Chapter -7

CONCLUSIONS AND
FUTURE WORK
CHAPTER 7

Chapter - 7. Conclusions and Future Work

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Name of the Sub-Title</th>
<th>Page No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1</td>
<td>Conclusions</td>
<td>148</td>
</tr>
<tr>
<td>7.2.</td>
<td>Future Work</td>
<td>150</td>
</tr>
</tbody>
</table>
CHAPTER 7

CONCLUSIONS AND FUTURE WORK

This chapter presents the conclusions of the research work presented in this thesis. It also gives future work that can be carried out further.

7.1 Conclusions

In this research, we proposed a novel architectural pattern for web application development. The architecture includes many design patterns that help improve access time and throughput of web applications. The design patterns are pertaining to database access, connection pooling, caching and so on. The whole architecture is based on the MVC pattern.

MVC gives freedom to improve layers such as Model, View and Controller further without violating their assumed roles. We were motivated by this fact and proposed a novel architectural framework XWADF for improving performance of web applications. We have tested the architecture using state-of-the-art web application development with and without the usage of our architecture.

We measured the performance of web application, both manually and also with LoadUIWeb 2 testing tool that simulated the performance of application in the presence of the increased number of sessions and concurrent access. Response Time and Throughput are the two measures employed to know the performance of web
applications. The experimental results revealed that the proposed architecture improves performance by order of magnitude in terms of response time and throughput.

We studied fault tolerance its characteristics with respect to software products including web applications. We focused solutions based on architectural or design patterns for making applications fault tolerant. We enhanced our architectural pattern XWADF further to incorporate FT patterns. The FT patterns we implemented are Fault Tolerant Exception Handling Pattern (FTEHP) and Fault Tolerant Security Pattern (FTSP).

The former handles any kind of runtime errors while the latter is specialized for handling security related faults. The implementation of these patterns is done using AspectJ as AOP framework. This is done to have clear separation of concerns and avoid pollution. It also promotes code reusability as aspects can handle cross cutting concerns like security and exception handling.

These two patterns are incorporated into XWADF so as to make it more robust in terms of fault tolerance besides its existing functionality for making applications highly scalable and available. We tested our architectural pattern and the underlying FT design patterns using two case study web applications namely HMS and LMS.

We also identified design patterns that can improve scalability and availability of web applications. We enhanced our architectural pattern XWADF to incorporate the identified design patterns. The
enhanced architecture is applied to the existing case study applications such as LMS and HMS. Response Time and throughput attributes are used to measure scalability while MTTR metric is used to measure the availability of applications.

The throughput and Response Time of applications before and after adapting our architectural pattern were tested with number of virtual users using the testing tool LoadUIWeb 2. The empirical results revealed that our architectural pattern can improve reliability of web applications and make them robust in terms of scalability and availability.

### 7.2 Future Work

The future directions for further improvement of the architecture includes the extension of the proposed architecture gradually in order to include more design patterns and services that can leverage web applications in terms of quality and performance of web applications. One of the additional design patterns that will taken in proposed architecture is for maintainability. To improve maintainability we can consider different design patterns for maintainability and enhance the XWADF architectural pattern.

Similarly we improve usability using this proposed XWADF architectural pattern. To do this we need different design patterns for usability and consider this design patterns in XWADF architectural patterns.
We can improve functionality by considering design patterns related to this functionality. All these design patterns are taken in XWADF to improve functionality. Likewise we can take different design patterns related to efficiency in XWADF and improve the efficiency of web applications.

In this way we can enhance the XWADF for different design patterns.