# BAB I
# PENDAHULUAN

## 1.1. Latar Belakang

Proses pengembangan perangkat lunak merupakan sesuatu hal yang tidak mudah, dibutuhkan kemampuan, pengetahuan dan teknologi lain yang mendukung. Pada proses pembangunan dan pengembangan sistem terdapat beberapa tahapan umum yaitu analisis, desain, *coding*, *testing* dan *implementation*. Tahapan-tahapan tersebut terdapat dalam banyak metode diantaranya adalah *Waterfall model, V model Agile methods, Incremental Process Model*. Hampir pada seluruh metode yang ada, tahapan pertama adalah kegiatan analisis, yaitu penguraian dari suatu sistem informasi yang utuh kedalam bagian-bagian komponennya dengan maksud untuk mengidentifikasikan dan mengevaluasi permasalahan-permasalahan, kesempatan-kesempatan, hambatan-hambatan yang terjadi dan kebutuhan yang diharapkan sehingga dapat diusulkan perbaikannya. Kegiatan analisis ini memiliki tujuan untuk merancang sistem baru maupun menyempurnakan sistem yang sudah ada sebelumnya. Analisis yang dilakukan adalah untuk mengetahui kebutuhan dari pengguna yang akan menggunakan sistem dan respon apa yang dilakukan oleh sistem untuk memenuhi dari kebutuhan pengguna tersebut. Sehingga pada tahapan analisis ini akan menghasilkan sebuah dokumen *SRS (Software Requirements Specification)* atau SKPL (Spesifikasi Kebutuhan Perangkat Lunak). (Sarwono, dan Kurniawan, 2019).

Dokumen *SRS (Software Requirements Specification)* atau SKPL (Spesifikasi Kebutuhan Perangkat Lunak) adalah sebuah dokumen yang berisikan spesifikasi kebutuhan sebuah perangkat lunak yang menjalankan fungsi tertentu dalam suatu lingkungan yang spesifik. Dalam *SRS*, suatu spesifikasi kebutuhan dijelaskan secara rinci beserta fungsi, kebutuhan sistem, dan batasan sistem yang bertujuan sebagai panduan dalam mengembangkan suatu perangkat lunak yang

sesuai dengan tujuan pembuatan perangkat lunak yang telah dikonsultasikan kepada *stakeholder*. (Azhary, dkk, 2019). Hal ini menyebabkan, kualitas dokumen *SRS* yang dihasilkan akan memberikan pengaruh terhadap proses pengembangan perangkat lunak berikutnya. *SRS* memiliki peran yang sangat penting dalam pengembangan perangkat lunak. *SRS* digunakan dalam implementasi desain, pengamatan proyek, verifikasi dan validasi, serta dalam pelatihan perangkat lunak. (Lelywiary, dkk, 2019).

*SRS* yang menjadi panduan dalam pembuatan sistem ini dapat membantu dalam pembuatan sistem menjadi lebih cepat. Penggunaan kebutuhan yang tepat dan sesuai yang diinginkan oleh pengguna akan menentukan kualitas dari sistem itu sendiri. Terkadang dalam pembuatan *SRS* itu sendiri membutuhkan waktu yang cukup lama, karena harus menyesuaikan dengan kebutuhan yang diinginkan oleh pengguna. Terkadang halangan ini dapat membuat pengerjaan sistem menjadi lebih lama dan memakan banyak biaya. Oleh karena itu, muncullah sebuah ide untuk menyelesaikan masalah ini yang dimana dapat membuat waktu pengerjaan *SRS* sistem menjadi lebih efisien, yaitu dengan membuat sistem pendeteksi kemiripan kebutuhan perangkat lunak pada spesifikasi kebutuhan perangkat lunak dengan menggunakan algoritma *rabin-karp*.

Sistem pendeteksi kemiripan kebutuhan perangkat lunak dengan menggunakan algoritma *rabin-karp* ini nanti akan digunakan untuk mengecek apakah ada fungsi *use case* yang mirip dari dokumen *SRS* yang berbeda, dan jika ada maka nanti fungsi *use case* yang mirip ini bisa digunakan kembali (*reuse)* untuk menghemat waktu, tenaga dan materi dalam pembuatan sistem tersebut. Sistem ini menggunakan algoritma *rabin-karp* yang akan membantu dalam proses pengecekan kemiripan pada kebutuhan perangkat lunak dari dokumen *SRS* berbeda dan menggunakan metode *Rational Unified Process (RUP)* dalam pengembangannya.

Algoritma *Rabin-karp* adalah salah satu algoritma pencarian *string* dikembangkan oleh Michael O. Rabin dan Richard M. Karp pada tahun 1987 yang menggunakan fungsi *hashing* untuk menemukan *pattern* di dalam *string* teks. Algoritma *Rabin-karp* merupakan algoritma untuk pencocokan *string multi*

*pattern*. Pada pencocokan *string multi pattern* paket atau informasi yang dicari berdasarkan beberapa susunan pola *string*. Algoritma *Rabin-karp* melakukan pergeseran dari kiri ke kanan, fungsi dari algoritma *Rabin-karp* menghasilkan efisiensi waktu yang baik dalam pencarian *string* yang memiliki lebih dari satu pola. Dengan penjelasan tersebut maka dengan menggunakan algoritma *rabin-karp* nanti akan sangat membantu dalam pengecekan kemiripan yang ada pada dokumen SKPL berbeda.

Metode *Rational Unified Process (RUP)* merupakan salah satu pengembangan perangkat lunak dengan menggunakan pendekatan yang disiplin dalam melakukan setiap tugas dan tanggung jawabnya dalam sebuah organisasi. Tujuan dari *RUP* itu sendiri adalah untuk dapat menjamin produksi berkualitas tinggi dan memenuhi semua kebutuhan pihak yang berkepentingan, termasuk waktu dan biaya sesuai dengan rencana yang telah disepakati sebelumnya. (Hutahean., dkk, 2019). Metode ini dirasa tepat untuk dapat mengembangkan sistem pendeteksi kemiripan kebutuhan perangkat lunak pada spesifikasi kebutuhan perangkat lunak dengan baik.

Dari permasalahan yang telah dipaparkan di atas maka dibutuhkan sebuah sistem pendeteksi kemiripan kebutuhan perangkat lunak untuk penggunaan kembali (*reuse*) kebutuhan yang sudah ada. Sistem ini nantinya akan dibuat dalam bentuk *website* dengan menggunakan bahasa *php* agar dapat dengan mudah diakses oleh pengguna.

## 1.2. Rumusan Masalah

Berdasarkan latar belakang yang ada di atas maka dapat diketahui rumusan masalahnya, yaitu bagaimana cara mendeteksi kemiripan kebutuhan perangkat lunak pada fungsi *use-case* spesifikasi kebutuhan perangkat lunak dengan menggunakan algoritma *rabin-karp*?

## 1.3. Tujuan

Adapun tujuan dari penelitian ini, yaitu untuk mendeteksi kemiripan kebutuhan perangkat lunak pada fungsi *use-case* spesifikasi kebutuhan perangkat lunak dengan menggunakan algoritma *rabin-karp*.

## 1.4. Manfaat

Adapun manfaat dari penelitian ini, yaitu untuk penggunaan kembali (*reuse*) fungsi kebutuhan perangkat lunak yang sudah ada.