Load Testing of Loan Search

Project Report

In completion of MS Project in EECS

Under the guidance of Dr Bose Bella

by Jiten Pai

Oregon State University

Table of Contents

1 Introduction ... 1

1.1 Objective ... 1

Broker Modules ... 1

Lender Modules ... 2

Administrative Module ... 2

1.2 Development and Testing Philosophy ... 2

2 System Architecture and Technology ... 3

2.1 Architecture ... 3

The MVC pattern ... 3

2.2 Implementation Platform ... 6

3 Load Testing ... 7

3.1 Definitions ... 7

3.2 Performance Expectations ... 8

3.3 Establishment of a Performance Baseline ... 8

3.4 Baseline Metrics ... 9

3.5 Bottlenecks Identified & Tuning ... 20

3.6 Post Tuning Metrics ... 21

4 Future Work ... 32

5 Appendix A – References ... 33

6 Appendix B – Application Screenshots........................34

Preface

The purpose of this project is to load test, and fine tune the loan search functionality of the Broker Blueprint web application, an innovative Business-to-Business (B2B) online service aiding mortgage lenders and brokers in today's highly competitive mortgage market.

Broker Blueprint enables brokers to search for suitable mortgage loans across various participating lenders by entering their borrowers' loan parameters on a single page.

The Broker Blueprint portal also expands the exposure of lending institutions' loan programs into untapped markets and gives participating lenders an edge over competing institutions.

1 Introduction

1.1 Objective

The purpose of this project is to develop an innovative Business-to-Business (B2B) online service aiding mortgage lenders and brokers in today's highly competitive mortgage market.

The Broker Blueprint portal has three main classes of users – Brokers, Lenders (lending institutions), and application administrators. Following is the breakdown of application modules by the user class and a brief description of module functionality:

Broker Modules

1. Loan Search
   This module eases brokers' loan search efforts by letting them enter borrower information just once and presenting eligibility results from participating lending institutions.
2. Loan Status Management
   This module enables brokers to maintain non-personal loan information and status of their borrowers online. The data maintained in this module is also the source data for the broker business intelligence dashboard.
3. Dashboard
   The dashboard module is the business intelligence presentation layer built on top of a data warehouse. Loan status data entered through the Loan Status Management page is transformed via a backend SQL based Extraction Translation and Loading (ETL) process and loaded into the data wares house.

   Besides the business intelligence charts, the dashboard also shows pipeline loan statuses for all the loans a broker has. A mortgage loan typically takes 4-8 weeks from submission to funding. During this time, the loan is said to be in the pipeline, and access to pipeline loan status information is a key supporting feature for broker users. The information entered and maintained in the Loan Status Management area drives the pipeline status charts.
4. Website Customization
   The Broker Blueprint portal enables brokers to choose a website template from a set of available templates for customization and hosting. The standard templates are stored in XSL format to which a transformation is applied with the customized data in real-time for building the target site.
5. Locator Service
   This is a Java Messaging System based supporting service run at the application layer which is transparent to users. Any new broker or lender address information inserted into the system or updates to existing address information in the system triggers an asynchronous service call by this service to the Yahoo! geocoder service with the address data. The Yahoo! geocoder service returns the validity of the address along with the latitude and longitude information for the address. The address coordinates are stored in the database and are used to calculate the

location of the nearest branch relative to the broker's office location for each lending institution in the loan search results.

6. Online Registration and Payment
   This module enables brokers to customize the service by picking the packages they are interested in subscribing to, and register themselves via online credit card payment. Addition of subscription packages is also handled by this module. Credit card validation and payments are handled by this module by making an SSL call to the Verisign Payflow service.

Lender Modules

7. Branch Management
   This module enables lending institutions to manage their branch information like location, contact information and address. The Locator Service uses this address data to retrieve and store location coordinates.

8. Loan Program Management
   Lending institutions use this module to create and update their loan program guideline, pricing, and broker commission data. This data is used during loan search by brokers to display qualifying loan programs for their search criteria.

Administrative Module

9. Online User Administration
   The Online User Administration module enables users with administrative access to add lender or broker accounts to the system. Broker accounts can be given access to a combination of available packages, and creation of broker accounts via this module bypasses subscription payment requirement.

With online tools like Loan Search and Loan Status Management, and status tools like Pipeline Management and Dashboard, brokers can more efficiently service their customers and identify key market segments with opportunities for growth based on historical trends.

1.2 Development and Testing Philosophy

With a philosophy of 'what you deliver', not 'what you plan to deliver', the Broker Blueprint project achieved whiteboard-to-production implementation within four months of inception. The Agile methodology helped continuously deliver quality user requirements against a backdrop of evolving requirements against maturing market conditions. Tight integration of systems, tools and business processes, and component-based techniques helped in becoming more responsive to new requirements.

2 System Architecture and Technology

2.1 Architecture

The Broker Blueprint portal is developed on a J2EE platform and implements the MVC architecture pattern by utilizing the Struts 1.1 framework to separate the Model, the View and, the Controller.

This architecture pattern was chosen for the ease of maintenance and the separation of business logic of the application from the presentation layer (user interface), and the data model; any changes to one component may raise the need to make changes to code in the other components; however, since there is code separation, ease of making changes is higher, and chances of inadvertent introduction of bugs is much lower.

The MVC pattern

Model-view-controller (MVC) is a software architecture that separates an application's data model, user interface, and control logic into three distinct components so that modifications to one component can be made with minimal impact to the others.

In broad terms, constructing an application using an MVC architecture involves defining three classes of modules.

- **Model:** The domain-specific representation of the information on which the application operates. The model is another name for the domain layer. Domain logic adds meaning to raw data (e.g. calculating the total monthly subscription charges for a broker account based on the packages chosen, calculation of the closest lender’s branch relative to a broker office location).
- **View:** Renders the model into a form suitable for interaction, typically a user interface element. MVC is often seen in web applications, where the view is the HTML page and the code which gathers dynamic data for the page.
- **Controller:** Responds to events, typically user actions, and invokes changes on the model and perhaps the view.

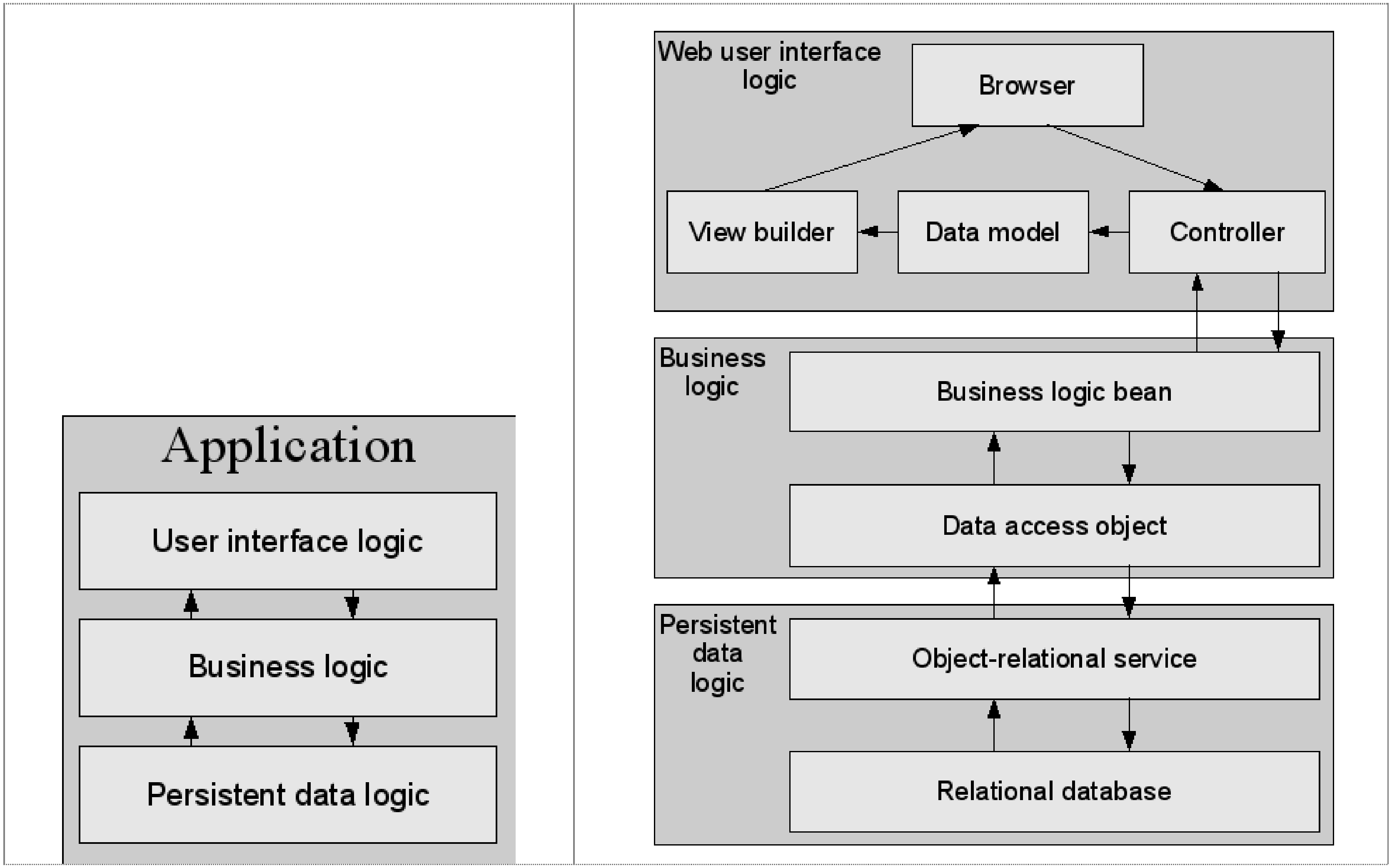


Figure 2.1.1 – System Architecture

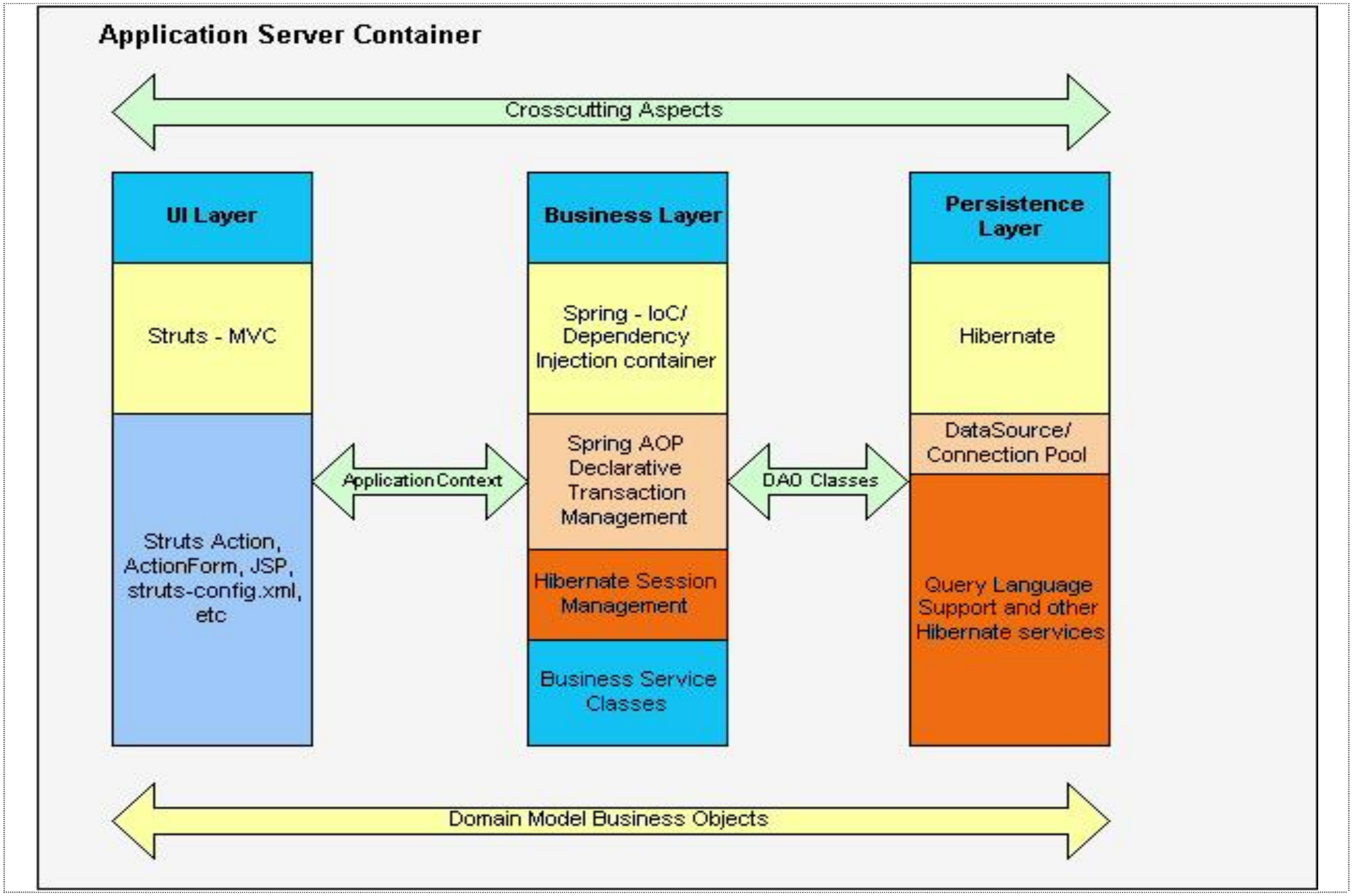

Application Server Container
Crosscutting Aspects
UI Layer
Struts - MVC
Struts Action, ActionForm, JSP, struts-config.xml, etc
Application Context
Business Layer
Spring - IoC/ Dependency Injection container
Spring AOP Declarative Transaction Management
Hibernate Session Management
Business Service Classes
DAO Classes
Persistence Layer
Hibernate
DataSource/ Connection Pool
Query Language Support and other Hibernate services
Domain Model Business Objects

Figure 2.1.2 – MVC Model

2.2 Implementation Platform

Java Platform, Enterprise Edition (Java EE) is the industry standard for developing portable, robust, scalable and secure server-side Java applications. Building on the solid foundation of Java SE, Java EE provides web services, component model, management, and communications APIs that make it the industry standard for implementing enterprise class service-oriented architecture (SOA) and Web 2.0 applications.

Some of the reasons why the J2EE platform was chosen over competing technologies are below:

1. **Choice of OS and Application Server:** Implementation of J2EE applications can be done on proven and reliable application servers like JBoss, which do not require license fees. In addition, these application servers can be run on a choice of free operating systems like Linux or Solaris.

2. **Integrated Development Environments:** Development environments like Eclipse or NetBeans are freely available to support development of J2EE applications.

3. **Connectors:** The Java Connector Architecture (JCA) is a standard for connecting to Enterprise Information Systems (EIS). JCA supports access to SAP, IBM CICS, PeopleSoft, Oracle, Siebel, Screens 3270, AS/400, Unisys, ADABAS-C, VSAM, Codasyl, IMS, Tuxedo, etc. Such a level of connectivity options is not available with any other implementation platform.

4. **Messaging Architecture:** The Java Messaging Service (JMS) standard is supported by multiple vendors such as IBM, TibCo, Progress, SpiritSoft, Fiorina, Swift, Open3, JBoss, etc.

5. **Mature O/R mapping tools:** It is best practice in Object Oriented development to decouple business objects from their underlying relational database representation. Java has several mature and robust products that support the mapping of relational databases to objects such as TopLink, CocoBase, Hibernate, OJB, etc.

6. **Standardized Authentication and Authorization:** Java has a standard way of providing Authentication and Authorization (JSSE). The choices in .NET are limited and in fact a well-published bug in the way IE handled digital certificates required a service pack to fix.

7. **Distributed Caching:** There are multiple vendors and open source projects that provide distributed caching (i.e. SpiritCache, Coherence, Gemstone, JCS, Oracle). Such a choice is not available with competing platforms.

3 Load Testing

Load testing of an application is done after the functionality of the application has been determined to be correct and matching user expectations. Load testing of an application is primarily used to ensure:

1. The system responds in a timely way to user requests, and it’s performance degrades gracefully with increasing user load
2. The system scales to an established number of concurrent users
3. The system remains stable under varying load
4. Ensuring Service Level Agreements are met for the established number of concurrent users

The load testing for the Broker Blueprint application was carried out with OpenSTA (Open, System Testing Architecture), a mature, open source web testing architecture. The OpenSTA toolset has the capability of performing scripted HTTP and HTTPS heavy load tests with performance measurements from Win32 platforms.

OpenSTA is available for download from http://opensta.org/

3.1 Definitions

Following are definitions for some typical terms used in the load testing any application

User Base
User base of an application is the total number of users who use an application. For loan search, the user base is the count of the number of broker users who are subscribed to the service.

Concurrent Users
The number of users using an application at any given moment is the Concurrent User count. This count varies by time of the day, day of the week, day of the month, and month of the year, as influenced by working hours of brokers and the housing market conditions.

Virtual Users (VUs)
A Virtual User (VU) is a simulation of a real user by the load testing software. The test pane defines the actions to be performed by sets of virtual users during a test run.

Timers
After recording load testing scripts, timers can be inserted into the scripts to time the response times when the scripts are run. With different number of VUs, the web application being tested returns different response times, and timers help record this at run time.

Data Parameterization

Parameterization of data refers to using varying, pre-defined input data when a load testing script is run. During recording, OpenSTA records data supplied by the user during the session. To simulate a realistic scenario, the input data used by virtual users needs to vary to overcome false response times due to caching at various levels in the technology stack (the web server, the application server, and the database server).

Virtual User Batches
During a test run with OpenSTA, the number of virtual users using the application can be ramped up over time by "releasing" virtual users over the test run, so as to observe the behavior of the web application with increasing virtual users.

Performance Baseline
Performance baseline of an application is the set of performance metrics for a given hardware and software configuration established by running a set of performance tests. Once the baseline is established, software and hardware configuration is modified so that improved and metrics are achieved, which meet the SLAs of various stakeholders.

3.2 Performance Expectations

Broker Blueprint expects an initial broker user base of up to 300 users. The maximum number of concurrent users expected to use the web application, most of them running the loan search function, is 100.

The maximum acceptable response time for loan search was set to 15 seconds.

User base = 300 users
Concurrent users of application = 100
Expected Response Time for loan search of <= 15 seconds

3.3 Establishment of a Performance Baseline

For the purpose of load testing, OpenSTA scripts were run against an environment which was a mirror image of the production setup in terms of hardware and software configuration. The steps used for the establishment of a performance baseline were as follows:

1. **Plan and model transactions**
   The transactions modeled for this test consisted of opening homepage, navigation to the login page, login action, navigation to the loan search page, loan search action (iterated five times), and the logout action.

   Timers were created to measure the start and end times of each navigation and action for the VUs.

Data used by VUs for the login and loan search transactions was parameterized so as to overcome the skewing of metrics due to caching at various levels in the web application.

2. **Model VU batches**
   VU batches were modeled so that at start time, three VUs would be released over a three-second ramp up time to start performing the transactions in the sequence listed above. Thereafter, every seven seconds, a new batch of three virtual users were released with the same ramp up time. Effectively, every 10 seconds, 3 new users would be introduced into the test until all the virtual users were released.

3. **Measure metrics with various total VUs**
   Five runs were made with the above VU batch model and 30, 50, 65, 75, and 100 total VUs in each run respectively.

   With these five baselines, a good indication of system behavior could be established with increasing user load. Given the VU release model, where VUs were released into the system over time, it would be improbable to have all the VUs active at any given time, as some of the VUs would have completed their transactions by the time new ones were released. The batch of 100 VUs had a maximum of just over 60 concurrent VUs active during the test run.

4. **Plotting of metrics**
   Metrics measured during various test runs were imported into Microsoft Excel as Comma Separated Value (CSV) files and were the basis for the measurement plots.

3.4 Baseline Metrics

The following plots were created with the metrics collected using OpenSTA for different numbers of VUs performing the same operations.

Three runs were preformed for each set, and the average of these metrics is shown in the tables and used in the adjoining plots.

VUs	T_ACT_LOANSEARCH	T_ACT_LOGIN	T_ACT_LOGOUT	T_NAV_LOANSEARCH	T_NAV_LOGIN
5		7.170			9.800
6			4.810		
9				7.550	
10	11.680				
12	11.700				
14	11.650				
15					9.750
18	11.700				
20	12.070	7.020		8.130	
21			5.140		
24	12.110				
25	11.725				
27	12.210				
29	12.260				
30	12.053	7.340	4.860	8.040	13.650

Table 3.4.1 – 30 Maximum VUs, Average of three test runs

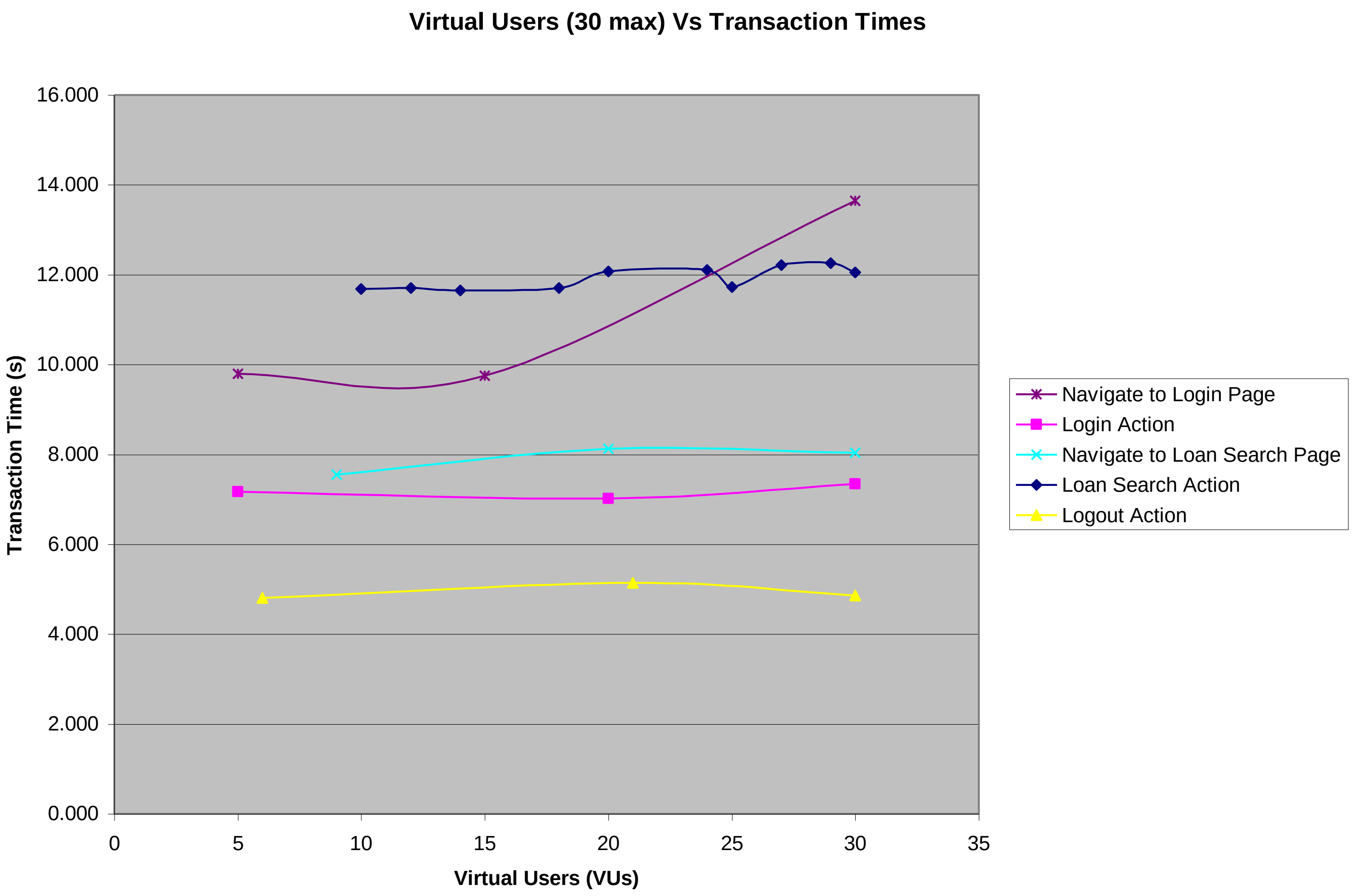


Figure 3.4.1 – 30 Maximum VUs, Average of three test runs

VUs	T_ACT_LOANSEARCH	T_ACT_LOGIN	T_ACT_LOGOUT	T_NAV_LOANSEARCH	T_NAV_LOGIN
6		6.970			9.815
9		7.020		7.480	
11			4.830		
12	11.720			7.760	9.860
15	11.700	6.980	4.800	7.480	
17	11.730		4.900		
21	11.758	7.390			9.970
23	11.947		4.840		
24	11.817	7.500		7.830	
26	11.700				
27	12.083			8.030	10.020
30	12.310	7.720			
32	11.866		5.080		
35	12.490		5.360		
36	12.837			9.330	12.040
38	13.813	12.610	7.710	9.580	
39	14.986	11.810	9.365	12.740	15.575
41	15.170	13.540	10.010	15.150	17.490

Table 3.4.2 – 50 Maximum VUs, Average of three test runs

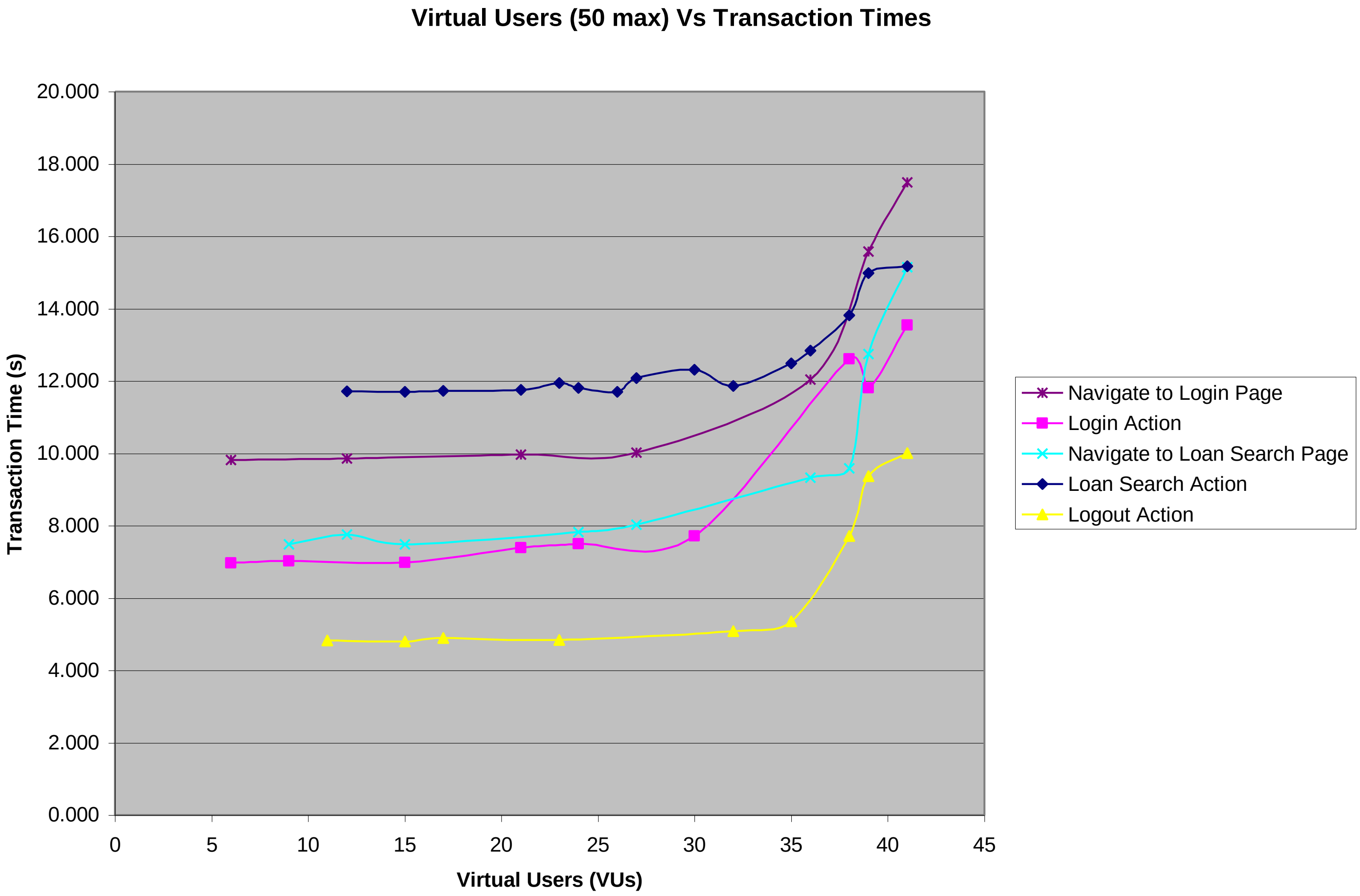


Figure 3.4.2 – 50 Maximum VUs, Average of three test runs

VUs	T_ACT_LOANSEARCH	T_ACT_LOGIN	T_ACT_LOGOUT	T_NAV_LOANSEARCH	T_NAV_LOGIN
6		6.960			10.025
9		7.020		7.530	
10			4.860		
12	11.740			7.790	10.070
15	11.710	7.030		7.550	9.800
17	11.900		5.150		
20	11.790		4.840		
21	12.230	7.360			9.930
23	11.675		4.830		
24	11.953	7.630		8.160	
27	12.080			8.070	10.840
28	11.687				
30	12.310	8.610			
31	11.838		4.960		
34	11.793		4.910		
36	13.000			9.530	14.650
38	11.840		4.950		
39	14.816	12.375	9.165	14.730	15.920
41	13.090			11.420	
43	15.773	14.515	10.325	15.520	18.070
44	16.633	16.445	11.720	18.240	19.815
47	16.710	16.490	14.300	18.990	20.210

Table 3.4.3 – 65 Maximum VUs, Average of three test runs

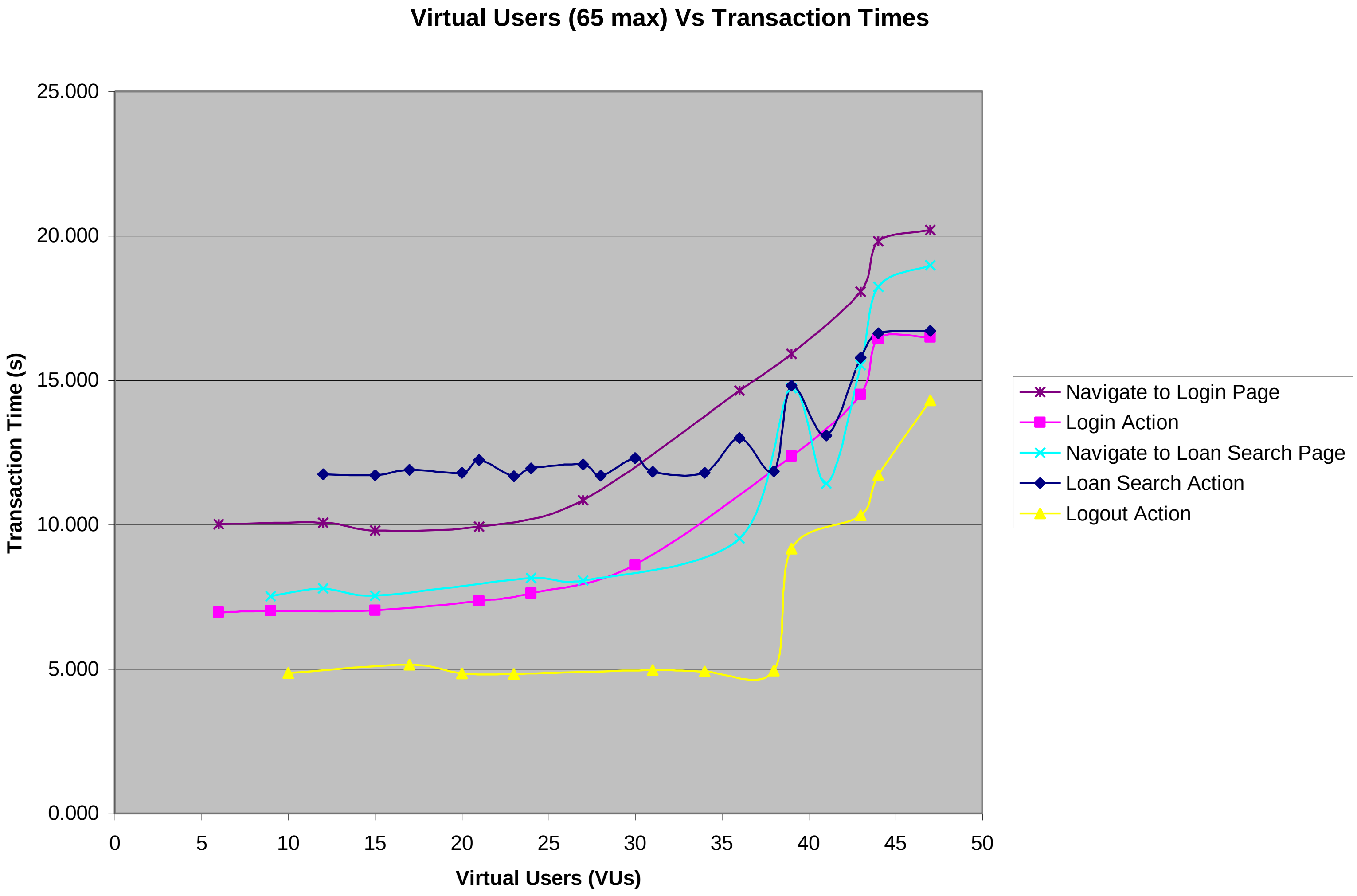


Figure 3.4.3 – 65 Maximum VUs, Average of three test runs

VUs	T_ACT_LOANSEARCH	T_ACT_LOGIN	T_ACT_LOGOUT	T_NAV_LOANSEARCH	T_NAV_LOGIN
6		7.040			10.170
9		7.010		7.830	
12	11.740			7.510	9.810
15	11.660	7.090		7.510	
17	11.680		4.810		
20	11.700		4.820		
21	11.750	7.140			9.780
24	11.747	7.140		7.590	
27	12.133			7.960	10.050
30	11.864	7.980	4.810		
34	11.773		5.090		
36	12.665			9.700	11.090
38	11.892				
39	14.677	11.450	9.430	14.290	15.900
42	14.694	15.215	9.447	16.070	18.930
43	16.770	16.180			19.820
45	13.920		10.830	14.500	
46	15.803	15.750	10.890	16.750	
47	17.380		12.460	18.990	
48	17.984	16.965	12.760	19.240	21.355
49	17.372	16.950	18.790	22.010	

Table 3.4.4 – 75 Maximum VUs, Average of three test runs

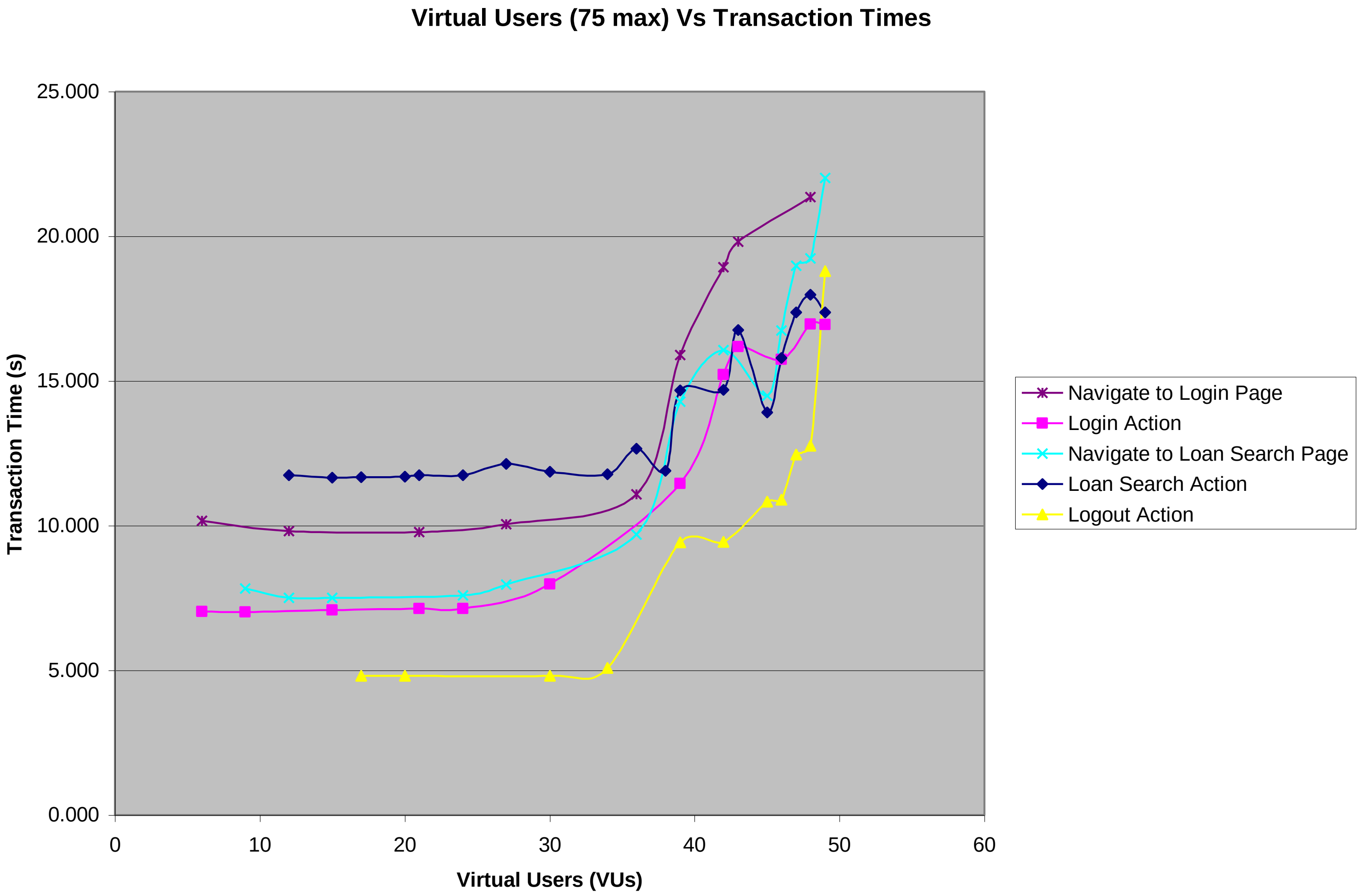


Figure 3.4.4 – 75 Maximum VUs, Average of three test runs

VUs	T_ACT_LOANSEARCH	T_ACT_LOGIN	T_ACT_LOGOUT	T_NAV_LOANSEARCH	T_NAV_LOGIN
6		7.320			9.875
9		7.050		7.550	
12	11.730			7.470	11.080
14			4.800		
15	11.740	7.290			
19	11.675		4.820		
21	11.800	7.230		7.590	9.920
24	11.853	7.150		8.020	
25	11.695		4.860		
27	12.107			7.760	10.050
29	11.730		4.865		
30	12.077	8.420			
34	11.737				
36	13.370			9.250	12.050
39	14.896	12.175	9.160	13.070	15.355
40	11.922		5.230		
42	15.890	13.440		17.280	18.970
43	16.437		11.770		18.700
44	16.800	16.035	12.520	17.700	19.920
45	13.914		6.565		
48	17.495	16.620	12.200	19.525	20.640
49	16.636	15.840	13.580	17.400	22.465
52	16.718		11.160	19.365	
54	18.872	19.095	14.530	21.460	23.450
55	18.110		13.700		
56	19.994			22.600	24.800
57	20.070	21.120	15.860		
58	19.333	21.800	14.360	23.690	
60	20.920	25.020		24.990	26.080
62	21.220			26.660	
63	21.582	24.160	17.420	26.650	28.515

Table 3.4.5 – 100 Maximum VUs, Average of three test runs

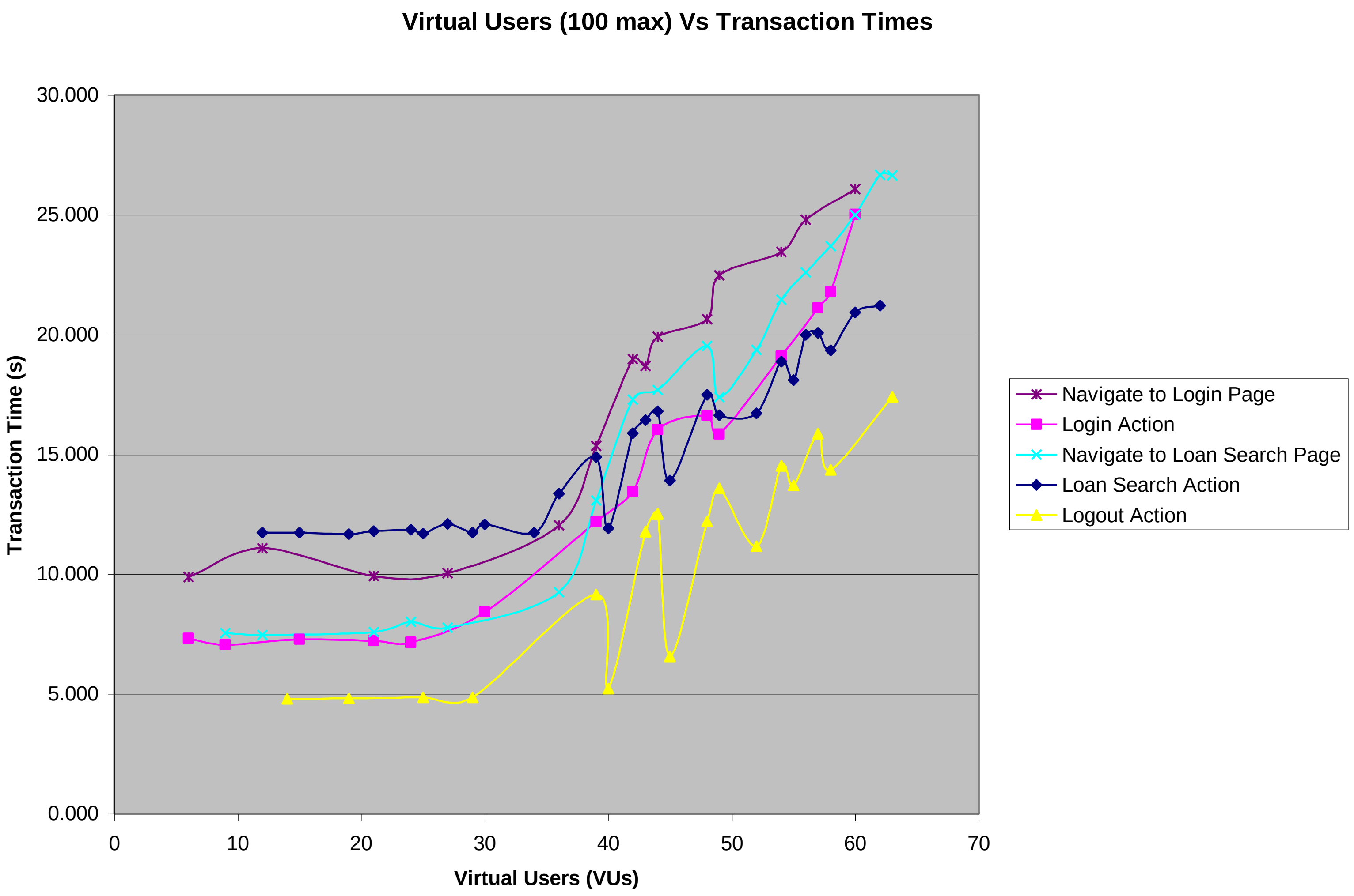


Figure 3.4.5 – 100 Maximum VUs, Average of three test runs

3.5 Bottlenecks Identified & Tuning

During various runs, application server and database server logs were monitored to ensure there were no errors. Additionally, manually analyzing these logs along with the metrics collected indicated primary bottlenecks in the following areas

Network

Latency in network communication between the application and database servers was identified as one of the potential bottlenecks in the system hampering the performance of loan search with increasing load.

The network switch between the two hardware servers was only capable of 100Mb/s, whereas the interfaces on both the servers were 1Gb/s capable. Due to the switch being slower than the interfaces, the effective network speed was only 100Mb/s.

To address this, the network switch was replaced with a 1Gb/s capable switch.

Java Virtual Machine heap memory configuration

The initial configuration for the heap memory allocated to the Java Virtual Machine (JVM) on the application server was set as follows:

Initial memory allocation (Xms parameter) = 128MB
Maximum memory allocation (Xmx parameter) = 256MB

Typically, in a production setup it is recommended to set the above two server parameters for the JVM both to the same number. This is to avoid delays due to repeated, on-demand incremental memory allocations by the JVM at run time.

To address the memory allocation issue, both the initial and maximum heap memory available to the JVM was set to 256MB.

Database index creation

Database indexes can be created to speed up certain queries against the data in the table. The `LOAN_DETAILS` table is primarily used for the loan search functionality.

By analyzing some of the typical queries used by brokers during loan search, the following indexes were created by the database administrator:

1. Index on Property Type
2. Index on Loan Purpose, and
3. Index on Documentation Type

3.6 Post Tuning Metrics

The following metrics were collected and respective charts plotted by running the same suite of five load tests after the identified bottlenecks were mitigated to a satisfactory degree.

As with the baseline metrics measurement, each test set was run three times and the average of the metrics is shown in the tables and used in the adjoining plots.

The time taken to perform loan search operation with up to 100 VUs using the system was brought down to under the 15 second acceptance mark.

It is also noticeable that the system performance degradation is not accelerated at the number of VUs on the system increased up to 100 – the performance curves are flatter. This indicates a more graceful degradation in performance than before the tuning of the system.

VUs	T_ACT_LOANSEARCH	T_ACT_LOGIN	T_ACT_LOGOUT	T_NAV_LOANSEARCH	T_NAV_LOGIN
4		6.750			9.510
6	11.480		4.460		
8				7.140	
12	11.510				
15	11.460				9.440
18	11.600	6.750			
21	11.475		4.470	7.710	
24	11.593				
27	11.635				
30	11.470	6.740	4.480	7.130	9.460

Table 3.6.1 – 30 Maximum VUs, Average of three test runs

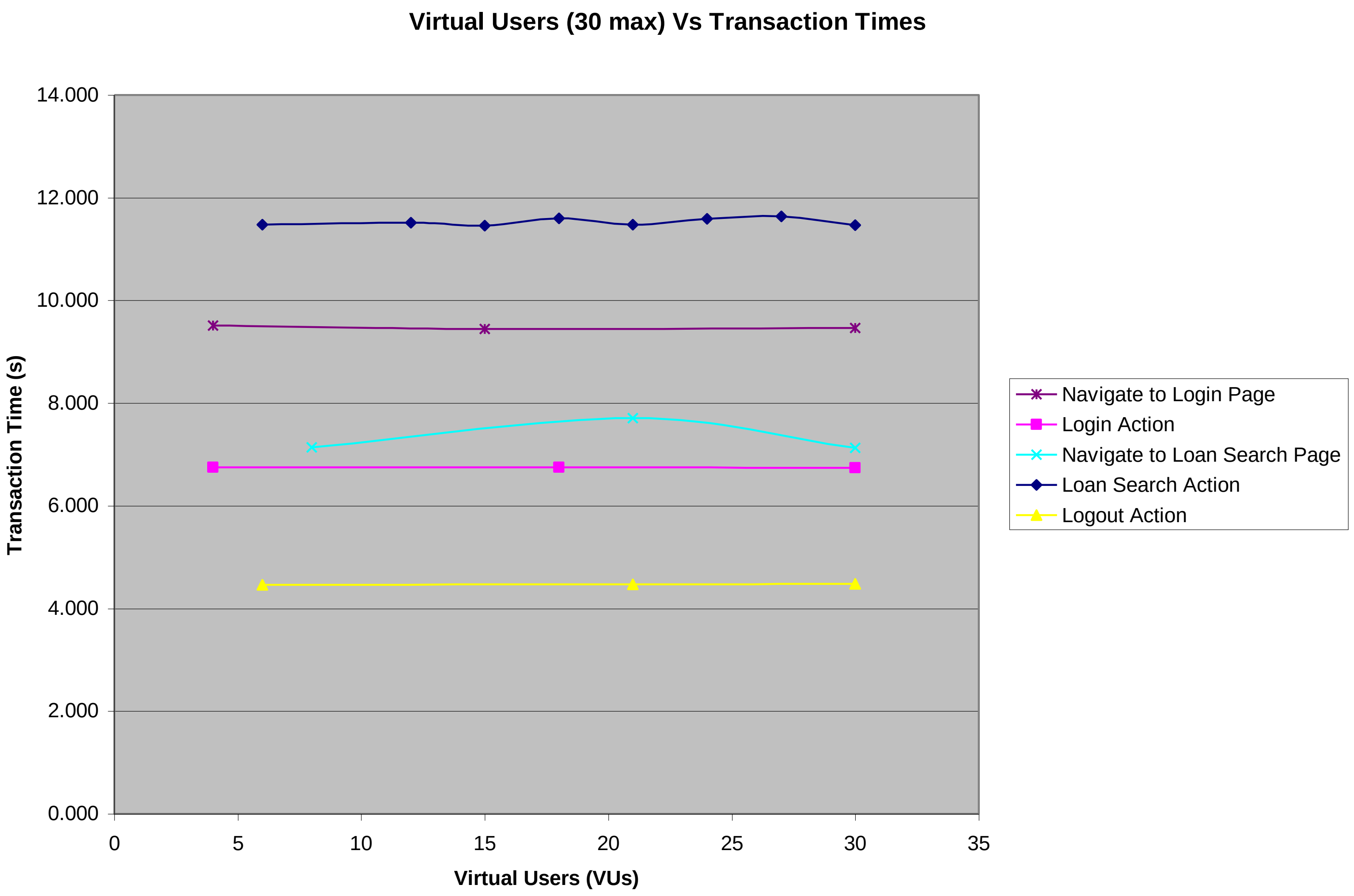


Figure 3.6.1 – 30 Maximum VUs, Average of three test runs

VUs	T_ACT_LOANSEARCH	T_ACT_LOGIN	T_ACT_LOGOUT	T_NAV_LOANSEARCH	T_NAV_LOGIN
4		6.730			9.420
5			4.510		
8	11.470			7.130	
11	11.470				
12	11.460				
15					9.420
17	11.470		4.450		
18	11.480	6.750		7.120	
21	11.583				
24	11.470				
26	11.485				
27	11.483	6.750	4.510		9.550
30	11.470		4.640	9.980	
31	11.470				
32	11.506	6.760		7.180	9.540
33	11.530	6.750	4.780	7.370	

Table 3.6.2 – 50 Maximum VUs, Average of three test runs

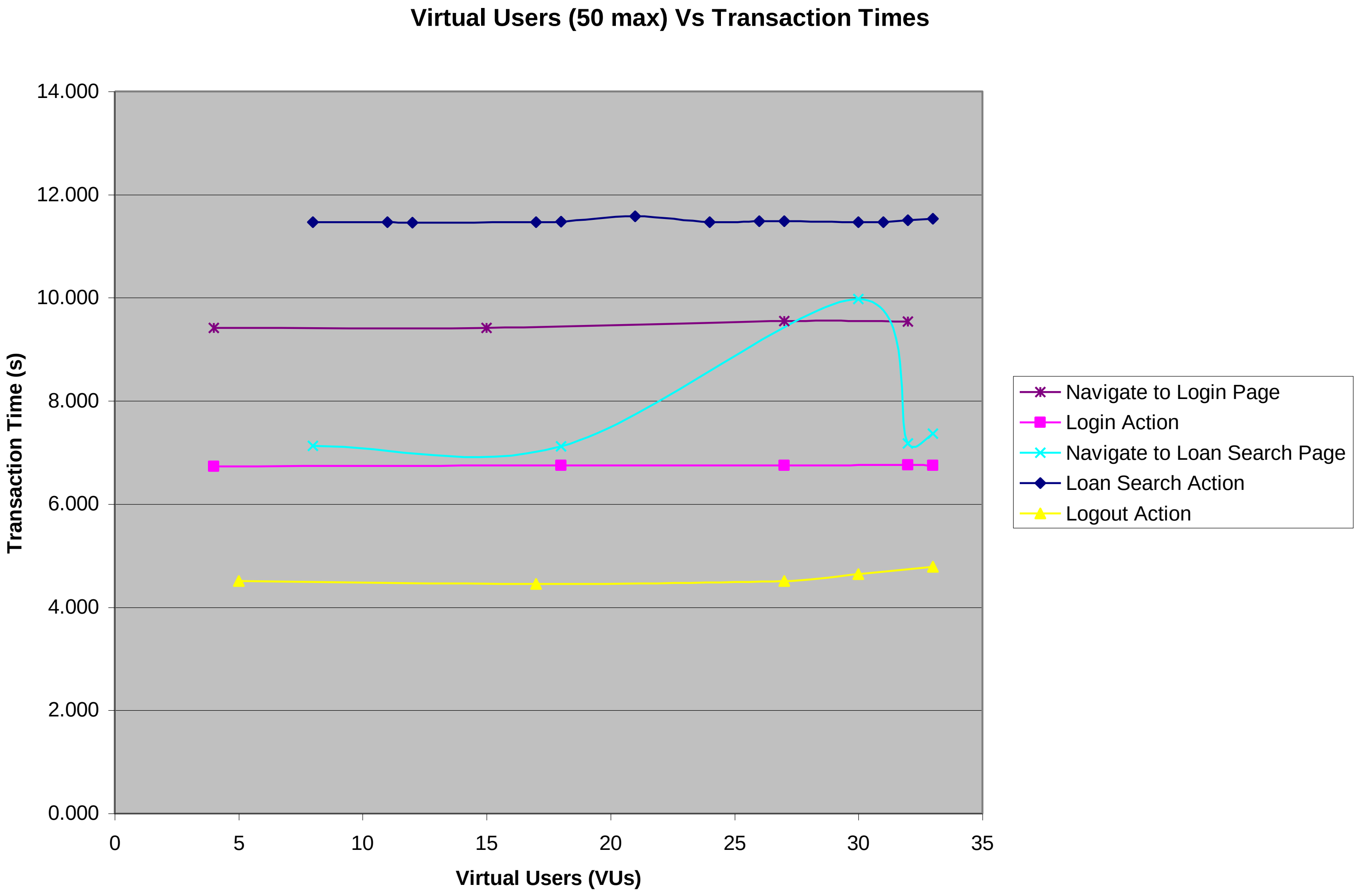


Figure 3.6.2 – 50 Maximum VUs, Average of three test runs

VUs	T_ACT_LOANSEARCH	T_ACT_LOGIN	T_ACT_LOGOUT	T_NAV_LOANSEARCH	T_NAV_LOGIN
4		6.730			9.550
8	11.470		4.460	7.150	
12	11.470				9.450
15		7.180			
16	11.470				
18	14.320			7.180	
20	11.470		4.460		
21	11.475				9.420
22	11.470				
24	11.475	6.940		7.160	
26	11.470				
30	11.482		4.470		9.420
31	11.480	6.743		7.140	
32	11.473		4.475	8.615	9.600
33	11.474	6.730	4.470	7.160	9.440

Table 3.6.3 – 65 Maximum VUs, Average of three test runs

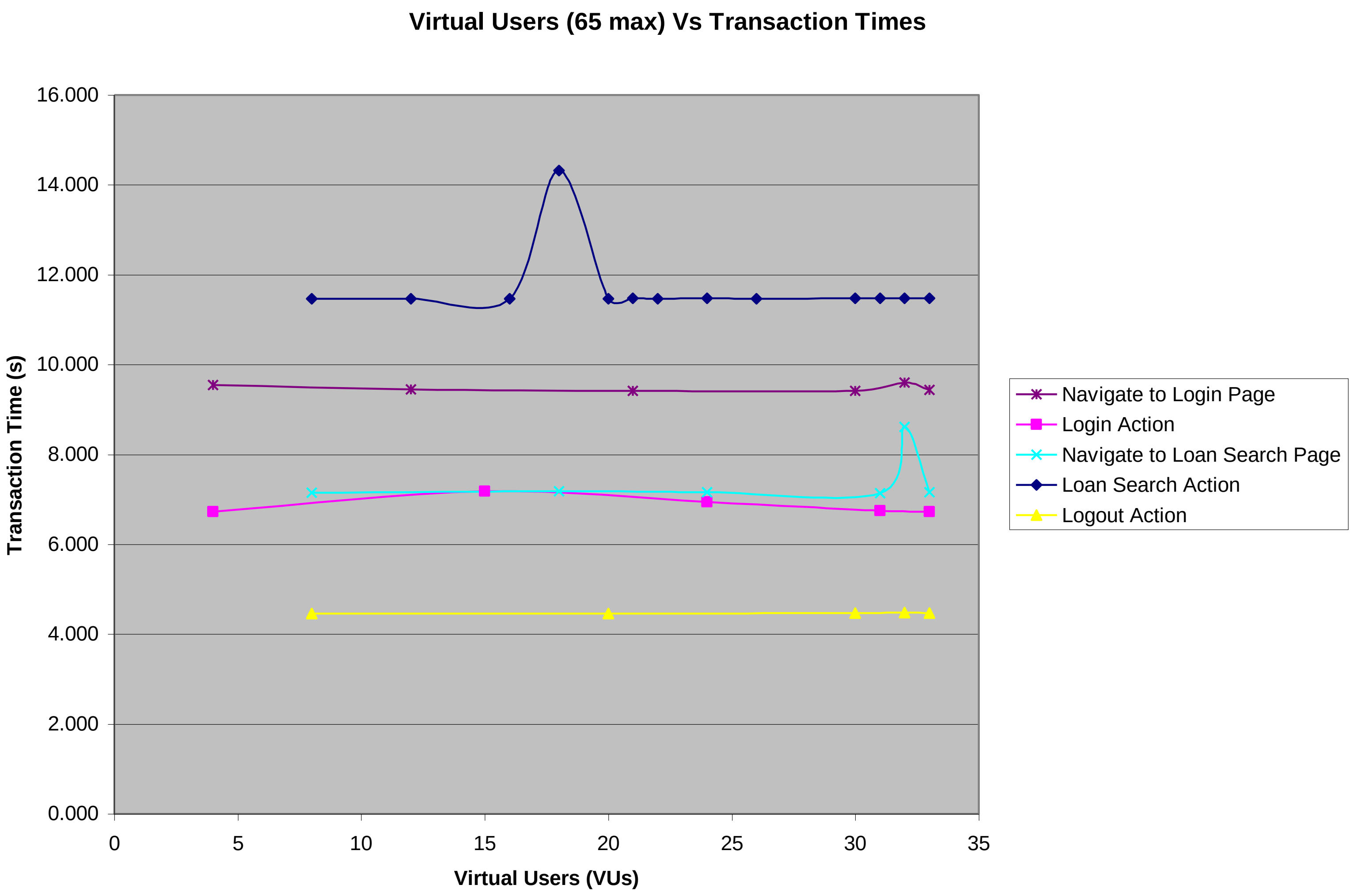


Figure 3.6.3 – 65 Maximum VUs, Average of three test runs

VUs	T_ACT_LOANSEARCH	T_ACT_LOGIN	T_ACT_LOGOUT	T_NAV_LOANSEARCH	T_NAV_LOGIN
4		6.740			9.420
8				7.150	
10			4.660		
12	11.470				9.420
14	11.480				
15	11.470	6.760			
18	11.470			7.140	
21	11.553		4.480		9.510
24	11.700	6.900			
25	11.735				
27				7.650	
30	12.198		4.710		
32	11.504				9.610
33	11.578	6.843	4.545	7.273	11.440
34	11.597	6.870	4.760	8.020	9.630

Table 3.6.4 – 75 Maximum VUs, Average of three test runs

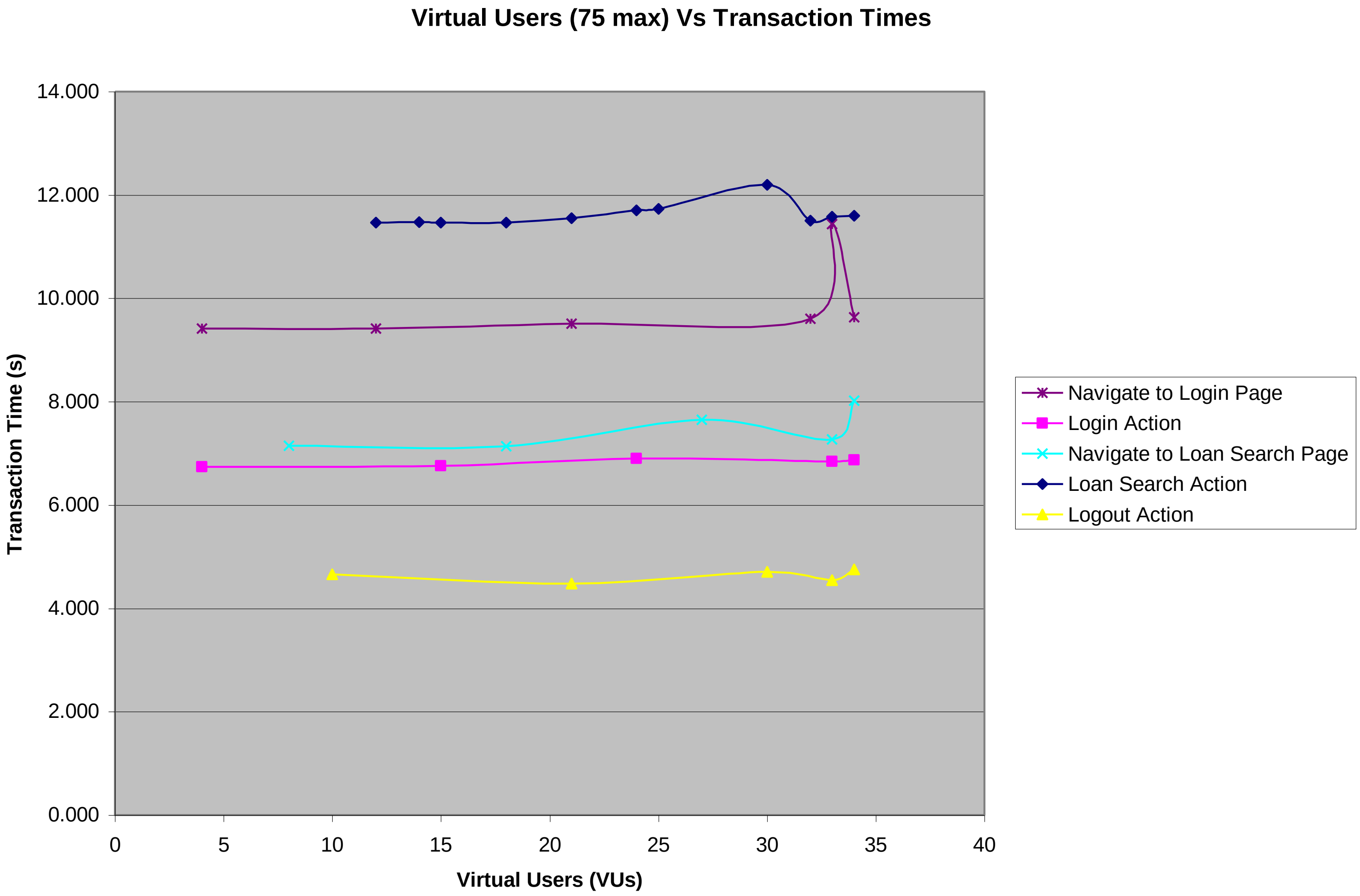


Figure 3.6.4 – 75 Maximum VUs, Average of three test runs

VUs	T_ACT_LOANSEARCH	T_ACT_LOGIN	T_ACT_LOGOUT	T_NAV_LOANSEARCH	T_NAV_LOGIN
4		7.100	4.560		9.550
7	11.480				
8				7.200	
11	11.500				
12	11.660				9.630
15	11.510	7.060			
18	11.480		4.460	7.160	
21	11.475				
22	11.480				
24	11.538				9.650
27		6.980		7.140	
28	11.460		4.500	10.100	
30	11.840		4.880		
31	11.558	6.805			9.740
32	11.507	6.755	4.475	7.140	9.465
33	11.741	6.720	4.475	7.303	9.465
34	11.497	6.770	7.780		

Table 3.6.5 – 100 Maximum VUs, Average of three test runs

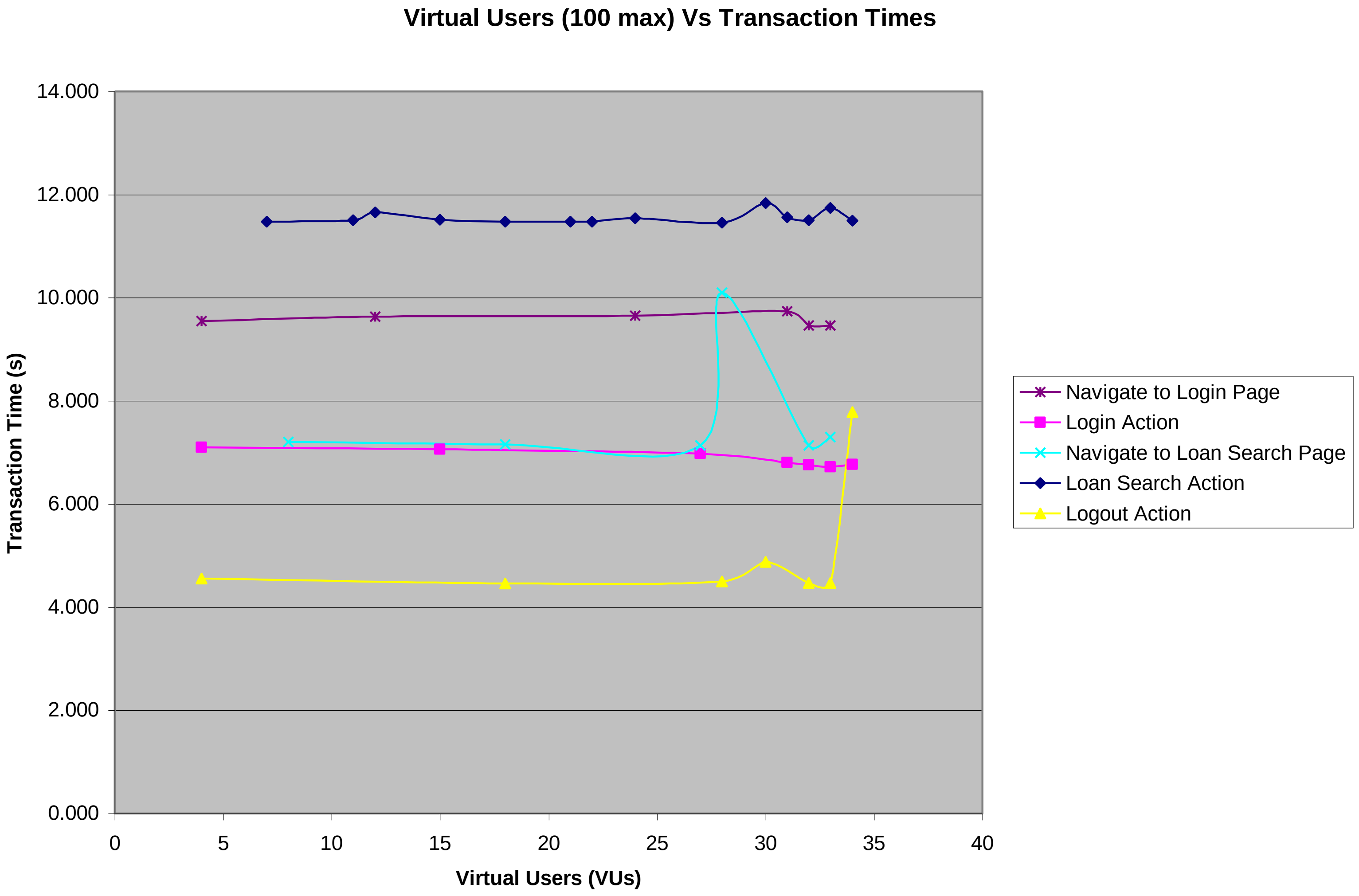


Figure 3.6.5 – 100 Maximum VUs, Average of three test runs

4 Future Work

There is further scope for improvement in the performance of the loan search functionality of the Broker Blueprint application in particular, and all of the available functionality as a whole. Some of the areas already identified for performance improvement are listed below and should be explored.

SQL tuning and schema refactoring
The SQL queries run for loan search are dynamically built by the application, and then executed. There is much scope for improving the loan search SQL queries. The ability to refactoring the schema along with tuning the SQL gives additional flexibility in tuning for optimum performance.

Application server configuration
The application server configuration can be further tweaked to achieve a better performance. The challenge for this is to run repeated tests with configuration changes to find the point of balance between

Static content server
One of the most beneficial elements in having a dedicated web server like Apache which takes web user requests and passes them on to the application server is that static content like images , java script and style sheet files can be served by the web server, while the application server deals with the logical component of the application.

This speeds up the response time dramatically, since web servers are highly optimized to serve static content, and the application server can dedicate its resources to the logical processing of the application requests.

Load Testing with SNMP metrics collection
Load testing can be performed with SNMP metrics collection configured so that database and application server metrics like transactions per second, CPU load, etc. can be collected, and a single time-lined view can be presented with all the metrics. This helps in better locating system bottlenecks and also to better understand system performance and reliability under varying load conditions.

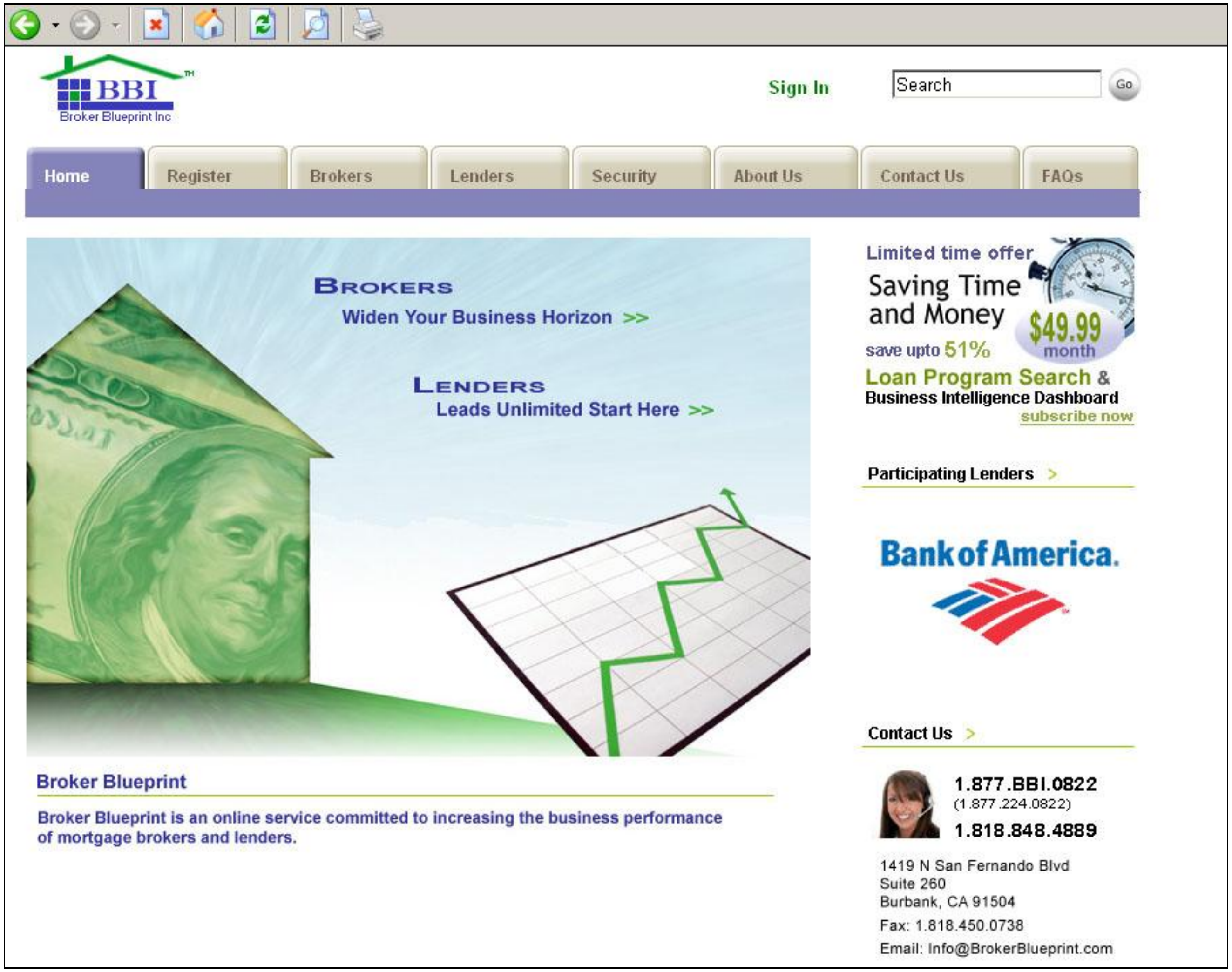
5 Appendix A – References

1. "**OpenSTA User Guide**" by OpenSTA Community
   http://www.opensta.org/docs/ug/

2. "**User Experience, not Metrics**" by Scott Barber
   http://www-128.ibm.com/developerworks/rational/library/4228.html

3. "Script Control Language Reference" by OpenSTA Community
   http://www.opensta.org/docs/sclref/

6 Appendix B – Application Screenshots



BROKERS
Widen Your Business Horizon >>

LENDERS
Leads Unlimited Start Here >>

Broker Blueprint

Broker Blueprint is an online service committed to increasing the business performance of mortgage brokers and lenders.

Limited time offer
Saving Time and Money
$49.99 month
save upto 51%
Loan Program Search & Business Intelligence Dashboard
subscribe now

Participating Lenders >

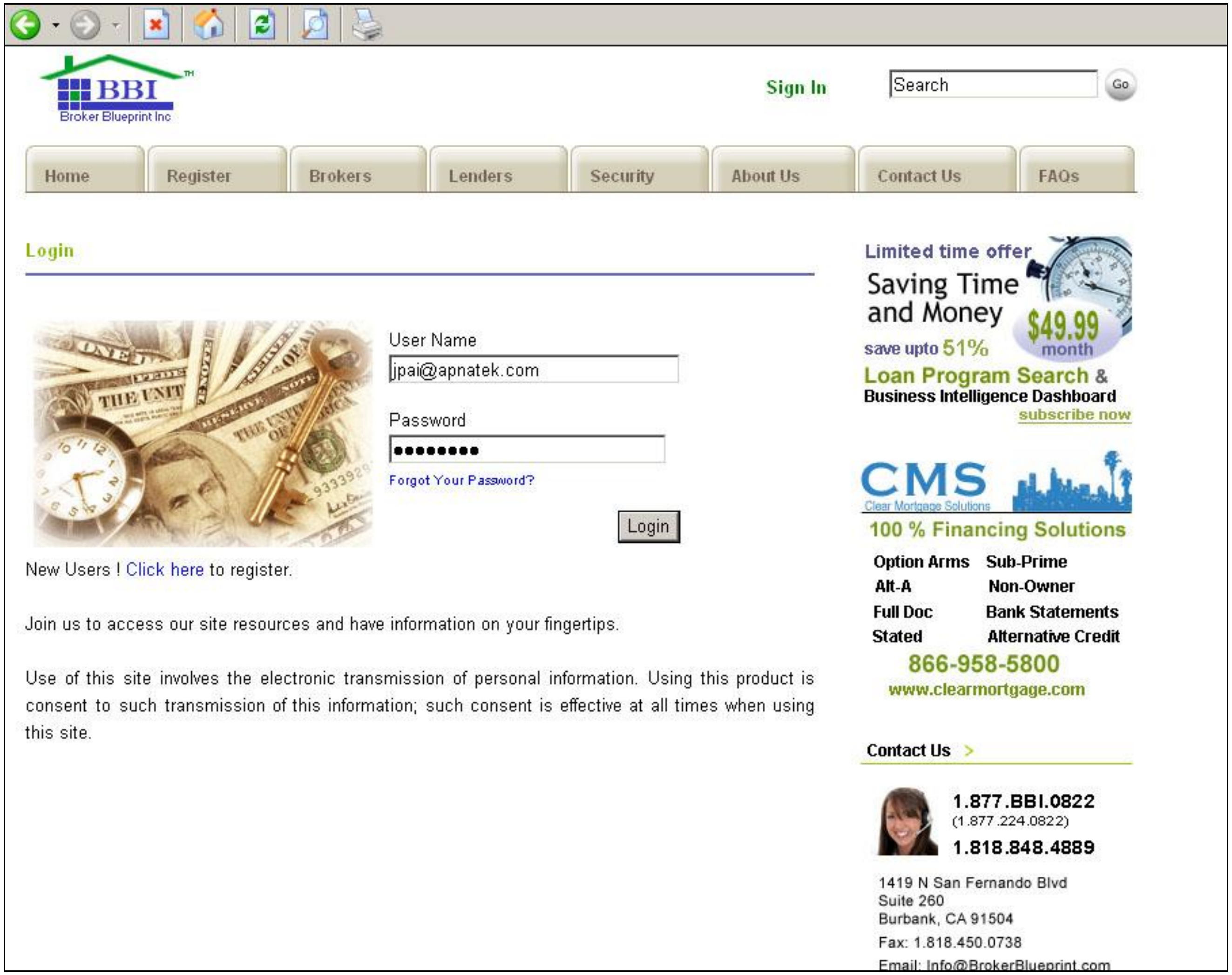
Bank of America.

Contact Us >

1.877.BBI.0822
(1.877.224.0822)
1.818.848.4889

1419 N San Fernando Blvd
Suite 260
Burbank, CA 91504
Fax: 1.818.450.0738
Email: Info@BrokerBlueprint.com

Figure 3.6.1 – Main Page (Unauthenticated User)



User Name: jpai@apnatek.com

Password: ●●●●●●●●●

New Users ! Click here to register.

Join us to access our site resources and have information on your fingertips.

Use of this site involves the electronic transmission of personal information. Using this product is consent to such transmission of this information; such consent is effective at all times when using this site.

Limited time offer
Saving Time and Money
$49.99 month
save upto 51%
Loan Program Search & Business Intelligence Dashboard
subscribe now

CMS Clear Mortgage Solutions
100 % Financing Solutions

Option Arms	Sub-Prime
Alt-A	Non-Owner
Full Doc	Bank Statements
Stated	Alternative Credit

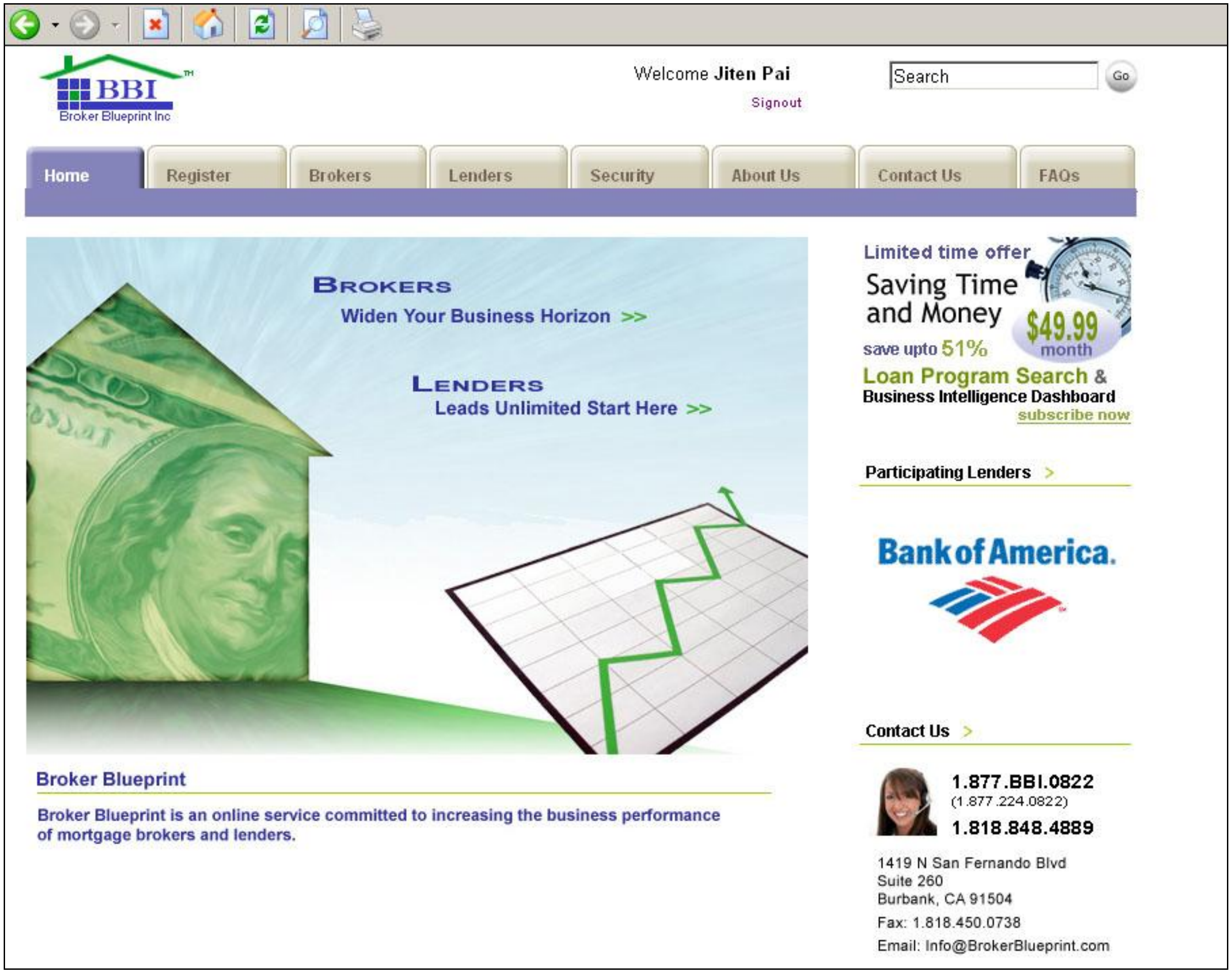
866-958-5800
www.clearmortgage.com

Contact Us >

1.877.BBI.0822
(1.877.224.0822)
1.818.848.4889

1419 N San Fernando Blvd
Suite 260
Burbank, CA 91504
Fax: 1.818.450.0738
Email: Info@BrokerBlueprint.com

Figure 3.6.1 – Login Page



Welcome **Jiten Pai**

Signout

Search | Go

BROKERS

Widen Your Business Horizon >>

LENDERS

Leads Unlimited Start Here >>

Limited time offer

Saving Time and Money

save upto 51%

$49.99 month

Loan Program Search & Business Intelligence Dashboard

subscribe now

Participating Lenders >

Contact Us >

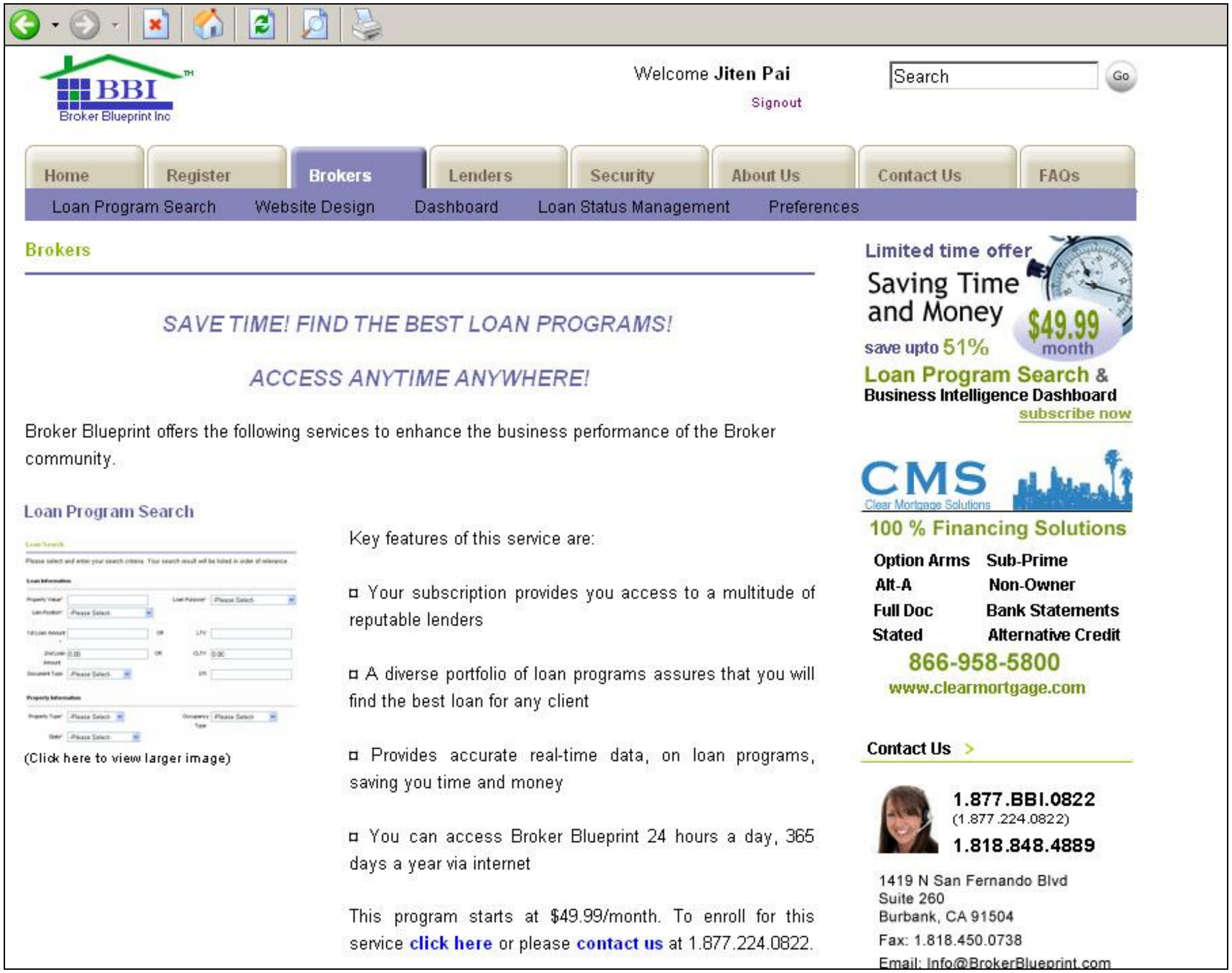
1.877.BBI.0822
(1.877.224.0822)
1.818.848.4889

1419 N San Fernando Blvd
Suite 260
Burbank, CA 91504
Fax: 1.818.450.0738
Email: Info@BrokerBlueprint.com

Broker Blueprint

Broker Blueprint is an online service committed to increasing the business performance of mortgage brokers and lenders.

Figure 3.6.2 – Main Page After Login Action (Authenticated User)



Brokers

SAVE TIME! FIND THE BEST LOAN PROGRAMS!

ACCESS ANYTIME ANYWHERE!

Broker Blueprint offers the following services to enhance the business performance of the Broker community.

Loan Program Search

(Click here to view larger image)

Key features of this service are:

¤ Your subscription provides you access to a multitude of reputable lenders

¤ A diverse portfolio of loan programs assures that you will find the best loan for any client

¤ Provides accurate real-time data, on loan programs, saving you time and money

¤ You can access Broker Blueprint 24 hours a day, 365 days a year via internet

This program starts at $49.99/month. To enroll for this service **click here** or please **contact us** at 1.877.224.0822.

Limited time offer
Saving Time and Money
$49.99 month
save upto 51%
Loan Program Search & Business Intelligence Dashboard
subscribe now

CMS
Clear Mortgage Solutions
100 % Financing Solutions

Option Arms	Sub-Prime
Alt-A	Non-Owner
Full Doc	Bank Statements
Stated	Alternative Credit

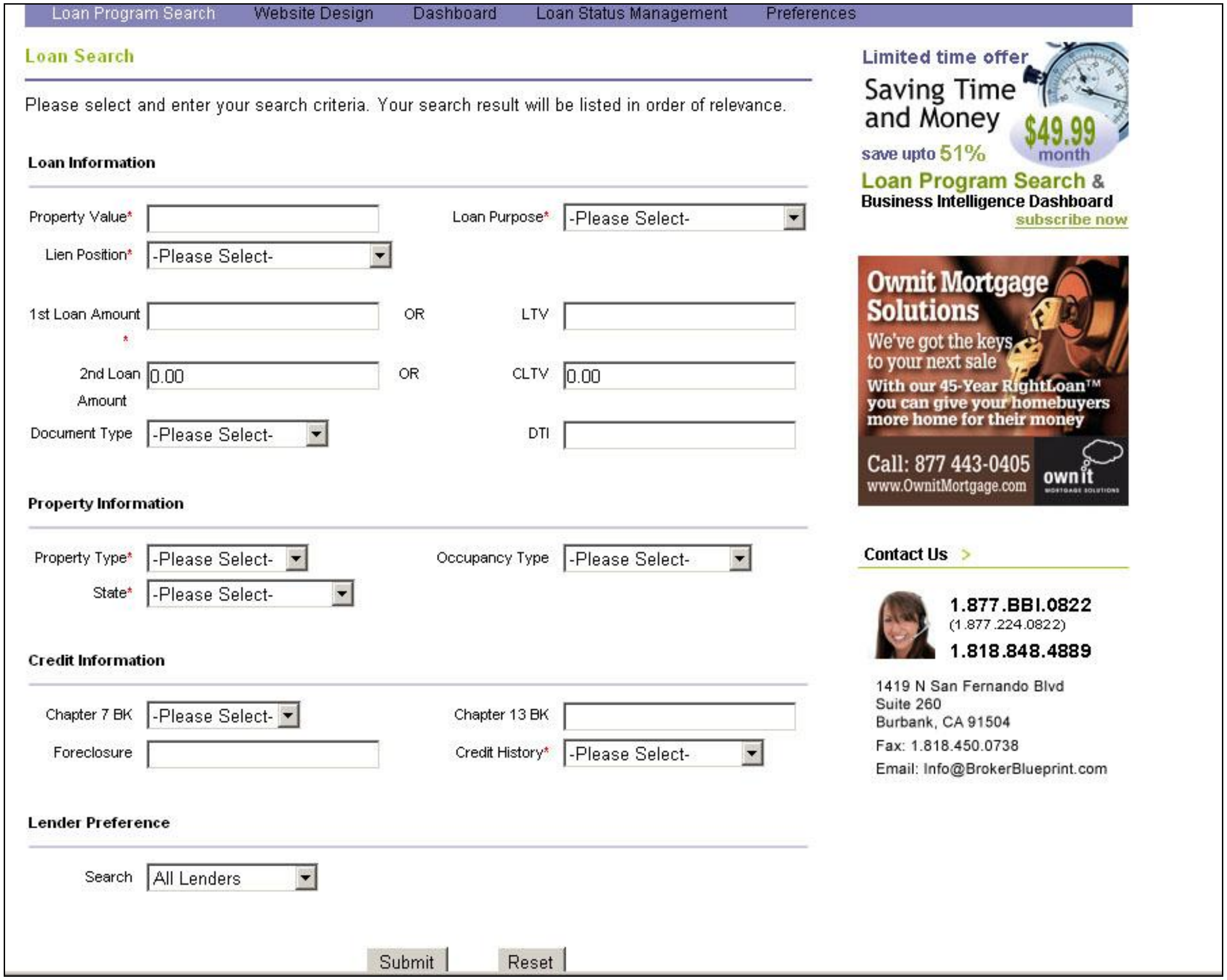
866-958-5800
www.clearmortgage.com

Contact Us >

1.877.BBI.0822
(1.877.224.0822)
1.818.848.4889

1419 N San Fernando Blvd
Suite 260
Burbank, CA 91504
Fax: 1.818.450.0738
Email: Info@BrokerBlueprint.com

Figure 3.6.3 – Navigation to Loan Search Page (Step 1)



Loan Search

Please select and enter your search criteria. Your search result will be listed in order of relevance.

Loan Information

Property Value*

Loan Purpose* -Please Select-

Lien Position* -Please Select-

1st Loan Amount * OR LTV

2nd Loan Amount 0.00 OR CLTV 0.00

Document Type -Please Select-

DTI

Property Information

Property Type* -Please Select-

Occupancy Type -Please Select-

State* -Please Select-

Credit Information

Chapter 7 BK -Please Select-

Chapter 13 BK

Foreclosure

Credit History* -Please Select-

Lender Preference

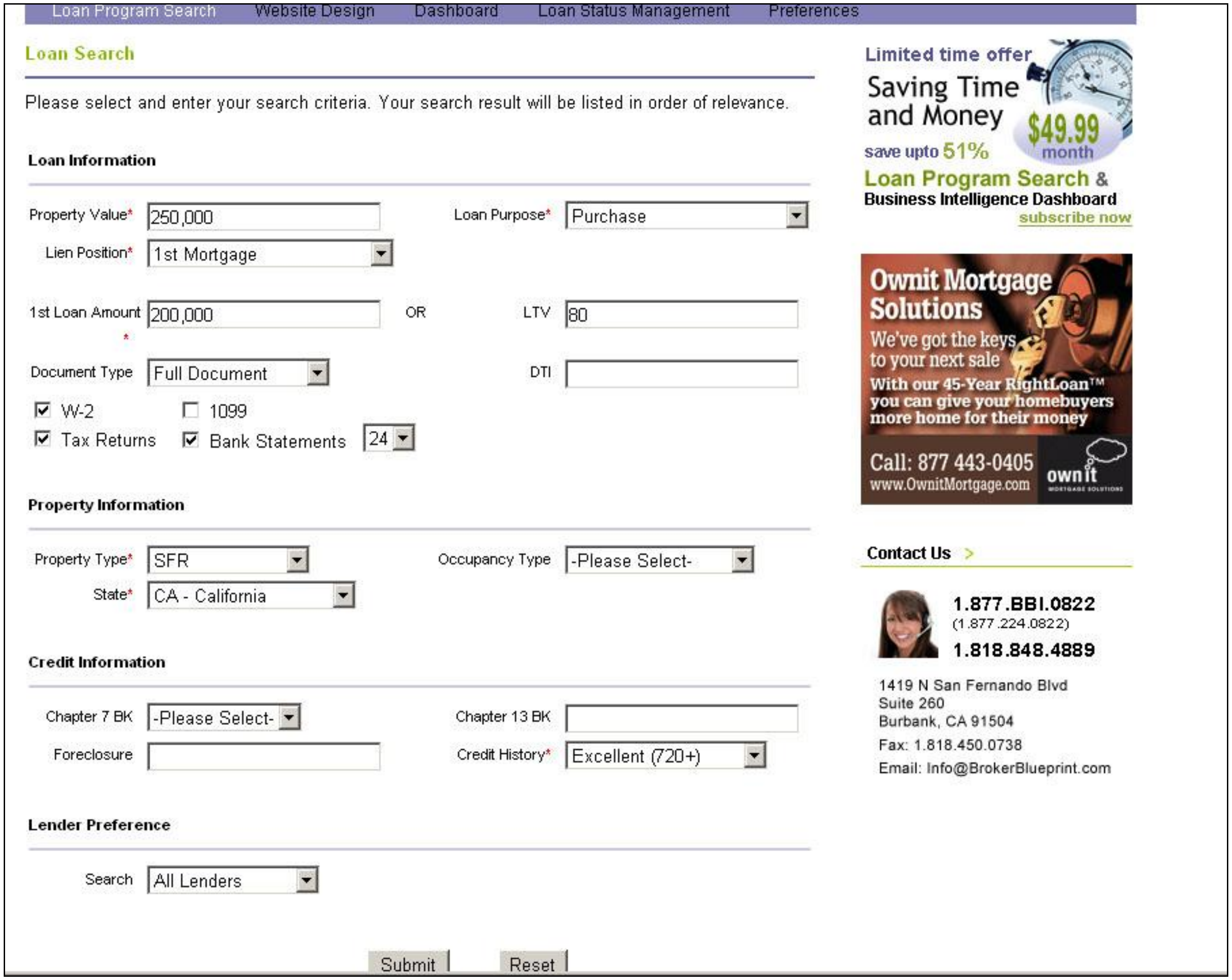
Search All Lenders

Contact Us >

1.877.BBI.0822
(1.877.224.0822)
1.818.848.4889

1419 N San Fernando Blvd
Suite 260
Burbank, CA 91504
Fax: 1.818.450.0738
Email: Info@BrokerBlueprint.com

Figure 3.6.4 –Loan Search Page



Loan Search

Please select and enter your search criteria. Your search result will be listed in order of relevance.

Loan Information

Property Value* 250,000
Loan Purpose* Purchase
Lien Position* 1st Mortgage
1st Loan Amount* 200,000 OR LTV 80
Document Type Full Document
DTI

☑ W-2 □ 1099
☑ Tax Returns ☑ Bank Statements 24

Property Information

Property Type* SFR
Occupancy Type -Please Select-
State* CA - California

Credit Information

Chapter 7 BK -Please Select-
Chapter 13 BK
Foreclosure
Credit History* Excellent (720+)

Lender Preference

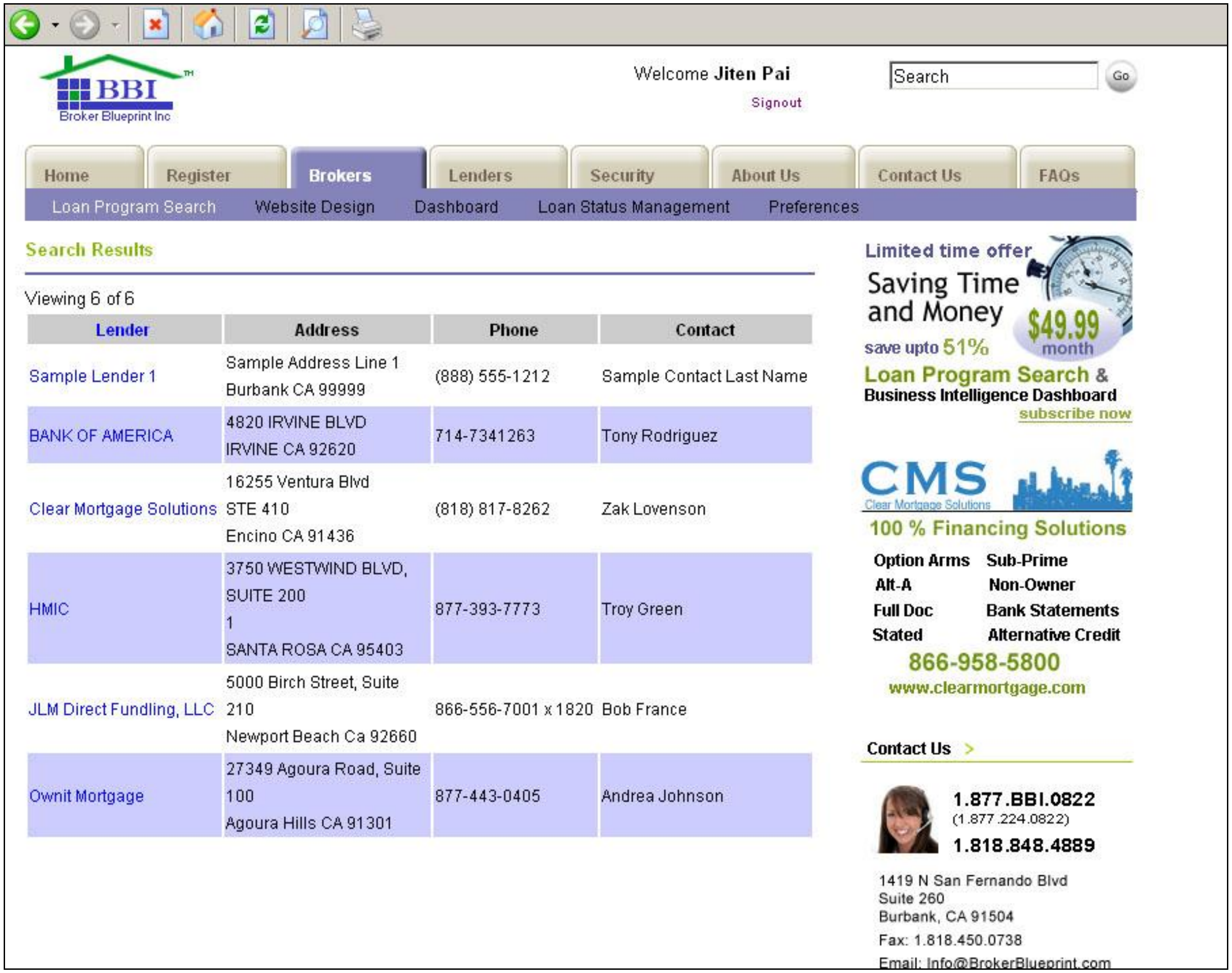
Search All Lenders

Contact Us >

1.877.BBI.0822
(1.877.224.0822)
1.818.848.4889

1419 N San Fernando Blvd
Suite 260
Burbank, CA 91504
Fax: 1.818.450.0738
Email: Info@BrokerBlueprint.com

Figure 3.6.5 – Loan Search Page with Search Data



Welcome **Jiten Pai**

Signout

Search Results

Viewing 6 of 6

Lender	Address	Phone	Contact
Sample Lender 1	Sample Address Line 1 Burbank CA 99999	(888) 555-1212	Sample Contact Last Name
BANK OF AMERICA	4820 IRVINE BLVD IRVINE CA 92620	714-7341263	Tony Rodriguez
Clear Mortgage Solutions	16255 Ventura Blvd STE 410 Encino CA 91436	(818) 817-8262	Zak Lovenson
HMIC	3750 WESTWIND BLVD, SUITE 200 1 SANTA ROSA CA 95403	877-393-7773	Troy Green
JLM Direct Fundling, LLC	5000 Birch Street, Suite 210 Newport Beach Ca 92660	866-556-7001 x 1820	Bob France
Ownit Mortgage	27349 Agoura Road, Suite 100 Agoura Hills CA 91301	877-443-0405	Andrea Johnson

Limited time offer

Saving Time and Money

$49.99 month

save upto 51%

Loan Program Search & Business Intelligence Dashboard

subscribe now

CMS

Clear Mortgage Solutions

100 % Financing Solutions

Option Arms	Sub-Prime
Alt-A	Non-Owner
Full Doc	Bank Statements
Stated	Alternative Credit

866-958-5800

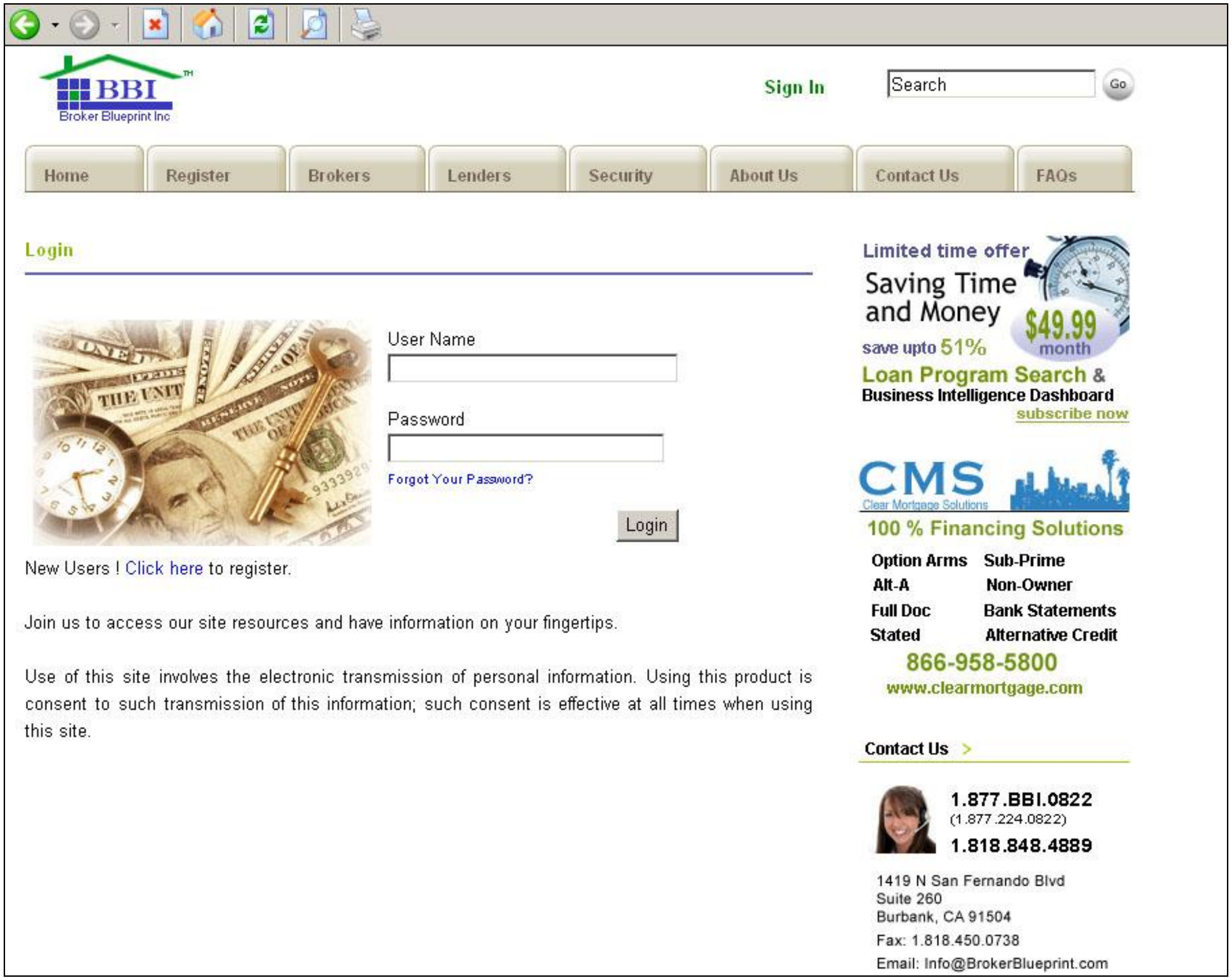
www.clearmortgage.com

Contact Us >

1.877.BBI.0822
(1.877.224.0822)
1.818.848.4889

1419 N San Fernando Blvd
Suite 260
Burbank, CA 91504
Fax: 1.818.450.0738
Email: Info@BrokerBlueprint.com

Figure 3.6.6 – Loan Search Results



Login

User Name

Password

Forgot Your Password?

New Users ! Click here to register.

Join us to access our site resources and have information on your fingertips.

Use of this site involves the electronic transmission of personal information. Using this product is consent to such transmission of this information; such consent is effective at all times when using this site.

Contact Us >

1.877.BBI.0822
(1.877.224.0822)
1.818.848.4889

1419 N San Fernando Blvd
Suite 260
Burbank, CA 91504
Fax: 1.818.450.0738
Email: Info@BrokerBlueprint.com

Figure 3.6.7 – User Logged Out