Members. Simple Members are allowed to rent maximum 2 movies while Premium customers have liberty to choose 10 movies at a time. As soon as a Simple Member buys a movie online, he has to pay for the movie rented while the Premium Member pays for the subscription fees. This payment for the subscription fees could be done monthly, quarterly or annually. While signing up, they are asked to give information such as name, address, membershipId etc. Fine is imposed if the movie is not returned on time. The User can see his information anytime with all the issued movies. We have created an admin to maintain all the information. He is responsible for managing the users as well as movies. Also, we have added the password encryption which will help to keep the member's account safe and secure. All these features makes our system better at the moment. Making it best would be a further challenge.

**Performance**

Apart from implementing technical details, one should always take care about web site's performance, scalability and robustness. As mentioned in a project description, we should take care of the real time scenarios of huge number of clients and movie data. In case of such bulky scenarios, database as well as website should work without any kind of problem and web site crash. Also in case of multiple queries or requests at the same time, it should work faster as much as possible. Below are the techniques those we have used in our project to improve the performance.

**1) Connection Pooling:**

Connection pooling is a one of the finest solution for performance when multiple requests occur at the same time. Normally when client do request to the database, it creates new connection to access data. Once data is fetched, connection gets closed. Same scenario executes every time when client make any request. This was a single client scenario. Now assume a case where thousands of clients make a request to the database at the same time. Also when number of client's requests gets increase, time required to fetch the content is also get increased due to the many resources utilization. So in such a case normal connection and disconnection is not feasible way for efficient working. So what we need in such a scenario is connection pooling. Main purpose of the connection pooling is performance enhancement. In connection pooling, we can open multiple number of connections (have to specify this number in java file) based on number of clients. This open connections assign to the client request when arrives and once client done with his task, he returns this connection to the pool back which will be available for the next client request. This way less time is spent in opening and closing of connections and heavy weight resources can be managed.

**2) INNODB over MYISAM engine in MYSQL:**

MYISAM uses table-level locking while MYSQL uses row-level locking system. So if for an example we have xxx no. of users updating anything into our database then in MYISAM the query processes one by one as table is locked by each and every query while in case with INNODB concurrent execution occurs as only row which is updating is locked. which reduces latency time. For example, a single query takes x milliseconds to update then in MYISAM have total time  $n \times x$  milliseconds while in MYSQL have only x milliseconds.

Our Database uses more Insert and Update queries as compared to Select queries. Thus, we uses INNODB as our MYSQL engine

**3) Caching:**

Caching is technique of storing the previously fetched data temporarily on server side and eliminates the costly trip to database.

In our project we use HashMap technique in which we use (key, values) pairs. When any customer signs in to our website whole customer detail is stored as value and key for the same is his membershipID. So whenever customer signs in again and if key for customer is already there in HashMap then there is no costly trip to database for fetching its details and directly returns the value from HashMap. Again, which increase the overall performance and reduce latency time.

**4) Database Tuning:**

- **Using full column\_names instead of \* in select query:** We have used full column\_names in all of our select queries instead of using \*. The reason behind this is when we fire "Select \* from table\_name" MYSQL initially have to fetch the column names and then have to fire the query while in using column\_names directly MYSQL firs not have to find out the column\_names and hence execution of our query is slightly faster.
- **Using PreparedStatement over Statement:** the main difference between PreparedStatement and Statement is that PreparedStatement precompiles the query for the first time and then execute it everytime while in Statement it have to compile the query each and every time.

For example

✓ In Statement query

```
" select customer_name from customer where membershipID="+membershipID+"";
stmt.executeUpdate();
```

here MYSQL have to make access plan (optimal way to execute SQL query) everytime it fires the query " select customer\_name from customer where membershipID=12345"

✓ While in Prepared Statement when we fire the query

```
"select customer_name from customer where membershipID=?"
ps.setString(1,membeshipID);
```

```
ps.executeUpdate();
```

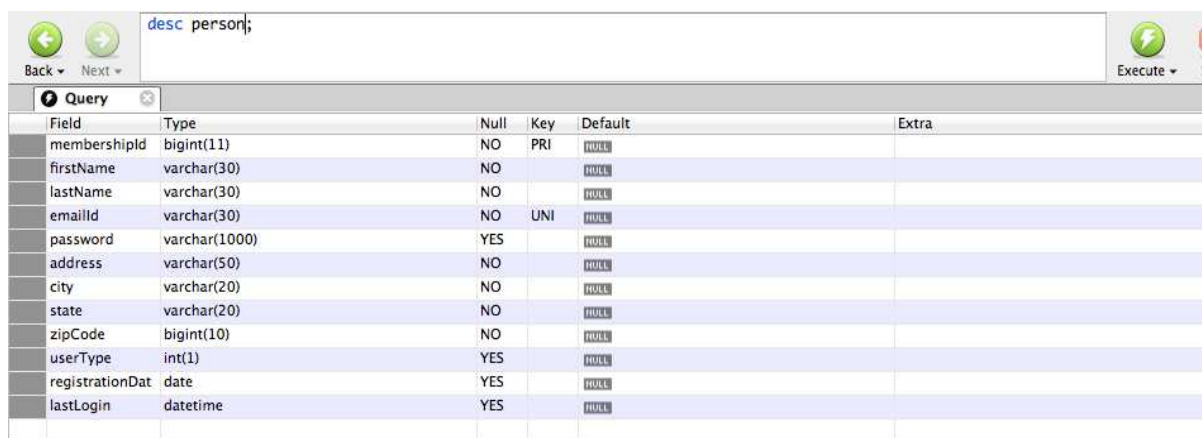
Whenever the query is fired to MYSQL. It makes its own access plan with " select customer\_name from customer where membershipID=" and this plan is used everytime when similar query is fired.

- **Avoided Like query:** We have used Like query in search functionality only and avoid to use it in any other functionality.

## Database Table Details:

### 1. Person:

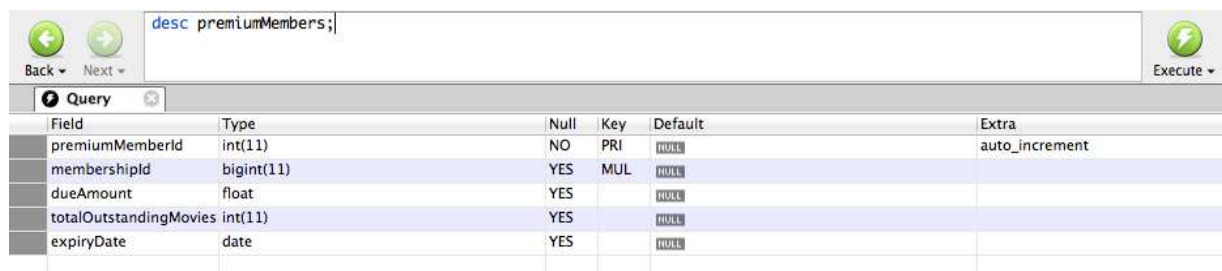
This table contains information of all the new customers added and their login information.



Field	Type	Null	Key	Default	Extra
membershipId	bigint(11)	NO	PRI	NULL	
firstName	varchar(30)	NO		NULL	
lastName	varchar(30)	NO		NULL	
emailId	varchar(30)	NO	UNI	NULL	
password	varchar(1000)	YES		NULL	
address	varchar(50)	NO		NULL	
city	varchar(20)	NO		NULL	
state	varchar(20)	NO		NULL	
zipCode	bigint(10)	NO		NULL	
userType	int(1)	YES		NULL	
registrationDate	date	YES		NULL	
lastLogin	datetime	YES		NULL	

### 2. PremiumMembers:

This table contains information about the Premium Members. All the customers having userType as 0 in Person table will be added to the table PremiumMembers.



Field	Type	Null	Key	Default	Extra
premiumMemberId	int(11)	NO	PRI	NULL	auto_increment
membershipId	bigint(11)	YES	MUL	NULL	
dueAmount	float	YES		NULL	
totalOutstandingMovies	int(11)	YES		NULL	
expiryDate	date	YES		NULL	

### 3. SimpleMembers:

This table contains information about the Simple Customers. All the customers having userType as 1 in Person table will be added to the table SimpleMembers.

Back Next desc simpleMembers; Execute

Field	Type	Null	Key	Default	Extra
simpleMemberId	int(11)	NO	PRI	HULL	auto_increment
membershipId	bigint(11)	YES	MUL	HULL	
rentForMoviesIssued	float	YES		HULL	
totalOutstandingMovies	int(11)	YES		HULL	

#### 4. **Movie:**

This table contains all the information about the existing movies available in the Video Library.

Back Next desc movie; Execute

Field	Type	Null	Key	Default	Extra
movieId	int(11)	NO	PRI	HULL	auto_increment
movieName	varchar(30)	YES		HULL	
movieBanner	varchar(30)	YES		HULL	
movieReleaseDate	date	YES		HULL	
movieRent	float	YES		HULL	
movieCategory	varchar(15)	YES		HULL	
availableCopies	int(11)	YES		HULL	

#### 5. **Cart:**

This table contains all the information about the movies available in the cart before final checkout.

Back Next desc cart; Execute

Field	Type	Null	Key	Default	Extra
cartId	int(11)	NO	PRI	HULL	auto_increment
membershipId	bigint(20)	NO	MUL	HULL	
movieId	int(11)	YES	MUL	HULL	

#### 6. **Transaction:**

This table contains all the movies purchased by all the Members. Once the member does the final checkout, data enters into the transaction table.

Back

Next

desc transactions;

Execute

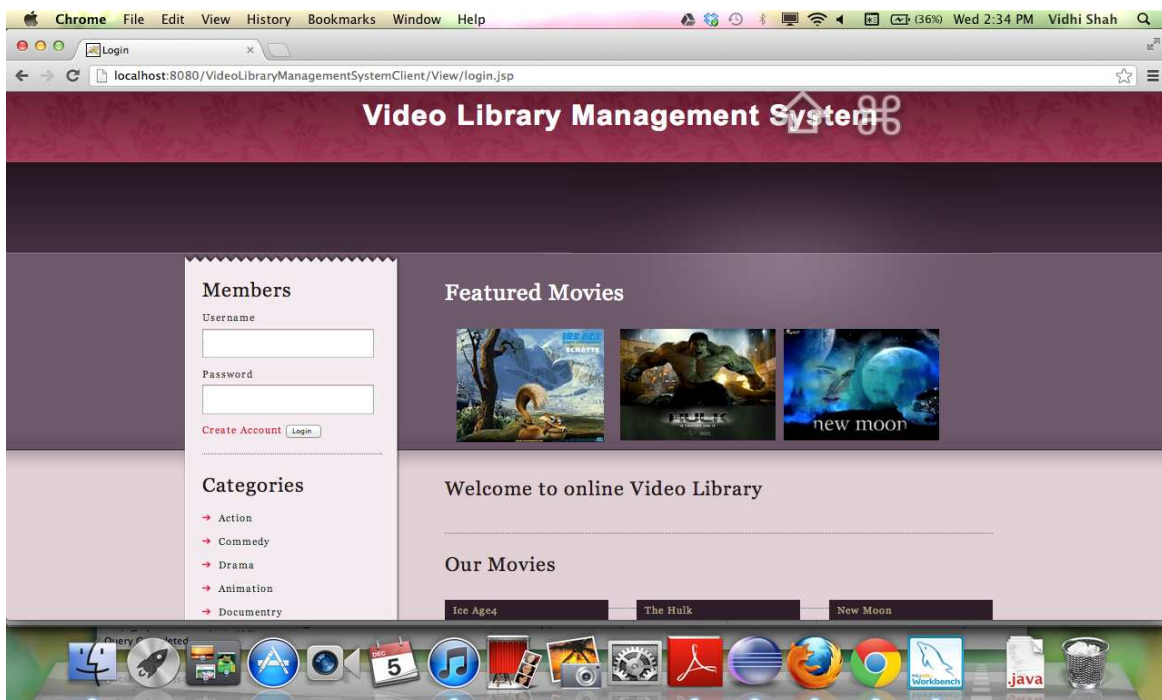
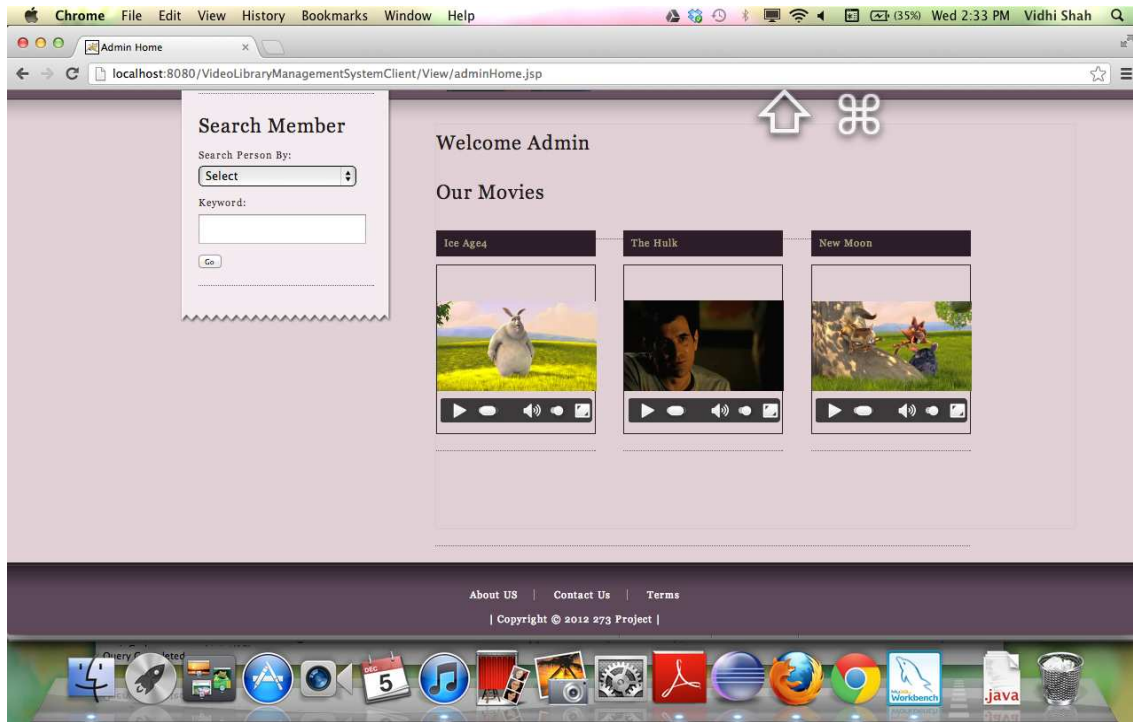
Query

Field	Type	Null	Key	Default	Extra
transactionId	bigint(20)	NO	PRI	NULL	auto_increment
membershipId	bigint(11)	YES	MUL	NULL	
movieId	int(11)	YES	MUL	NULL	
issueDate	date	NO		NULL	
dueDATE	date	YES		NULL	
actualReturnDate	date	YES		NULL	
fineAmount	float	YES		NULL	



## Screenshots

### Home Screen



**Admin****Add Movie**

Search Movie

Search Movie By:

Select

Keyword:

Go

Search Member

Search Person By:

Select

Keyword:

Go

Featured Movies

Add Movie

All fields are compulsory

Movie Name: movie8

Release Date: 12/10/12

Movie Category: Documentary

Movie Banner: banner8

Available Copies: 10

Rent Amount: 12

**Update Movie**

Search Movie

Search Movie By:

Select

Keyword:

Go

Search Member

Search Person By:

Select

Keyword:

Go

Featured Movies

Update Movie

Movie Name: movie4

Release Date: 12/6/12

Movie Category: Comedy

Movie Banner: banner4

## List Movies

**Search Movie**

Search Movie By:

Select

Keyword:

Go

**Search Member**

Search Person By:

Select

Keyword:

Go

**Featured Movies**

**List of Movies**

Number	Movie Name	Banner	Release Date	Category	Rent(\$)	Copies Available	Update	Delete
1	movie1	banner1	2012-12-10	Action	15.0	7	<a href="#">Update</a>	<a href="#">Delete</a>
2	movie2	banner2	2012-12-04	Comedy	11.0	7	<a href="#">Update</a>	<a href="#">Delete</a>
3	movie3	banner3	2012-12-02	Family	22.0	15	<a href="#">Update</a>	<a href="#">Delete</a>
4	movie4	banner4	2012-12-06	Comedy	11.0	13	<a href="#">Update</a>	<a href="#">Delete</a>
5	movie5	banner5	2012-12-03	Action	12.0	6	<a href="#">Update</a>	<a href="#">Delete</a>
6	movie6	banner6	2012-12-19	Horror	15.0	10	<a href="#">Update</a>	<a href="#">Delete</a>

## Search Movie

**Search Movie**

Search Movie By:

Name

Keyword:

movie3

Go

**Search Member**

Search Person By:

Select

Keyword:

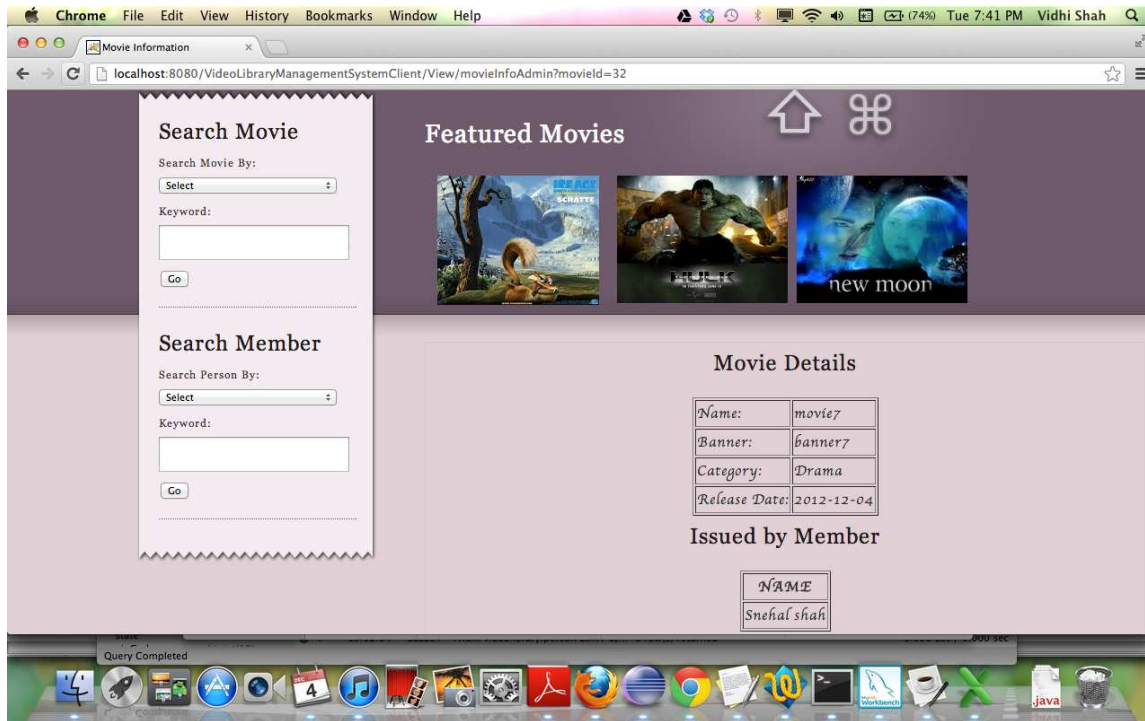
Go

**Featured Movies**

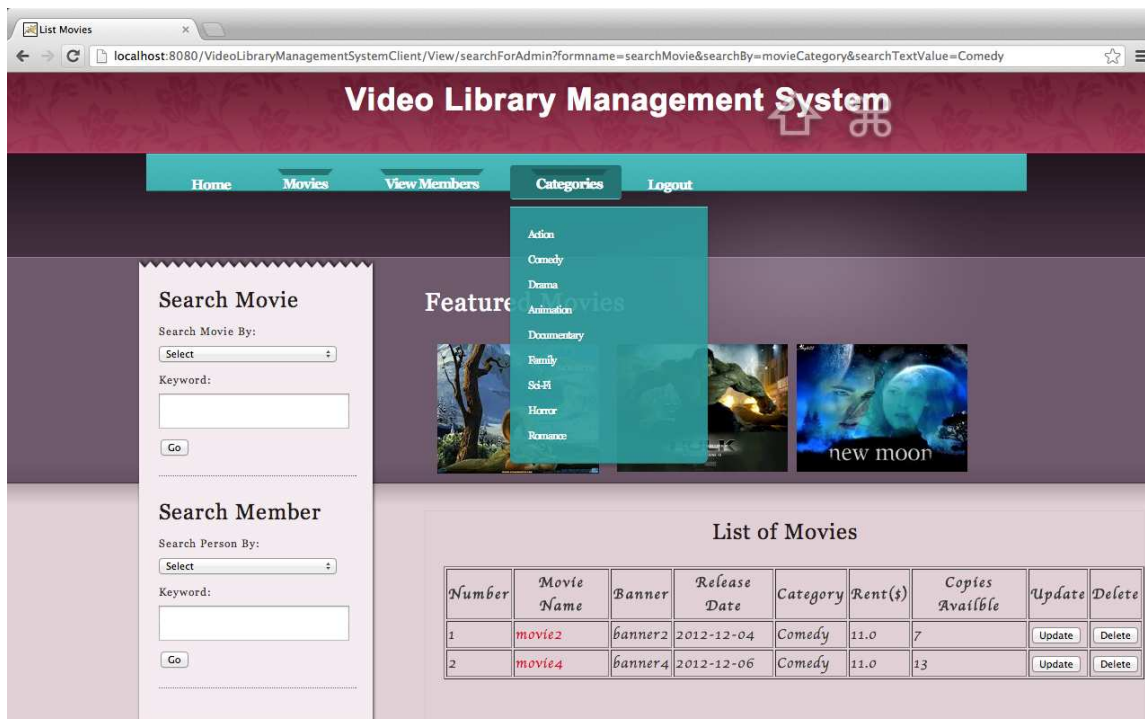
**List of Movies**

Number	Movie Name	Banner	Release Date	Category	Rent(\$)	Copies Available	Update	Delete
1	movie3	banner3	2012-12-02	Family	22.0	15	<a href="#">Update</a>	<a href="#">Delete</a>

## Movie information plus issued by members



## Movies according to Category





## List of All Members

**Search Movie**

Search Movie By:

Keyword:

**Search Member**

Search Person By:

Keyword:

**Featured Movies**

**List of All Members**

Number	Membership Id	Member Name	Email	Type	Delete
1	333333331	Snehal shah	snehal@s.com	Premium	<input type="button" value="Delete"/>
2	333333332	vidhi shah	vidhi@v.com123456	Premium	<input type="button" value="Delete"/>
3	333333333	sweta patel	s@s.com	Simple	<input type="button" value="Delete"/>
4	333333334	meet mehta	meet@m.com	Simple	<input type="button" value="Delete"/>
5	333333335	abhi shah	abhi@a.com	Simple	<input type="button" value="Delete"/>
6	333333336	kalash shah	kalash@k.com	Simple	<input type="button" value="Delete"/>

## Member information plus movies issued to him

**Search Movie**

Search Movie By:

Keyword:

**Search Member**

Search Person By:

Keyword:

**Person Details**

Email:	333333331
Name:	Snehal shah
Email:	snehal@s.com
City:	san jose

**Movies Issued by Member**

Name
movie1
movie2
movie7

## List of All Simple Members

The screenshot shows a web application interface for a video library management system. The browser address bar indicates the URL: `localhost:8080/VideoLibraryManagementSystemClient/View/adminHome?formname=displaySimple`. The page has a navigation bar with links: Home, Movies, View Members, Categories, and Logout. On the left, there are two search forms: 'Search Movie' and 'Search Member', each with a 'Select' dropdown, a 'Keyword' text input, and a 'Go' button. The main content area is divided into two sections: 'Featured Movies' displaying three movie posters (Schatze, Hulk, and New Moon), and 'List of Simple Members' displaying a table of members.

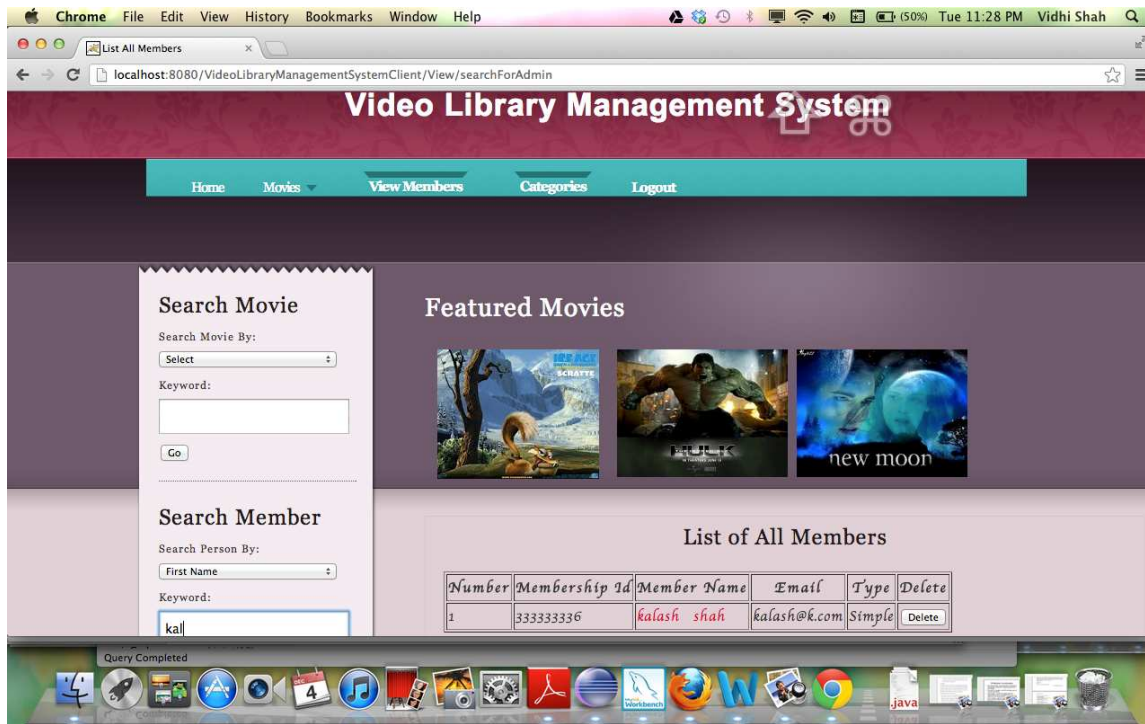
Number	Membership Id	Member Name	Email	Outstanding Movies	Delete
1	333333333	sweta patel	s@s.com	0	Delete
2	333333334	meet mehta	meet@m.com	0	Delete
3	333333335	abhi shah	abhi@a.com	0	Delete
333333336		kalash shah	kalash@k.com	0	Delete

## List all Premium Members

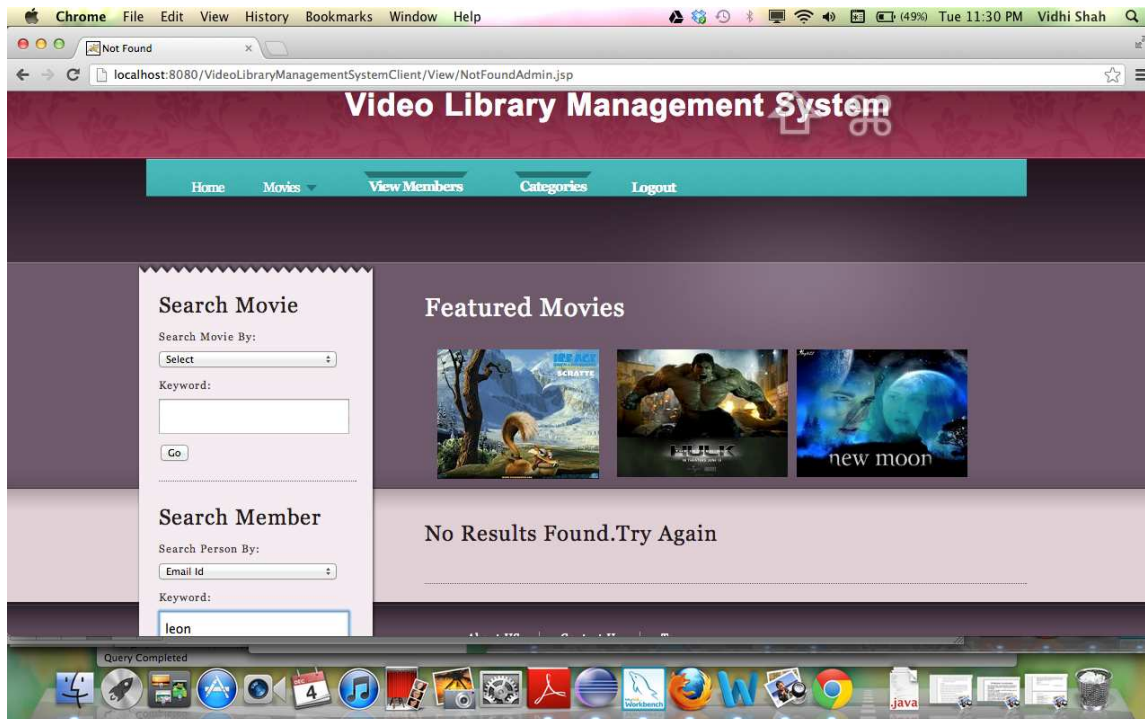
The screenshot shows the same web application interface as the previous one, but the browser address bar indicates the URL: `localhost:8080/VideoLibraryManagementSystemClient/View/adminHome?formname=displayPremium`. The layout is identical, but the 'List of Premium Members' table is displayed instead of the simple members table.

Number	Membership Id	Member Name	Email	Due Amount	Outstanding Movies	Delete
1	333333331	Snehal shah	snehal@s.com	50.0	0	Delete
2	333333332	vidhi shah	vidhi@v.com123456	50.0	0	Delete

## Search Person

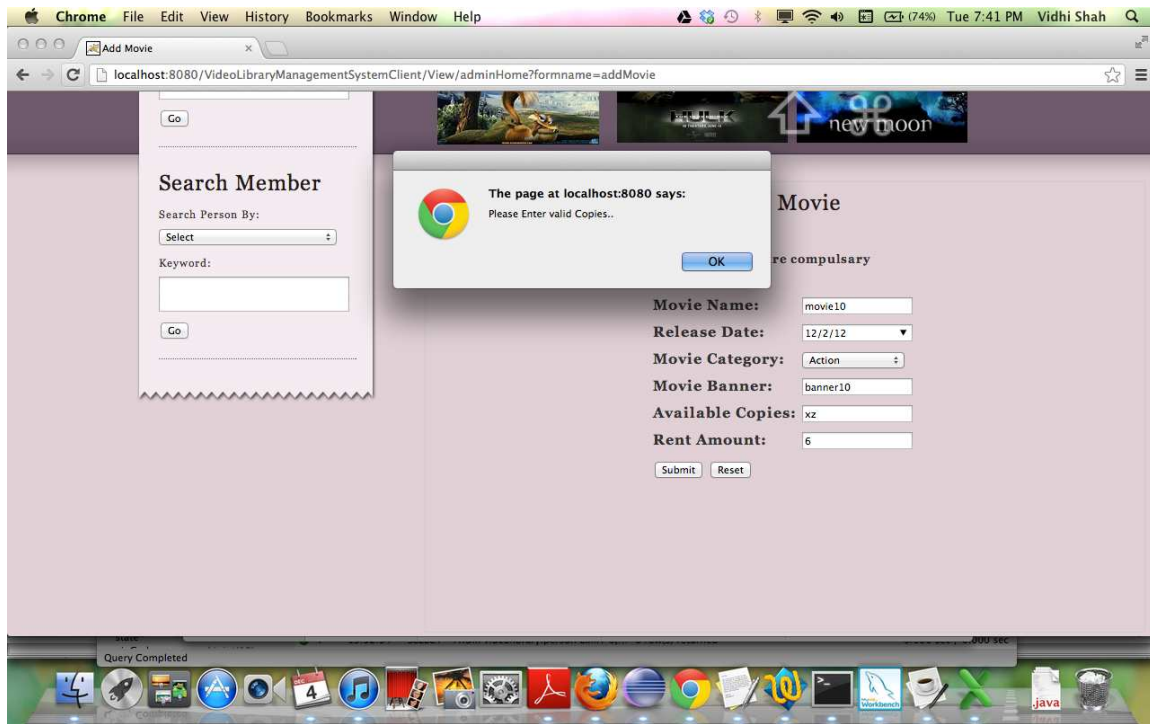


## Search value Not Found

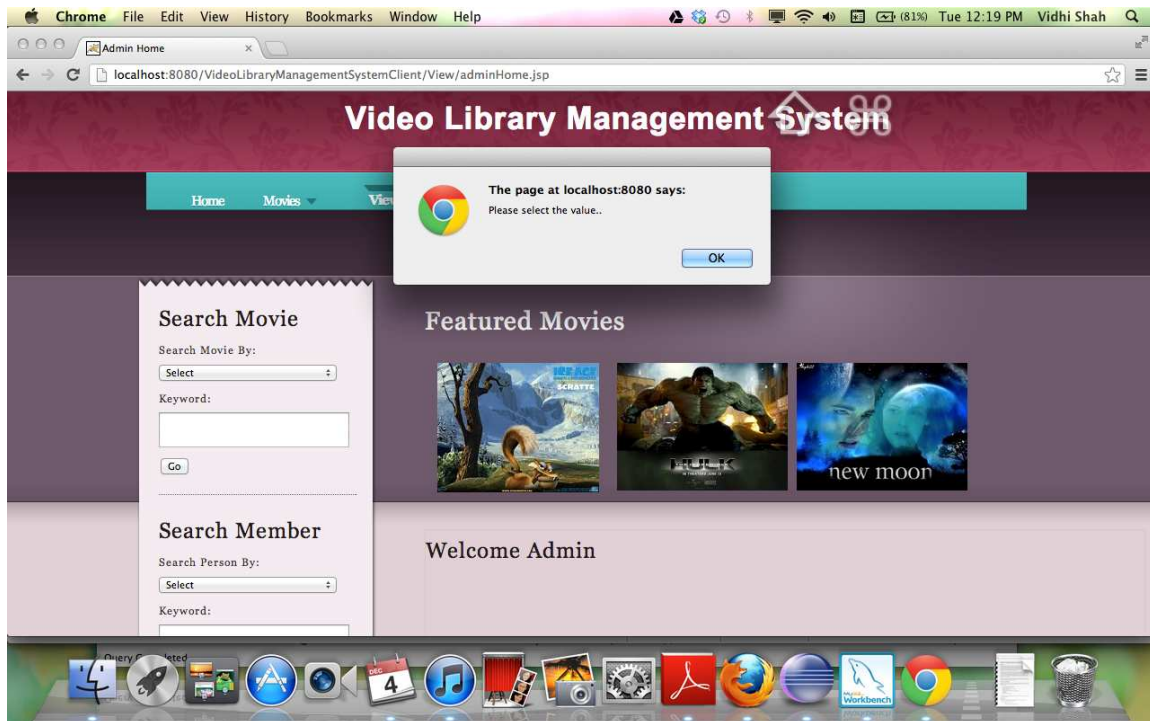




## Invalid input for Available copies



## Go Pressed before selecting any option while searching





## Member SignUp

Chrome File Edit View History Bookmarks Window Help

localhost:8080/VideoLibraryManagementSystemClient/View/signup.jsp

new moon

**Sign Up**

Membership Id:	111-11-11111 (xxx-xx-xxxx)
First Name:	vidhi
Last Name:	shah
Email:	vidhi@yahoo.com
Password:	*****
Confirm Password:	*****
Address:	v
City:	San Jose
State:	California
Zipcode:	95051
Membership Type:	Premium

Categories

- Action
- Comedy
- Drama
- Animation
- Documentary
- Family
- Sci-fi
- Horror
- Romance

When membership Id entered in invalid format

Chrome File Edit View History Bookmarks Window Help

localhost:8080/VideoLibraryManagementSystemClient/View/signup.jsp

new moon

**Members**

Username

Password

Create Account Login

**Featured Movies**

**Categories**

- Action
- Comedy
- Drama
- Animation
- Documentary
- Family
- Sci-fi
- Horror
- Romance

**Sign Up**

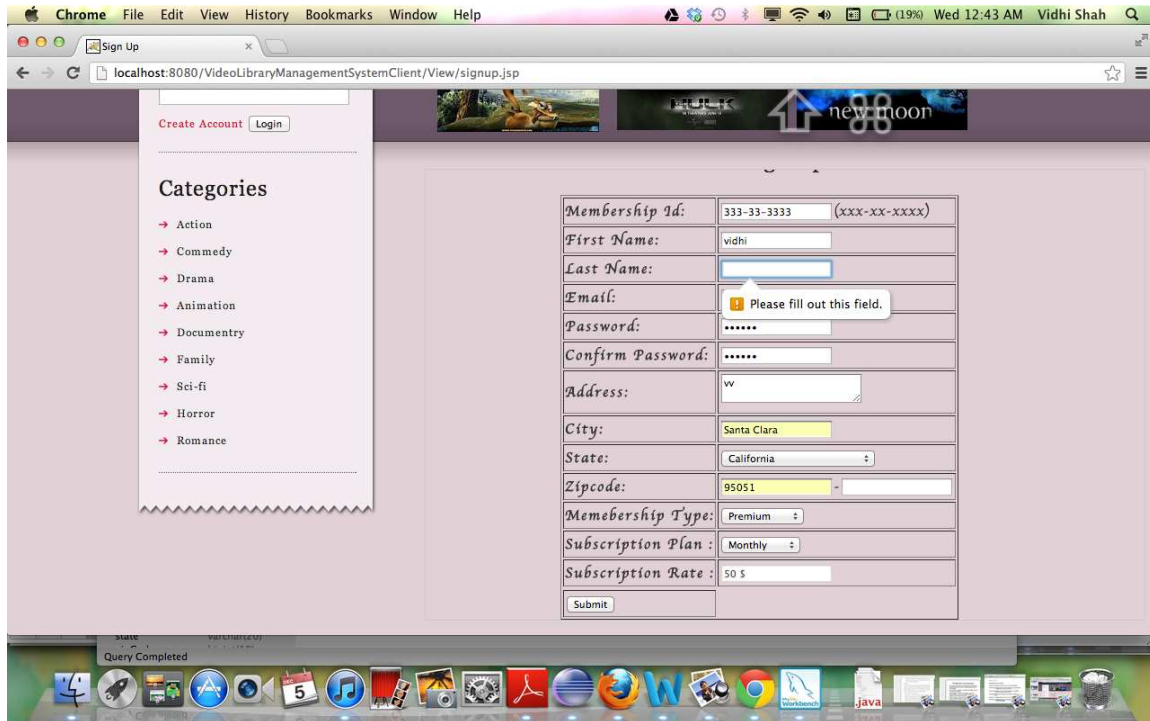
Membership Id:	333-33- (xxx-xx-xxxx)
First Name:	vidhi
Last Name:	shah
Email:	vidhi@v.com
Password:	*****
Confirm Password:	*****
Address:	vv
City:	Santa Clara
State:	California
Zipcode:	95051
Membership Type:	Premium
Subscription Plan:	Monthly
Subscription Rate:	50 \$

Submit

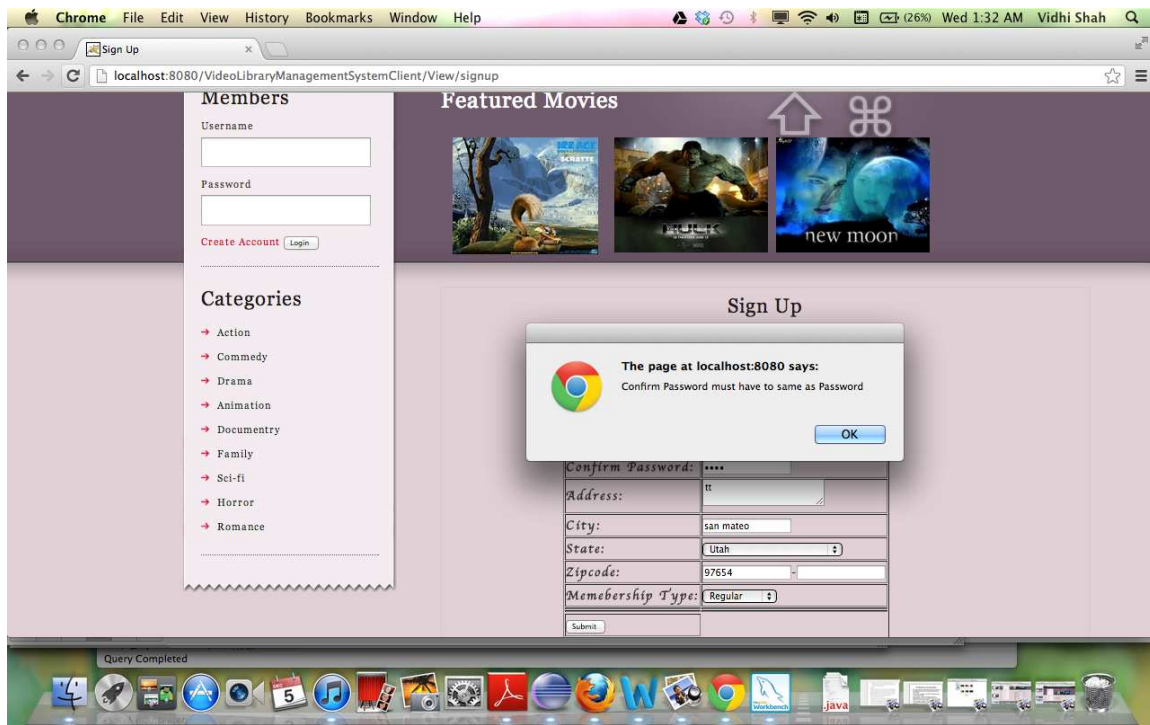
The page at localhost:8080 says:  
Enter valid membershipID in valid format.

OK

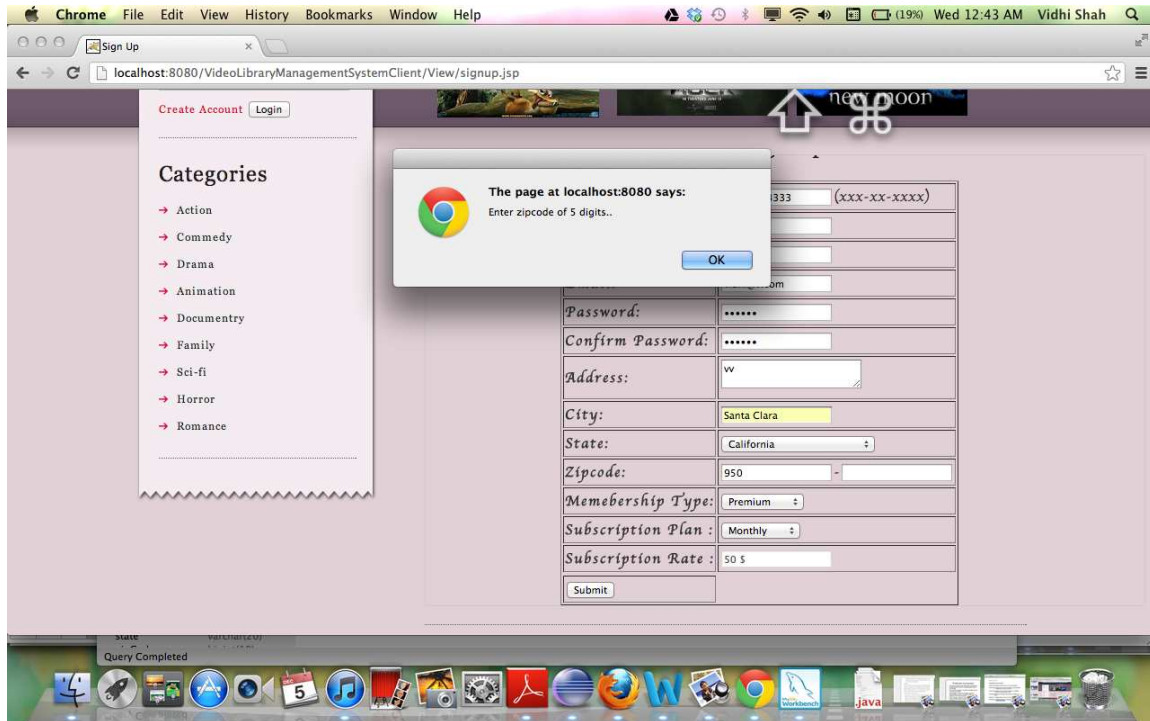
## Empty Fields validation



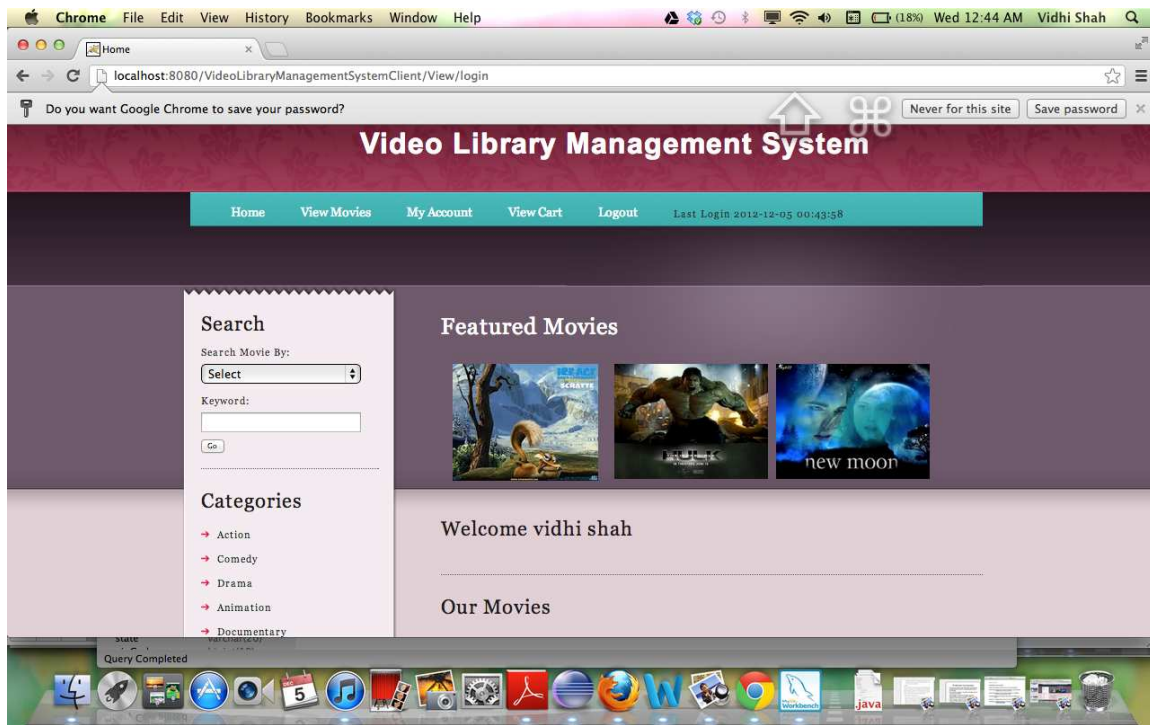
## Password mismatch



## Invalid zip code validation



## Successful Login (Premium Member)



## View All Movies

The screenshot shows a web browser window displaying the 'View All Movies' page. The browser's address bar shows the URL: `localhost:8080/VideoLibraryManagementSystemClient/View/memberhome?formname=allMovies`. The page layout includes a search sidebar on the left, a featured movies section at the top right, and a main list of movies below.

**Search Sidebar:**

Search Movie By:   
 Keyword:

**Categories:**

- Action
- Comedy
- Drama
- Animation
- Documentary
- Family
- Sci-Fi
- Horror
- Romance

**Featured Movies:**

Three movie posters are displayed: 'The Secret of NIMH', 'The Hulk', and 'New Moon'.

**List of Movies:**

Number	Movie Name	Banner	Release Date	Category	Rent(\$)	Copies Available	Issue
1	movie1	banner1	2012-12-10	Action	15.0	6	<input type="button" value="Rent It"/>
2	movie2	banner2	2012-12-04	Comedy	11.0	6	<input type="button" value="Rent It"/>
3	movie3	banner3	2012-12-02	Family	22.0	15	<input type="button" value="Rent It"/>
4	movie4	banner4	2012-12-06	Comedy	11.0	13	<input type="button" value="Rent It"/>
5	movie5	banner5	2012-12-03	Action	12.0	6	<input type="button" value="Rent It"/>
6	movie6	banner6	2012-12-19	Horror	15.0	10	<input type="button" value="Rent It"/>
7	movie7	banner7	2012-12-04	Drama	17.0	8	<input type="button" value="Rent It"/>
8	movie7	banner7	2012-12-04	Drama	17.0	9	<input type="button" value="Rent It"/>
9	movie8	banner8	2012-12-10	Documentary	12.0	10	<input type="button" value="Rent It"/>
10	movie10	banner10	2012-12-02	Action	6.0	6	<input type="button" value="Rent It"/>

## Add Movies to Cart

The screenshot shows the 'Add Movies to Cart' page. The browser's address bar shows the URL: `localhost:8080/VideoLibraryManagementSystemClient/View/homescreen`. The page layout is similar to the previous one, but the main content area is titled 'Cart'.

**Search Sidebar:**

Search Movie By:   
 Keyword:

**Categories:**

- Action
- Comedy
- Drama
- Animation
- Documentary
- Family
- Sci-Fi
- Horror
- Romance

**Featured Movies:**

Three movie posters are displayed: 'The Secret of NIMH', 'The Hulk', and 'New Moon'.

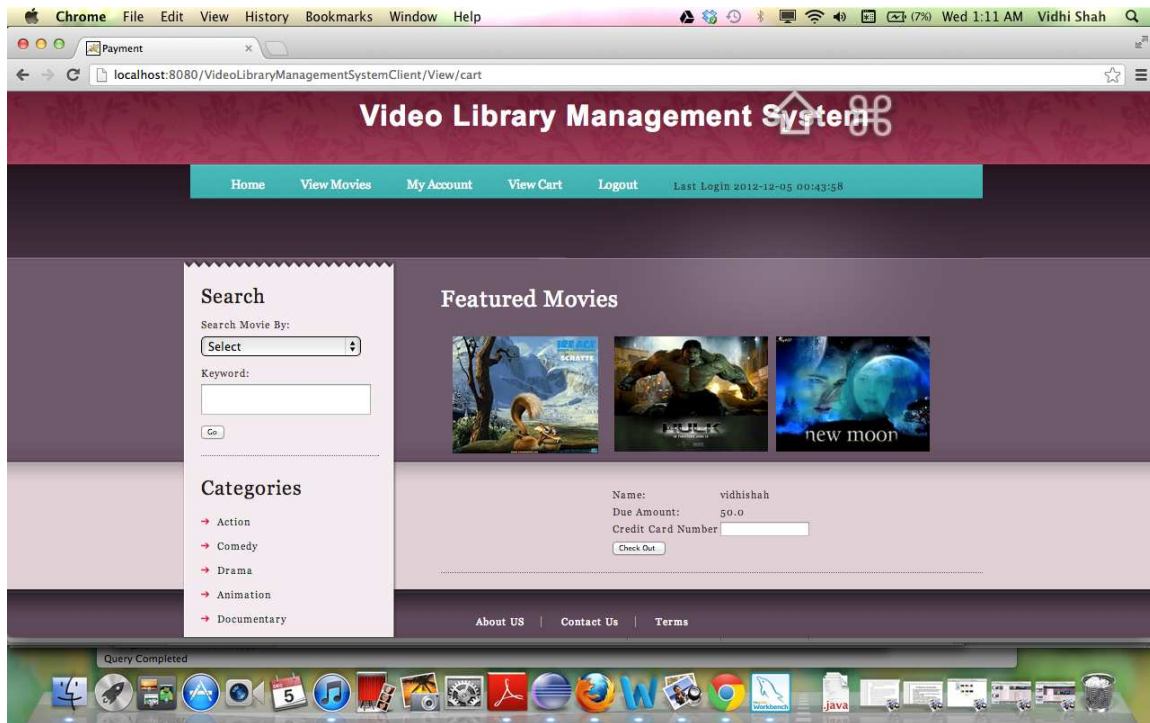
**Cart:**

Number	Movie Name	Rent(\$)	Remove
1	movie4	11.0	<input type="button" value="Remove"/>
2	movie5	12.0	<input type="button" value="Remove"/>
3	movie6	15.0	<input type="button" value="Remove"/>
4	movie5	12.0	<input type="button" value="Remove"/>

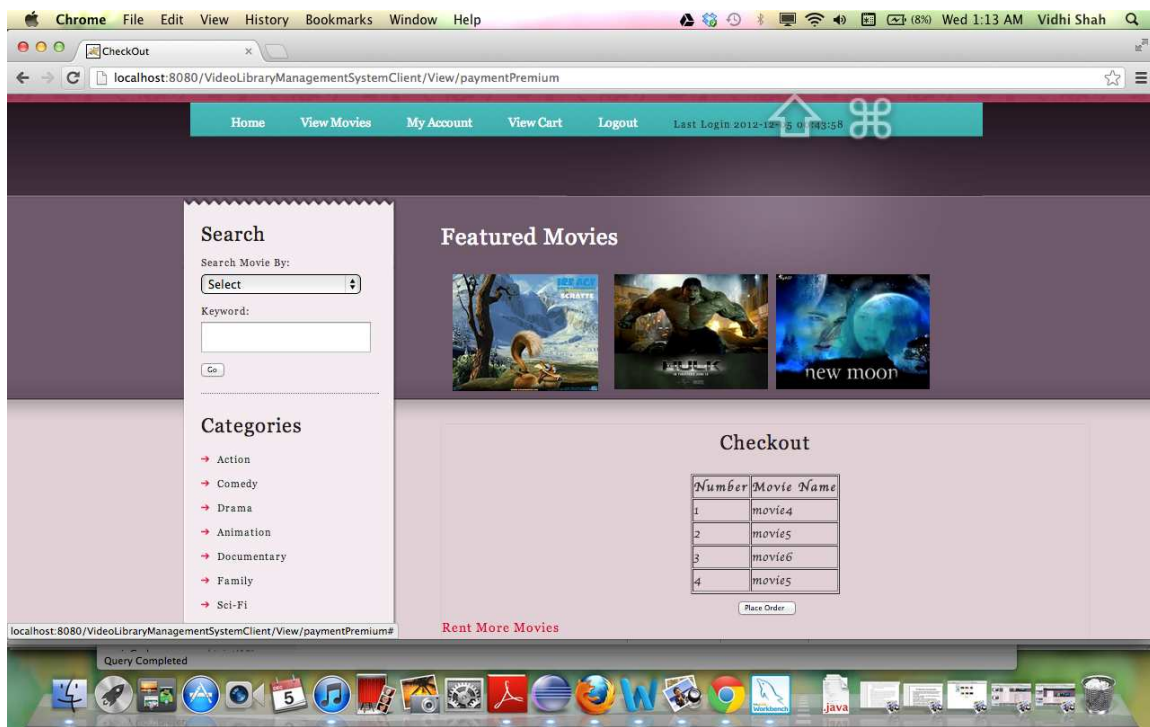
[Rent More Movies](#)



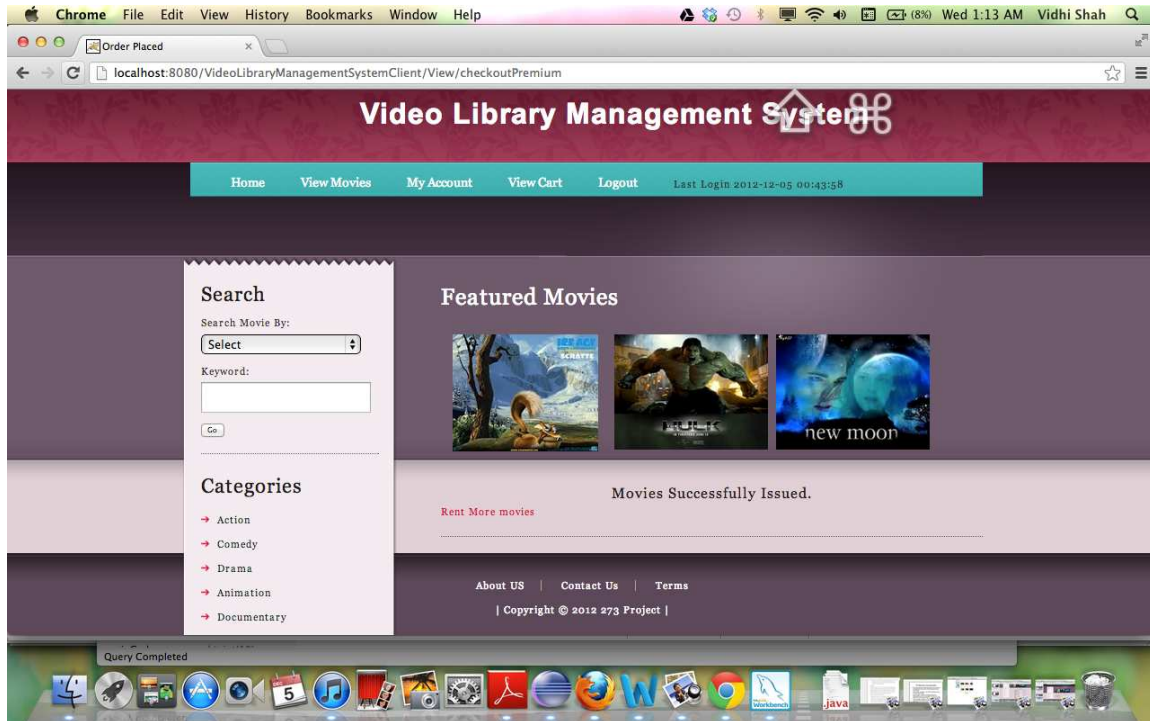
## Pay the subscription fee



## Place Order



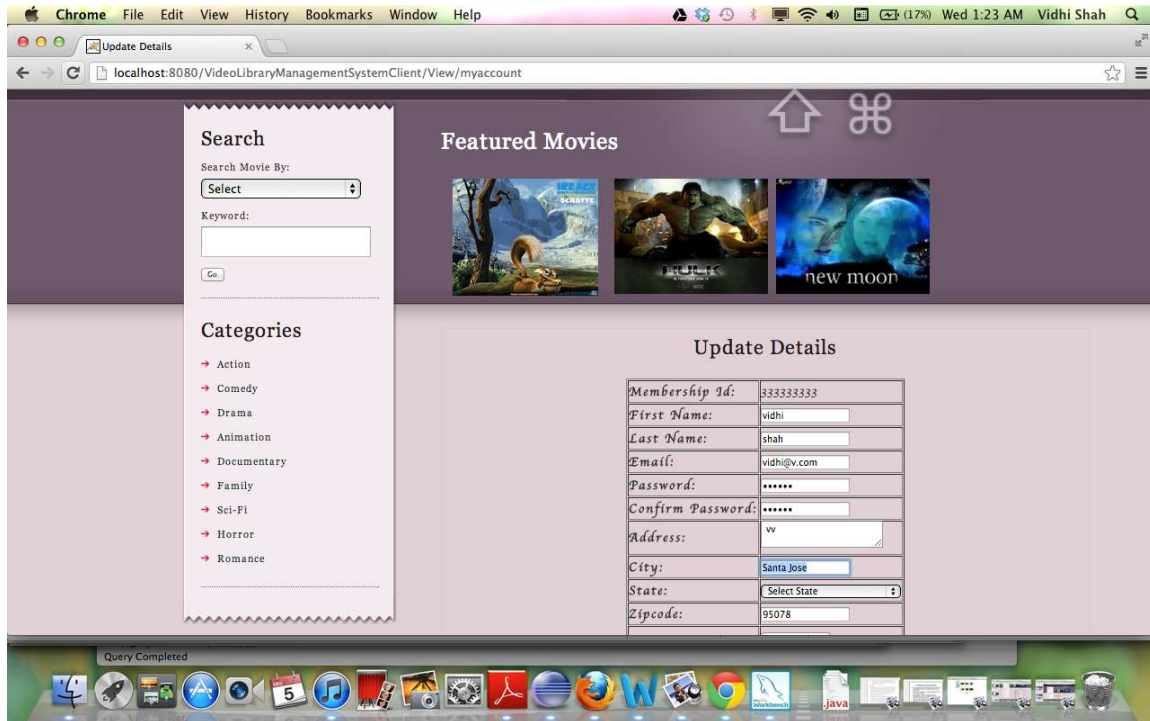
## Movies Successfully Issued



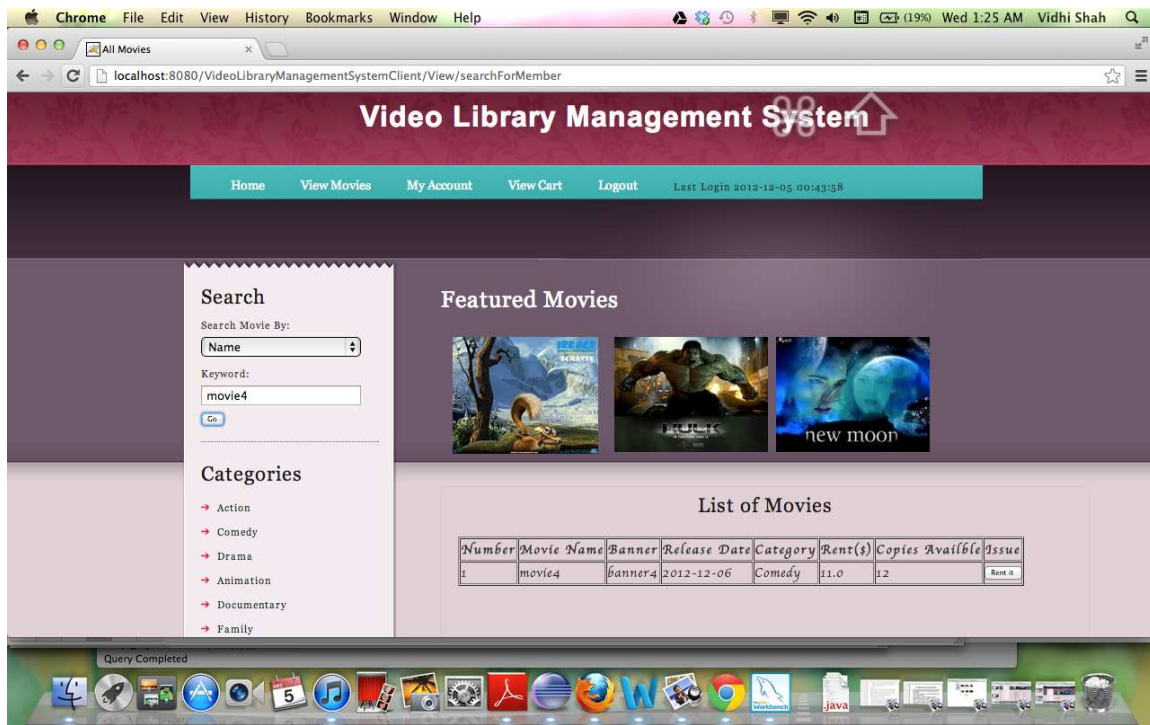
## View Account with list of issued books yet to be returned



## Update Personal Information



## Search movie by any name





## View movies by category 'Action'

The screenshot shows the 'Video Library Management System' interface. The search results for the 'Action' category are displayed in a table under the heading 'List of Movies'.

Number	Movie Name	Banner	Release Date	Category	Rent(\$)	Copies Available	Issue
1	movie1	banner1	2012-12-10	Action	15.0	6	Rent it
2	movie5	banners	2012-12-03	Action	12.0	4	Rent it
3	movie6	banners	2012-12-03	Action	6.0	6	Rent it

The interface also includes a search bar, a list of categories (Action, Comedy, Drama, Animation, Documentary), and featured movie banners.

## Cart (Simple Member)

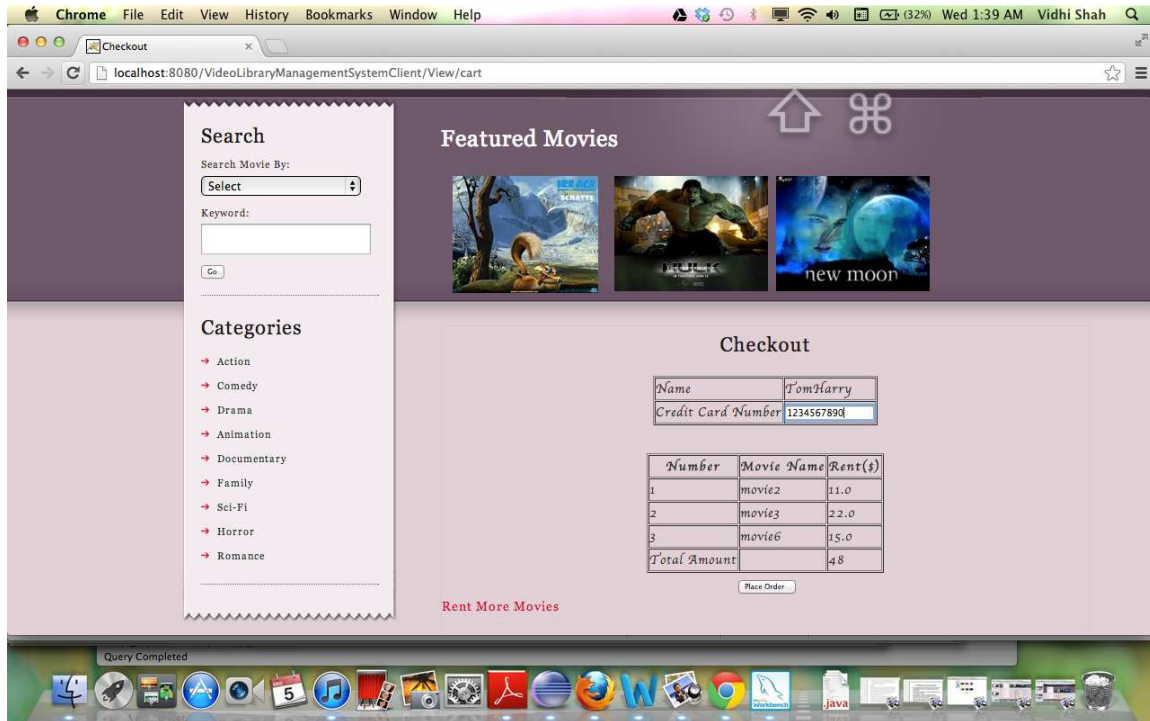
The screenshot shows the 'Video Library Management System' interface with the user's cart displayed. The cart contains three items, each with a 'Remove' button.

Number	Movie Name	Rent(\$)	Remove
1	movie2	11.0	Remove
2	movie3	22.0	Remove
3	movie6	15.0	Remove

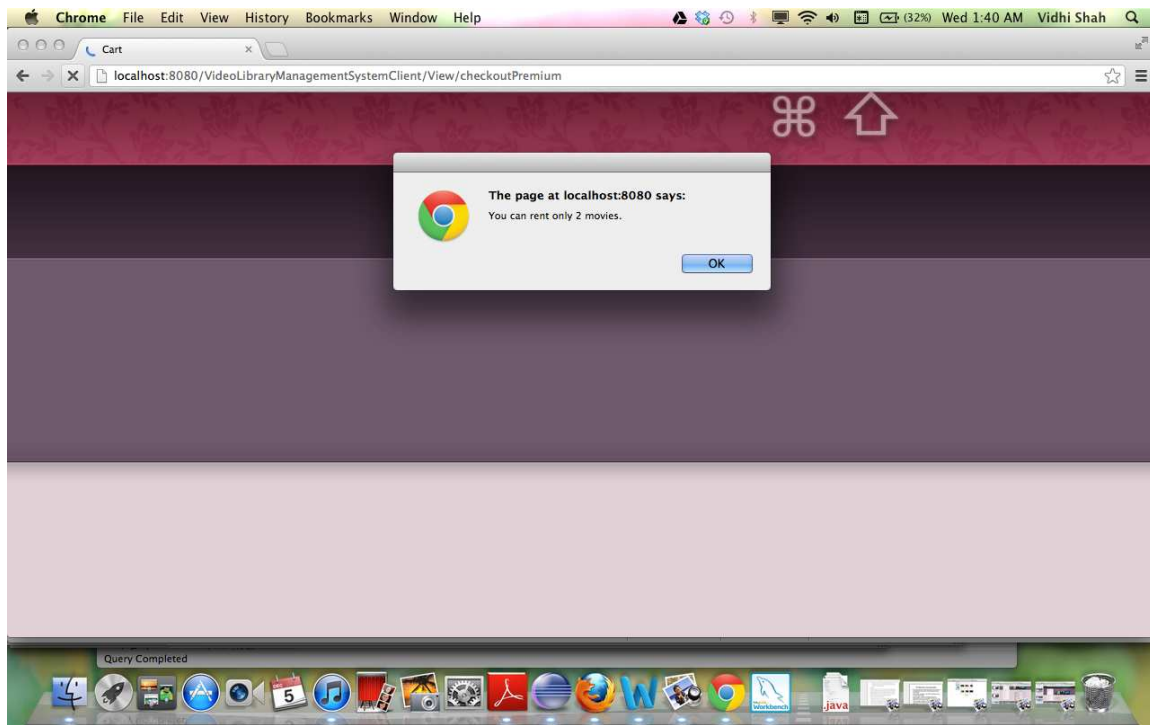
Below the cart table is a 'Checkout' button. The interface also includes a search bar, a list of categories (Action, Comedy, Drama, Animation, Documentary, Family, Sci-Fi, Horror, Romance), and featured movie banners.



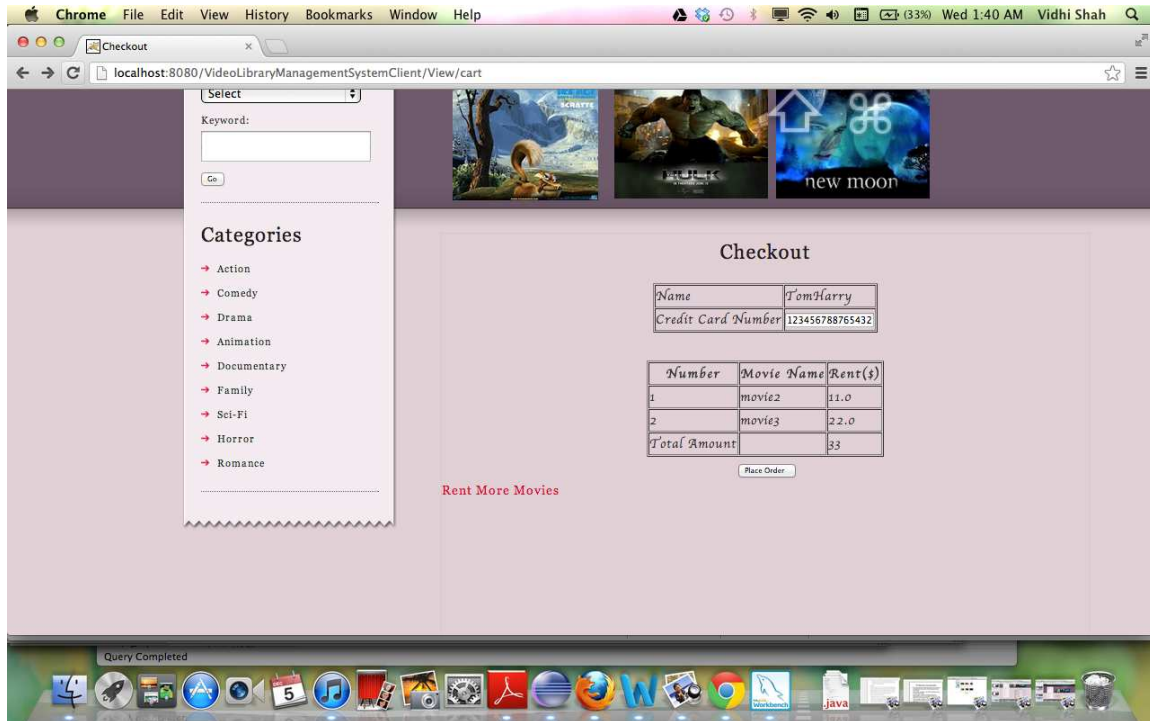
## Trying to place order



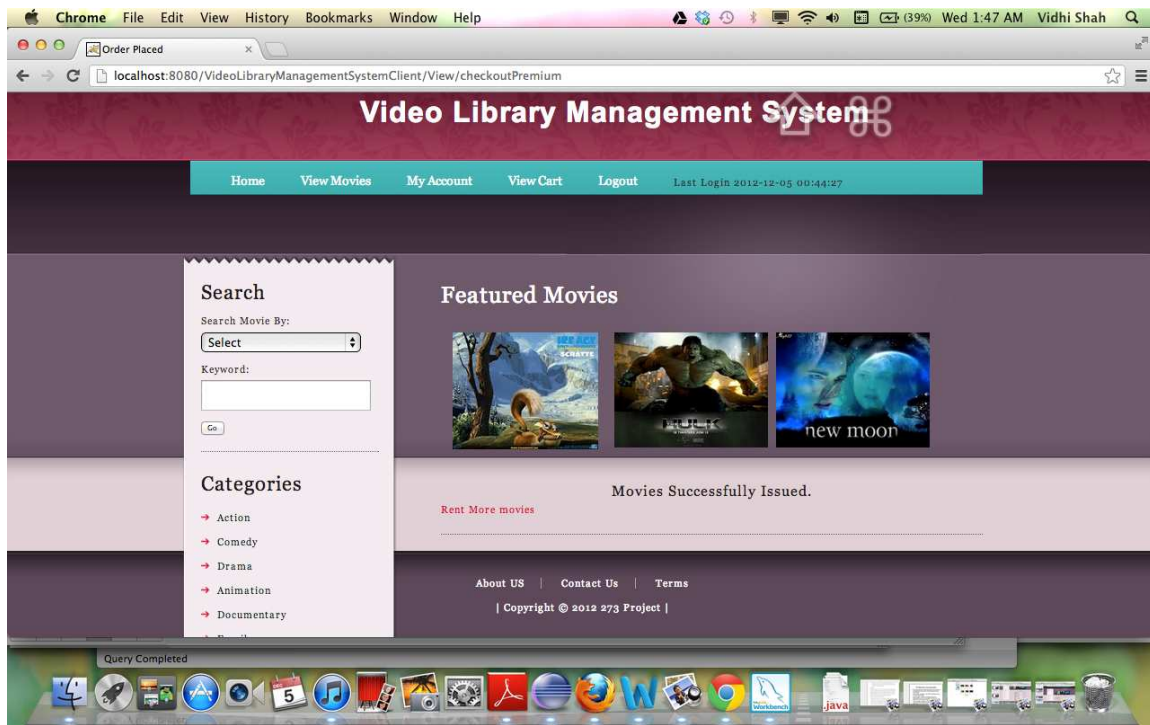
## Simple Members can issue only 2 books



## Pay Rent and checkout



## Movies successfully issued

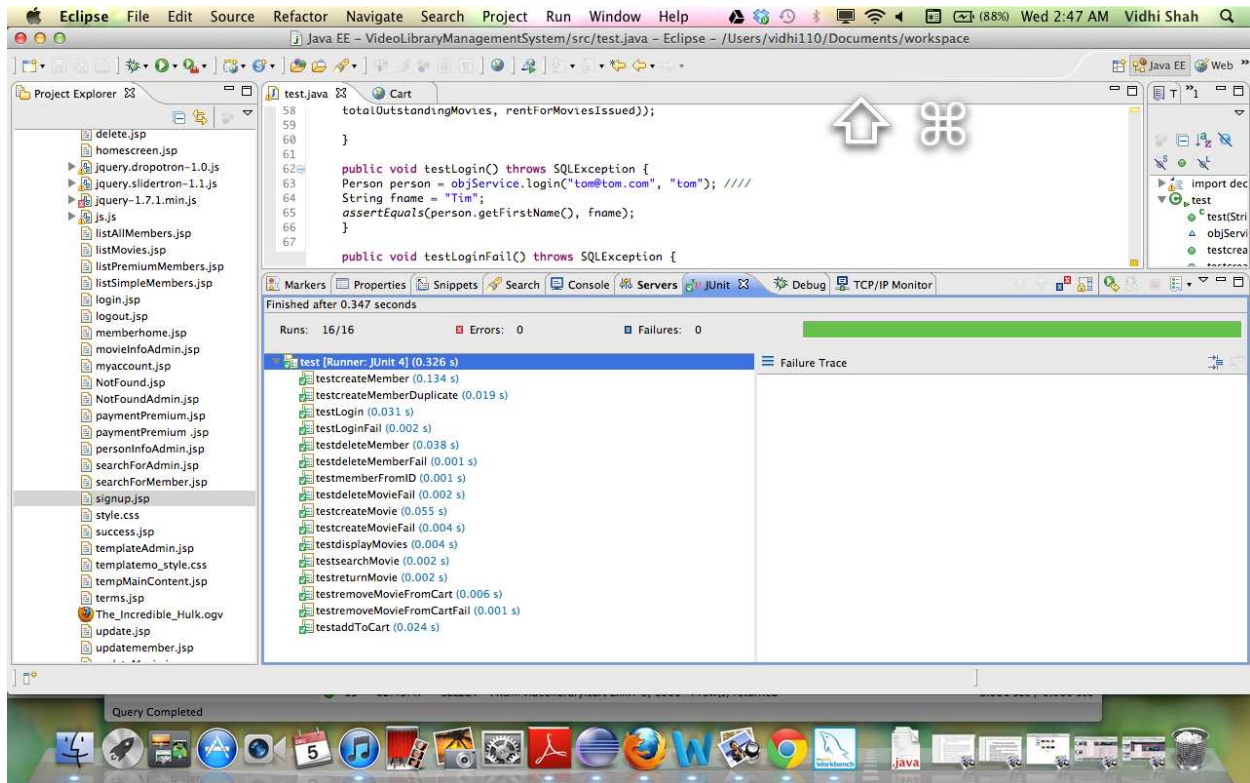


## Software Testing

Software Testing is a very important and integral part of software development cycle. By doing testing, we can be confident of our project in terms of quality. A product needs to be tested to find missing or wrong functionality as well as to improve performance of software using few testing techniques like performance testing.

### 1) Unit testing with Junit:

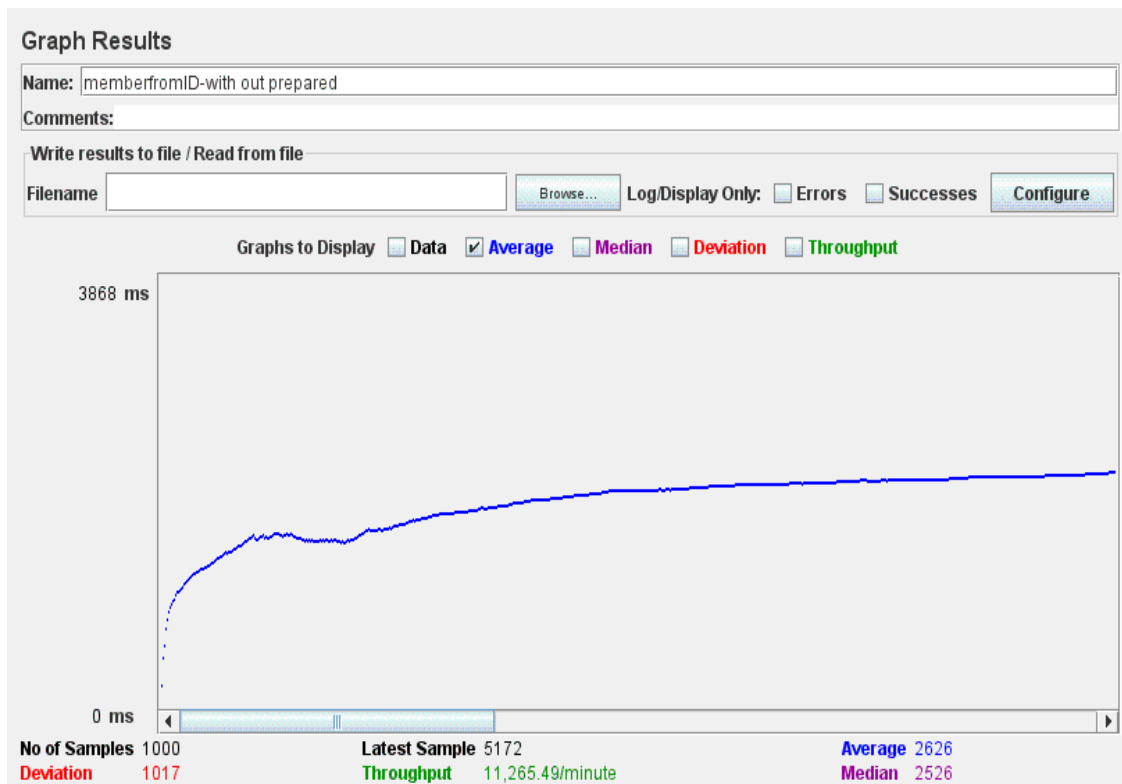
In our project, we have used Junit for unit testing. When we are debugging our software, we are trying to know what went incorrect but with unit testing, we set or confirm our expectations or correct result that should be reflected at the end of successful execution. When it fails, you can track where something went wrong and correct it. Also one Junit code created, it can be used every time to check validity of your code. So base line is Junit is used to validate code against correct result under different kind of inputs and find issues to fix. We have focused on unit testing and below are an output of the test cases:



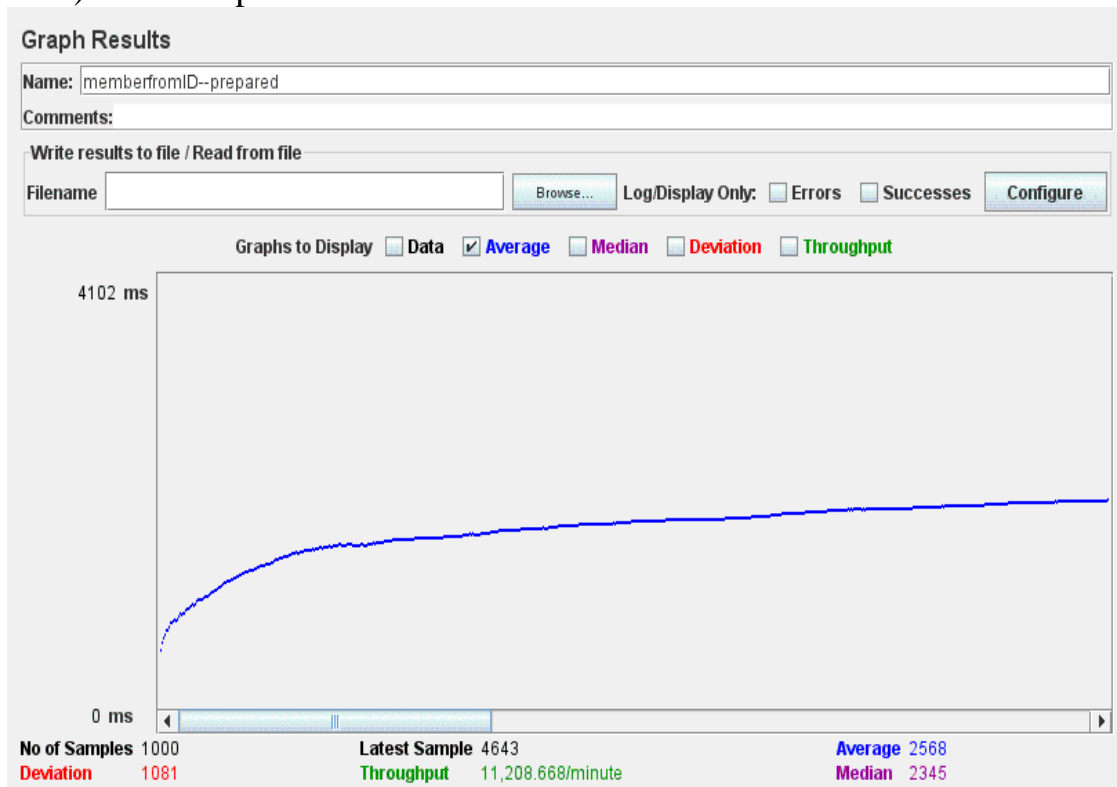
## 2) Performance testing with JMeter:

Performance testing is a part of [software engineering](#) and it is used to determine how a system responds in case of heavy load in terms of stability and responsiveness. It is also used to measure scalability and reliability of software. In our project, we have used JMeter for performance measurement. Apache JMeter is a load testing tool and can be used for measuring performance of different services mainly web applications. It is pure java desktop application. To simulate load on a server, JMeter can be used very efficiently. Also we can find use JMeter to validate that our application is returning expected results by creating test scripts with assertion. We have used JMeter to find performance of web application at different stages and results are as below:

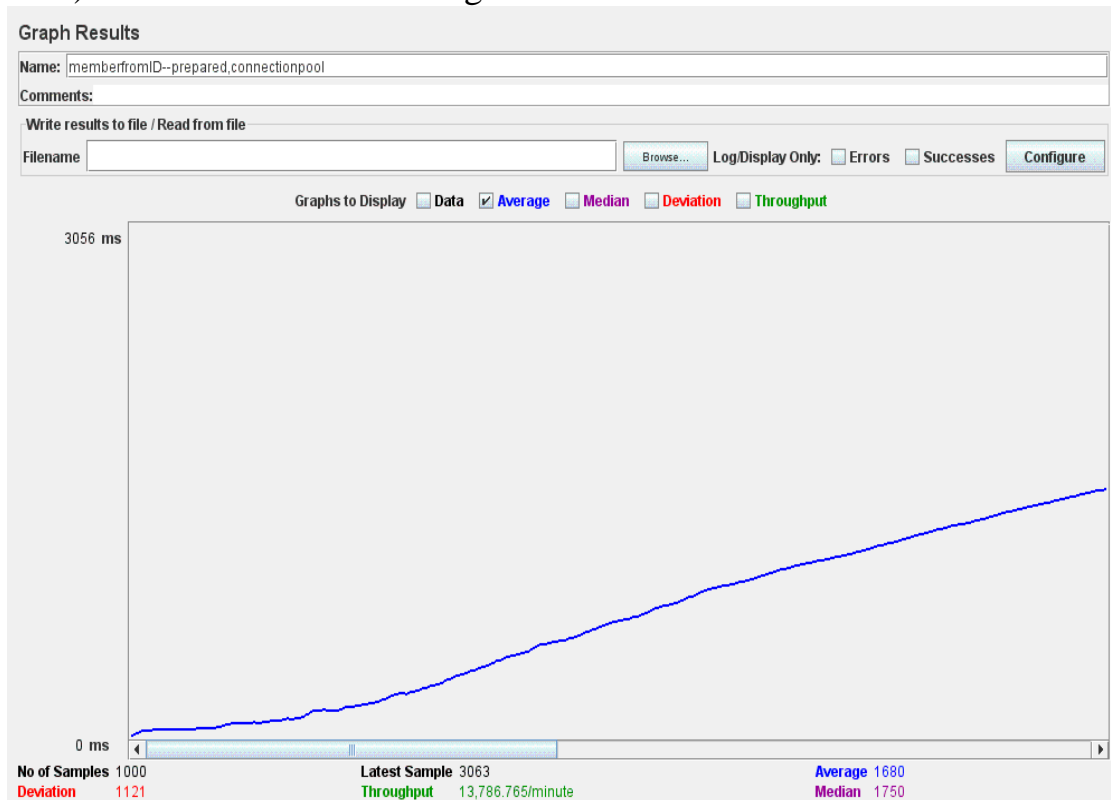
### 1) Base Class



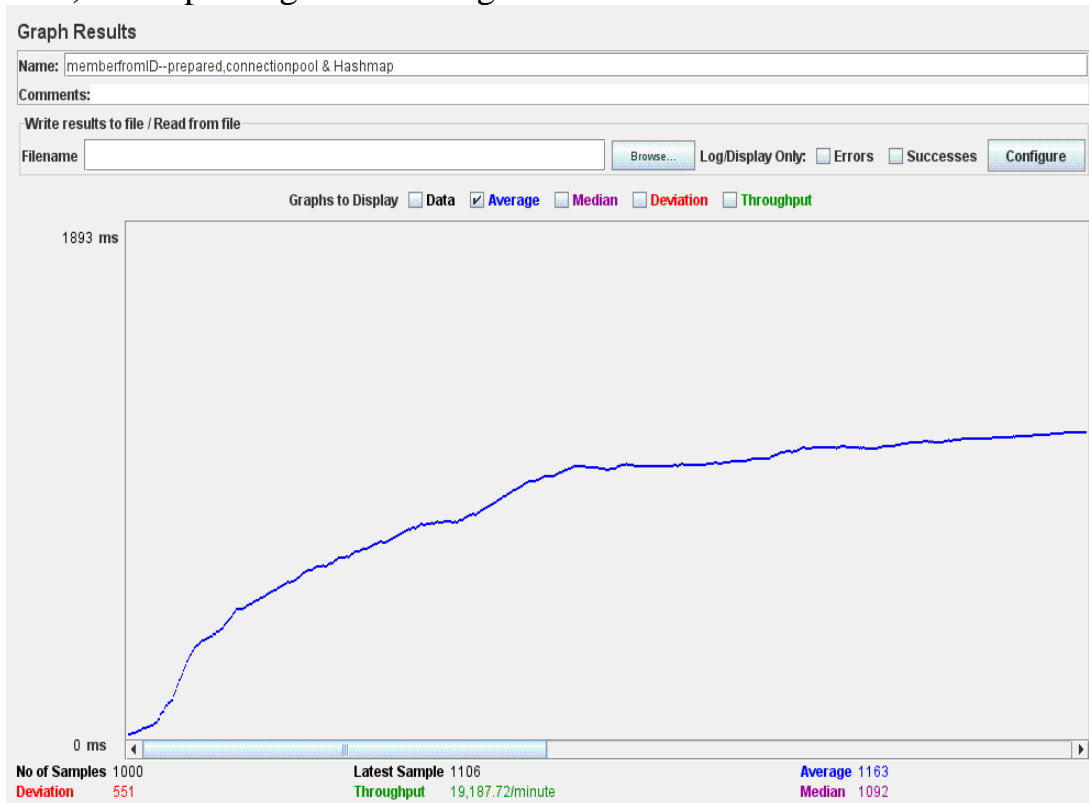
## 2) With Prepared Statement:



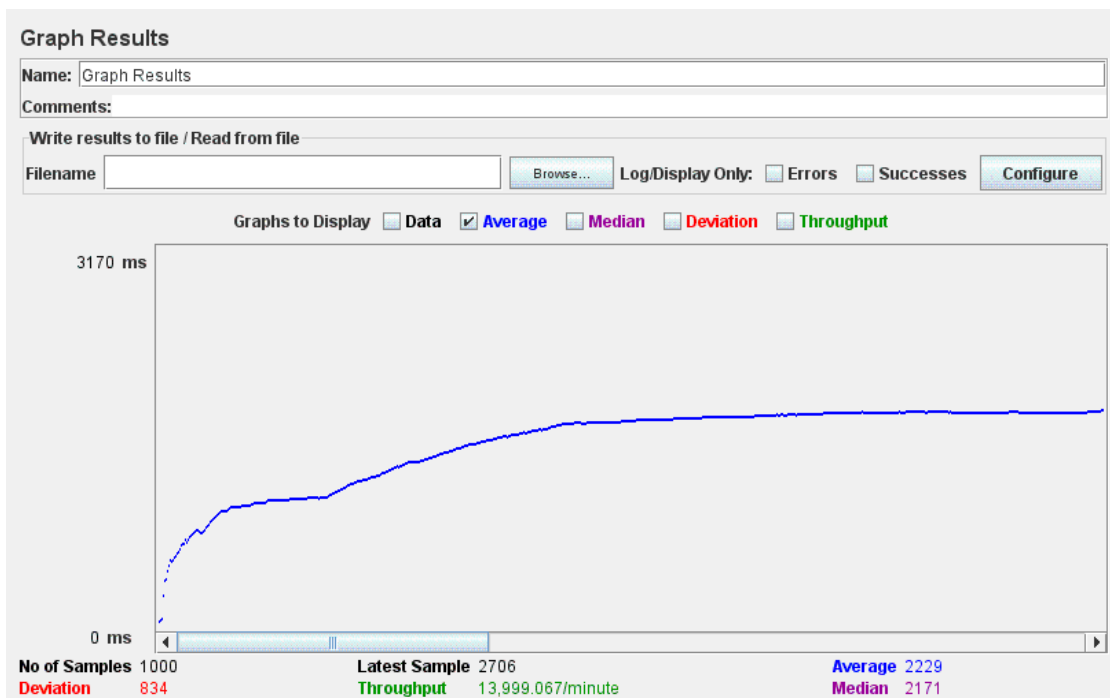
## 3) With Connection Pooling



#### 4) With pooling and Caching



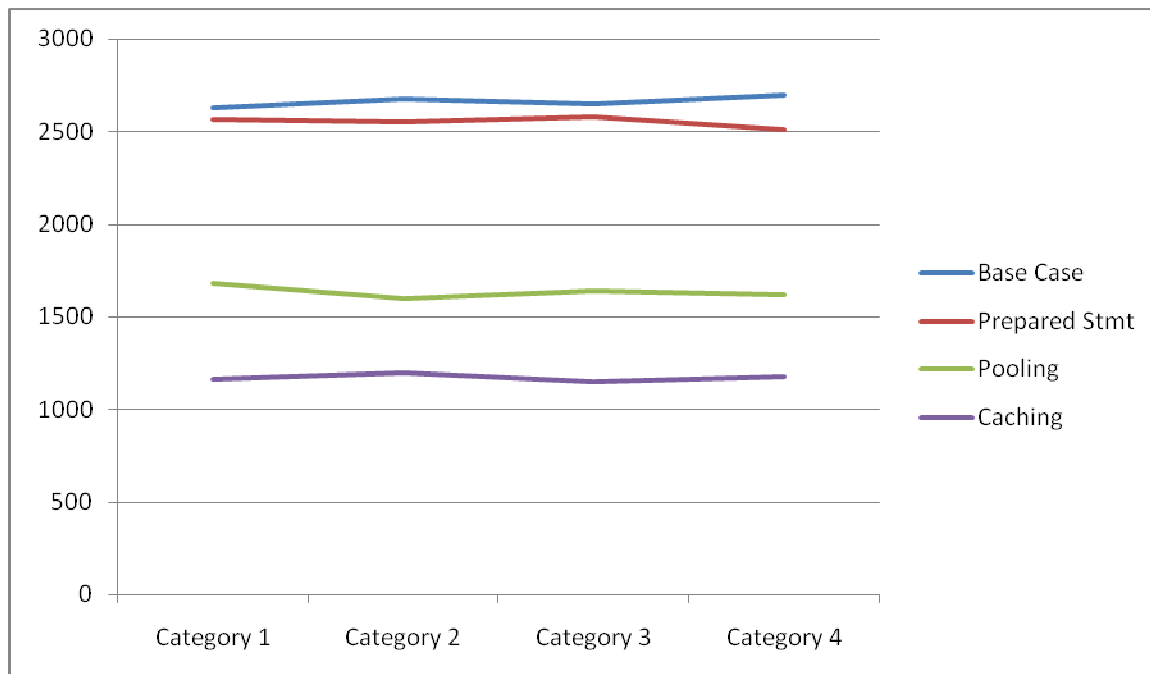
#### 5) Display Movies without pooling



## 6) Display Movies with Pooling



## 7) Final Graph:





## Extra

### Deployed on Google App Engine:

We have studied google app and tried to move your video library management system to google cloud. We have installed Google App Engine SDK and WST and tried to deploy whole project. We got some problems in deploying web services as Google App Engine did not support web services directly. As an alternate solution, you have to use wsgen tool (find under \$jdk/bin/wsgen.exe) so that wsdl file is created on server side. On client side we have to use wsimport tool (find under \$jdk/bin/wsimport.exe) to extract the wsdl file on client side and use it. But while using it on client side we had some problems. Anyway, we deployed some .jsp pages (login, about, terms, contact us) to our below mentioned site:

<http://videosystem273.appspot.com>

## Conclusion

Our implemented Video Library Management System provides functionality like creating members and movies, manage account and movie details and maintain all renting movie related transactions. We also take care about system's scalability and reliability by increasing its performance with the help of few techniques like Connection Pooling, InnoDB, use of Prepared Statement, Caching and also other database tuning techniques. And using these, we have managed to deal with heavy resources. Right now, our Video Library Management System is capable to maintain large database like 10000 simple customers, premium customers and movies in database without any kind of issues.

## Observations and lessons learned

Web service is a core and very crucial in the middle tier architecture. Connection pooling boosts the performance and mainly helps to minimize resource load. Servlets and Jsp pages with CSS give web site eye catchy look. InnoDB is also very useful in case of accessing database very frequently and also very useful in data integrity. Junit helps us to resolve minor and can say hidden issues which were not possible to find with manual testing. To learn JMeter and run our application on it was the best experience. It really helps to observe the improvement of performance after applying different performance enhancement techniques. Thanks to Google drive, we could share and centralized our code, share our work and report status. Due to that we face fewer issues at the integration time and also save our time. The most exiting observations were time differences with and without applying performance enhancement techniques like connection pooling and caching.

Besides this, we learned how to work in team with task distribution and with the help of WSDL, it's easy to develop client side code without knowing server side code. And due to WSDL, there were no dependency between client side and server side development which is nice practice and will be helpful in future. Also learned that always give 40% time to development and 60% time to Integration. After integration, definitely you will face issues which were totally unknown and tough to solve because change in one's code may affect many other files. So be ready to face at least 50% more issues after integration compared to development.