BAB V
SIMPULAN, IMPLIKASI, DAN REKOMENDASI

5.1 Simpulan

Berdasarkan hasil penelitian dan pengujian secara empiris, maka didapatkan simpulan bahwa *refactoring* dengan ASATs memiliki dampak *code smells* dapat tereksplorasi. Setiap perangkat lunak dibagi menjadi sembilan versi telah dilakukan *refactoring code smells* secara tunggal dan kumulatif dengan menurunkan intensitas *code* smells pada Calculator (60%), Todolist (71%), dan Openflood (93%). Hasil setiap perangkat lunak yang telah dilakukan *refactoring* dilanjutkan dengan pengujian sesuai skenario pengujian di perangkat fisik, sehingga diperoleh bahwa *refactoring Member Ignoring Method* dan *Critical* menghasilkan kinerja yang baik pada konsumsi CPU dengan rata-rata -7,87% dan -9,90%. Terdapat juga hasil kinerja yang baik pada konsumsi memori bahwa *refactoring Blocker* dan tujuh *code smells* dengan rata-rata -3,52% dan -2,09%. Sehingga dapat disimpulkan bahwa temuan *code smells Blocker* dan *Critical* dapat mengurangi konsumsi CPU dan memori. Hasil statistik yang diperoleh secara signifikan dapat mengutamakan bahwa *refactoring code smells* bersifat krusial untuk menyesuaikan persyaratan kebutuhan non-fungsional pada perangkat lunak khususnya penggunaan sumber daya dan untuk melakukan proses perangkat lunak secara *multitasking*. Selain itu, terkadang *refactoring* tunggal sudah dapat cukup signifikan untuk mengurangi konsumsi CPU dan memori, sehingga penelitian ini memiliki manfaat untuk pengembang atau perusahaan perangkat lunak dengan mengurangi biaya yang tinggi, usaha yang tidak memakan banyak tenaga, dan waktu pemeliharaan yang lebih singkat.

5.2 Implikasi

Penelitian ini memiliki implikasi dengan mengeksplorasi kembali *code smells* yang belum diteliti. Setiap perangkat lunak yang memiliki *code smells* dapat dianalisis dengan SonarQube dan dilakukan *refactoring* untuk memperoleh konsistensi kebutuhan non-fungsional khususnya sumber daya CPU dan memori. Oleh karena itu, setiap perangkat lunak memiliki pengaruh yang berbeda terhadap *refactoring code smells* sesuai kebutuhan, tergantung cara melakukan *refactoring* seperti secara tunggal ataupun kumulatif.

5.3 Rekomendasi

Berdasarkan proses dan hasil penelitian, maka terdapat rekomendasi terhadap penelitian selanjutnya, sebagai berikut:

1. Melakukan eksplorasi *code smells* Android *DrawAllocation*, *WakeLock*, *Recycle*, *ObsoleteLayoutParam*, *ViewHolder*, *OverDraw*, *UnusedResource*, *UselessParent* terhadap penggunaan CPU dan memori.
2. Melakukan penelitian dengan *corpus* yang mengandung kompleksitas kode program yang tinggi seperti *Facebook*, *Signal*, *Telegram*.
3. Melakukan pengujian setiap perangkat lunak yang telah dilakukan *refactoring* maupun belum dilakukan *refactoring* untuk lebih menghemat waktu.
4. Mengembangkan *tools* untuk otomatisasi *refactoring code smells* yang berfokus untuk mengurangi konsumsi CPU dan memori agar lebih menghemat tenaga dan waktu.
5. Melakukan penelitian *refactoring code smells* terhadap bahasa pemrograman selain Java dan Kotlin.