ARCHITECTURE FOR AUTOMATIC SOURCE CODE COMMENT GENERATION

R.M.N.S Samarasinghe

(199360G)

This dissertation was submitted in partial fulfilment of the requirements for the degree of MSc in Computer Science Specializing in Software Architecture

Department of Computer Science and Engineering

University of Moratuwa

Sri Lanka

June 2022

DECLARATION

I declare that this is my work. This Thesis Report does not incorporate without acknowledgement any material previously submitted for a degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief. It does not contain any previously published material written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to the University of Moratuwa the non-exclusive right to reproduce and distribute my thesis, in whole or in part in print, electronic or other media. I retain the right to use this content in whole or interest in future works.

....

R.M Nuwan Shashika Samarasinghe Date

I certify that the declaration above by the candidate is accurate to the best of my knowledge and that this project report is acceptable for evaluation for the Thesis Report.

....

Dr. Indika Perera Date

(Supervisor)

ABSTRACT

This research tries to give a highly available software architecture to a source code commenting tool. Code commenting is helpful for many phases like code updates, bug fixes, etc. Developers need to know which task each code is performing, and also, adding comments for each code is essential. But most developers take more time to add comments which has created the problem of wasting time. Using an automated comment generator tool for source code, we can avoid that problem and save time for another task. Currently, there is not a properly defined architecture for an automated comment generator tool. Therefore, we are trying to give an evaluated architecture for that tool that supports overcoming the problem.

In this architecture, we plan to follow the microservice architectural concepts. Then modularization of each component based on the service it is going to provide. Also, plan to use ActiveMQ for the queue technique, Scheduling techniques to reschedule things, No SQL databases for data saving, Caching techniques to store temporary data, etc. This procedure ensures that it will provide a highly available architecture. Also, this architecture will have the following quality attributes: Maintainability, Performance, Interoperability, Usability, Availability, Reliability, Testability, Modifiability, Scalability, and Reusability. Finally, we use presentations, proof of concept, and ATAM & CBAM methods to validate whether the architecture is commercially viable or not.

Key Words: Development, Zookeeper, ActiveMQ, REST, Database, Queue, Asynchronous, Cache.

ACKNOWLEDGEMENTS

My profound gratitude goes to Dr Indika Perera, my supervisor, for the knowledge, supervision, advice, and guidance provided with his expertise throughout making the thesis a success.

My appreciation goes to my family for the motivation and support provided throughout my life. Also, I would like to thank my colleagues in the MSc batch and at my workplace for the help and support provided in managing my research work.

TABLE OF CONTENTS