BAB II

LANDASAN TEORI

2.1 Penelitian Terdahulu

Penelitian terdahulu merupakan sub bab landasan untuk mendapatkan acuan dengan mendapatkan informasi dari penelitian dengan topik yang mirip. Acuan tersebut berguna untuk mencegah plagiarisme. Kajian pendahuluan dari penelitian terdahulu dirangkum dalam tabel berikut:

Tabel 2. 1 Tabel Penelitian Terdahulu

No	Judul Jurnal	Penulis/Tahun	Hasil	Kesimpulan
1	*Credit Card Sales Performance Dashboard* [2]	Darrel John Beltran, Yves Kangleon, Ariel Kelly Balan, Joel de Goma / 2021	Penggunaan visualisasi membantu pengguna untuk memahami informasi yang dibutuhkan untuk menghasilkan wawasan secara efektif dan membantu mereka dalam mengambil keputusan untuk mendorong kinerja bisnis pada dengan lebih cepat.	Penggunaan *dashboard* membantu dalam pengambilan keputusan ketika data yang perlu dianalisa tidak dapat dicerna secara mentah.
2	Perancangan Sistem Informasi *Monitoring* Data Sewa Atm Dengan Metode *Rapid Application Development* (Rad) Berbasis Web (Studi Kasus : Pt. Bank Mandiri (Persero) Tbk Area Tangerang Kisamaun) [4]	Surtikanti1, Zainal Ali Syahron / 2023	Sistem yang dikembangkan dapat membantu PT Bank Mandiri, sistem juga mampu mempermudah monitoring data sewa ATM, dan mendapatkan metode pengelolaan dan *monitoring* data yang lebih efisien dan efektif untuk dijadikan bahan referensi dalam pengembangan sistem baru	Pengembangan aplikasi menggunakan metode *System Development Life Cycle* dengan model *RAD* dapat berhasil menciptakan aplikasi yang dapat mempermudah pekerjaan pegawai PT Bank Mandiri
3	*Development of Enterprise Resource*	Muhammad Raihan Gifari	Penggunaan metode *RAD*	Metode pengembangan

No	Judul Jurnal	Penulis/Tahun	Hasil	Kesimpulan
	Planning (ERP) using the Rapid Application Development (RAD) Method for the Garment Industry in Indonesia[14]	Qowindra, Jansen Wiratama / 2023	sukses dalam mengembangkan sistem dalam waktu yang relatif singkat untuk menghasilkan 5 modul : *Accounting, Sales, Purchasing, Inventory,* dan *Manufacturing*	*RAD* dapat dengan sukses memotong waktu pengembangan besertakan menghasilkan sistem yang dapat bekerja dengan baik
4	*PYTHON: A PROGRAMMING LANGUAGE FOR SOFTWARE INTEGRATION AND DEVELOPMENT* [22]	M. F. Sanner / 2020	Pendekatan kerangka skrip ini memiliki banyak kekuatan dan manfaat. Telah dibuktikan: peningkatan dramatis dalam produktivitas, serta kemampuan untuk menggunakan kode berulang.	Penggunaan bahasa *python* merupakan bahasa pemrograman yang sesuai dimana banyak bagian dari kode yang akan berulang besertakan dengan adanya fitur *library* untuk memudahkan pengembangan.
5	*Web Application Vulnerability Detection Using Taint Analysis and Black-box Testing*[25]	Heribertus Yulianton, Agung Trisetyarso, Wayan Suparta, Bahtiar Saleh Abbas, Chul Ho Kang/2020	Metode pengujian *black box* berbasis *equivalence partitions* membantu proses pembuatan case pengujian	Setelah mengetahui kesalahan dapat dilakukan perbaikan untuk dapat menghasilkan aplikasi dengan kualitas lebih baik
6	*DESIGN AND DEVELOPMENT OF A WEB-BASED DATA VISUALIZATION SOFTWARE FOR POLITICAL TENDENCY IDENTIFICATION OF TWITTER'S USERS USING PYTHON DASH FRAMEWORK* [30]	Alexandros Britzolakis / 2020	Aplikasi serupa telah dibuat dengan *machine learning* yang dapat diamati orang dan biasanya mengecualikan data terstruktur. Selain itu, dapat menyimpulkan bahwa analisis media sosial terbukti berguna untuk menganalisis lanskap politik. negara.	Penggunaan bahasa pemrograman *python* sesuai untuk pengembangan sistem yang mengolah banyak data.

No	Judul Jurnal	Penulis/Tahun	Hasil	Kesimpulan
7	*JavaScript frameworks: Angular vs React vs Vue* [31]	Elar Saks / 2019	Penelitian membandingkan kerangka kerja daripada *javascript* dalam popularitas, kesulitan penerapan, agar pengguna dapat memilih kerangka kerja yang tepat dalam pembuatan aplikasi.	Penggunaan *framework react* sesuai dalam pengembangan aplikasi visualisasi.
8	*DESIGN OF SALES PERFORMANCE DASHBOARD BASED ON SALES FUNNEL & SALES FORCE AUTOMATION THEORIES: A CASE OF AN INDONESIAN ISLAMIC BANK*[34]	ALEXANDROS BRITZOLAKIS / 2020	*Dashboard* berhasil dikembangkan menggunakan *dash framework* yang dapat membaca sentimen politik berdasarkan data yang didapatkan dari *twitter*	*Framework dash* sesuai untuk digunakan dalam pengembangan *dashboard*
9	*THE DEVELOPMENT OF PERFORMANCE DASHBOARD VISUALIZATION WITH POWER BI AS PLATFORM* [35]	Surlisa Widjaja, Tuga Mauritsius / 2019	*Dashboard* kinerja intelijen bisnis dapat dijadikan acuan dalam proses pengambilan keputusan. Keterlibatan pengguna akan meningkatkan kualitas *dashboard*.	Keberadaannya sebuah *dashboard* dapat mempermudah pengambilan keputusan pihak manajeman, pengembendan *dashboard* juga dapat dilaksanakan dengan lebih baik jika ada masukan dari user
10	Implementasi dan Evaluasi Visualisasi Data Interaktif pada Publikasi Laporan Bulanan Data Sosial Ekonomi Indonesia[19]	Hafidz Isa Nasruddin Lizana, Farid Ridho / 2021	Dengan menggunakan metode LBDSE 6 jenis visualisasi berhasil diimplementasikan	Kegunaan dari chart berbeda – beda dan dapat disesuaikan dengan data yang ingin divisualisasikan.

Berdasarkan tabel 2.1 dapat didapatkan informasi yang didapatkan daripada penelitian terdahulu. Dimulai daripada informasi akan *SDLC, RAD, Dashboard, Framework, Dash, React, Python, Javascript, Black box testing*, dan aplikasi berbasis web. Penelitian akan memberikan fokus kepada *dashboard* dan r*apid*

application development yang akan digunakan sebagai fokus utama daripada penelitian.

Penelitian terdahulu memberikan masukan bahwa dalam pengembangan suatu aplikasi diperlukan sebuah alur untuk dapat mengembangkan sebuah aplikasi dengan efektif yang bernamakan *system development life cycle* (*SDLC*)[8]. Pada penelitian terdahulu dibahas juga tentang berbagai jenis *SDLC* yang telah terbukti dalam pengembangan suatu sistem[9]. Penelitian terdahulu juga membahas tentang kelebihan dan kekurangan daripada masing – masing model *SDLC*[10]. Penelitian terdahulu juga membahaskan tentang penggunaan jenis visualisasi yang tepat untuk tiap data yang ingin ditampilkan[19][20].

Penelitian terdahulu juga membahas tentang metode pengembangan *Rapid Application Development* (*RAD*), pada salah satu penelitian terdahulu terdapat beberapa kasus dimana metode *RAD* tidak sesuai dalam pengembangan aplikasi [13]. Terdapat penelitian terdahulu yang membahaskan tentang penggunaan metode *RAD* dalam mengembangkan aplikasi secara umumnya untuk mendapatkan gambaran umum akan penggunaan metode *RAD* [17][4][16][14]. Terdapat juga penelitian terdahulu yang menggunakan metode pengembangan *RAD* untuk mengembangkan dashboard yang memberikan acuan pengembangan *dashboard* menggunakan metode *RAD*[21][18] [35].

Penelitian terdahulu juga membahas tentang *black box.* Terdapat penelitian terdahulu yang membahas secara detail tentang penjelasan model black box[28]. Terdapat penelitian yang menggunakan metode *testing black box* untuk mengetes aplikasi yang sudah selesai dibuat[23][24][26], dan terdapat juga penelitian yang mengetes aplikasi yang telah dikembangkan menggunakan metode pengujian *black box*[25][27]. Pada penelitian terdahulu terdapat pengetesan terhadap aplikasi berbasis web yang kemudian dapat digunakan sebagai acuan pengembangan aplikasi[11][15][30].

Penelitian terdahulu juga membahas tentang bahasa pemrograman yang digunakan dalam pengembangan suatu aplikasi. *Python* dibahas dalam penelitian

terdahulu sebagai bahasa yang dapat digunakan oleh pemula atau yang sudah berpengalaman dalam pengembangan aplikasi[29][22]. *Javascript* juga dibahas sebagai bahasa pemrograman yang sesuai untuk membangun *front end* yang akan berinteraksi dengan pengguna [31]. Penelitian terdahulu juga membahas tentang *framework* yang digunakan dalam pengembangan suatu aplikasi framework seperti dash [30]dan react [31] yang kemudian akan digunakan sebagai dasar daripada aplikasi. Penelitian terdahulu juga membahas tentang jenis *chart* yang seharusnya digunakan dalam pembuatan

2.2 *System Development Life Cycle*

System Development Life Cycle (SDLC) merupakan metodologi yang digunakan untuk merancang desain, membangun, dan memelihara sistem dan aplikasi. Terdapat banyak model SDLC yang dapat digunakan seperti: *waterfall, spiral model, agile, RAD, V-Model,* dan *Iterative and Incremental Model*. SDLC menjabarkan fase penting untuk para perancang aplikasi. SDLC telah diselidiki oleh banyak peneliti yang kemudian mengusulkan model sendirinya lengkap dengan kelebihan dan kekurangannya masing – masing. Pada akhirnya semua metode *SDLC* memiliki serancangan fase yang harus diikuti dan diselesaikan oleh pengembang dan perancang sistem untuk dapat menghasilkan sistem yang dibutuhkan[9][10].

2.3 *Rapid Application Development*

Metode pengembangan *System Development Life Cycle (SDLC)* dengan model *Rapid Application Development (RAD)* adalah sebuah proses pengembangan software yang menekankan proses pengembangan software dengan waktu yang singkat. Metode *System Development Life Cycle* dengan model RAD menghabiskan banyak waktu pada dalam tahap perencanaan namun proses pengembangan aplikasi justru menggunakan waktu yang relatif lebih sedikit dikarenakan proses pengembangan aplikasi yang fleksibel[17].

Metode *System Development Life Cycle* dengan model RAD terdiri dari 4 metode tahapan utama dalam pengembangan software. Pertama merupakan *requirement planning* dimana masalah diidentifikasikan untuk mencari tahu apa

yang diperlukan dari aplikasi yang akan dikembangkan. Pada tahapan kedua user desain akan dibuat aplikasi protytype yang akan terus menerus disempurnakan sesuai dengan umpan balik dari pengguna sesuai dengan pengumpulan tanggapan dan saran yang diberikan. Tahapan ketiga merupakan pembuatan aplikasi sesuai dengan desain yang telah ditetapkan pada tahapan desain aplikasi. Tahapan terakhir merupakan *cutover* dan penyelesaian aplikasi, setelah *software* yang dibuat telah memuaskan spesifikasi yang dimintakan pengguna maka dilakukan sekali lagi pengecekan *error*. Jika sistem telah memenuhi permintaan dan tidak mengalami error maka *software* akan difinalisasikan dan akan diberikan persetujuan oleh klien[16].

Tahapan *requirement planning* merupakan tahap awal dalam suatu pengembangan sistem *RAD*. Pada tahapan perencanaan kebutuhan dilakukan identifikasi masalah dan desain sistem yang diperoleh dari pengguna pengguna yang bertujuan untuk mengidentifikasi tujuan dari aplikasi dan kebutuhan informasi yang diinginkan. Pada tahap ini keterlibatan kedua belah sangatlah penting dalam mengidentifikasi kebutuhan untuk pengembangan suatu sistem.

Di dalam tahap *user design* memerlukan masukkan dari pengguna yang terlibat untuk dapat melakukan desain sistem yang sesuai dengan kebutuhan user. Untuk dapat mencapai tujuan pada tahapan ini dilakukan proses desain dan proses perbaikan desain secara berulang-ulang apabila masih terdapat ketidaksesuaian desain terhadap kebutuhan pengguna yang telah diidentifikasi. Proses user desain mengumpulkan *feedback yang* digunakan untuk mereview desain sistem yang telah dibuat dan disepakati. Pada tahapan ini juga pengembanga aplikasi harus terus-menerus melakukan perbaikan pada sistem serta melakukan integrasi dengan bagian – bagian lain dari sistem sambil terus mempertimbangkan feedback dari pengguna.

Jika proses berjalan lancar maka dapat berlanjut ke tahapan berikutnya. Tahapan *construction* dimana aplikasi difinalisasikan dan dibuat secara sepenuhnya. Pada proses *construction* aplikasi sudah final dan tidak dapat diubah lagi.

Cutover atau penyelesaian produk merupakan tahapan dimana pengembang aplikasi mengakhiri pengembangan sistem yang telah disetujui pada tahapan sebelumnya. Sebelum sistem diserahkan kepada pengguna, terlebih dahulu dilakukan proses pengujian terhadap program untuk mendeteksi kesalahan yang ada pada sistem yang dikembangkan [16].

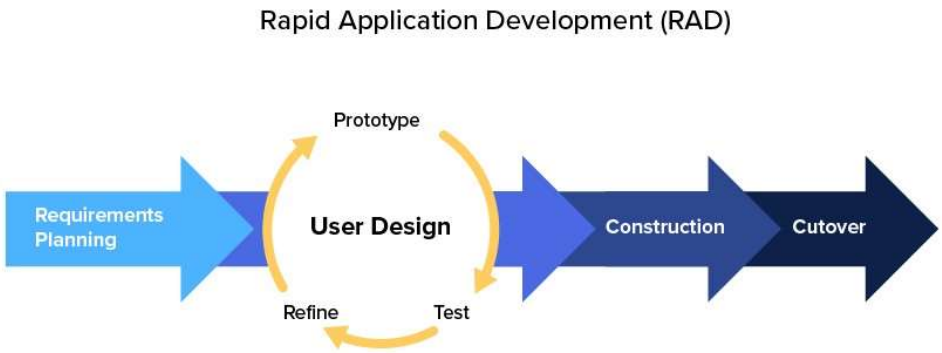
Metode *RAD* memiliki kelebihan dan juga kekurangan jika dibandingkan dengan metode pengembangan lain:

Kelebihan penggunaan metode *RAD*:[14]

1. Dapat menggunakan kembali komponen yang ada (*reusable object*) sebelumnya sehingga tidak perlu membuat dari awal lagi
2. Integrasi proses yang lebih cepat dan efektif
3. Dapat menyesuaikan program dengan kebutuhan dan keinginan user dengan lebih mudah
4. Mengurangi kemungkinan terjadinya kesalahan atau error

Kekurangan penggunaan metode *RAD*[13]:

1. Memerlukan kolaborasi tim yang kuat dan kompeten
2. Memerlukan interaksi yang sering dan efektif antara pengguna dan *programmer*
3. Tidak sesuai untuk proyek yang berskala besar
4. Metode *RAD* tidak cocok untuk menciptakan satu aplikasi secara sekuensial, dan lebih baik digunakan untuk membuat modular yang terpisah.



Gambar 2. 1 Siklus Hidup *RAD* [15]

2.4 *Dashboard*

Dashboard adalah cara menampilkan berbagai jenis data visual di satu tempat. *Dashboard* berguna untuk menyampaikan informasi secara ringkas dan mudah untuk dimengerti. Seringkali, ini mencakup hal-hal seperti indikator kinerja utama atau metrik bisnis penting lainnya yang perlu dilihat dan dipahami oleh orang awam kepentingan secara sekilas[7].

Dashboard berguna di berbagai industri, *dashboard* dapat menyediakan berbagai jenis data dengan berbagai rentang tanggal untuk membantu pengguna memahami: apa yang terjadi, mengapa hal itu terjadi, apa yang mungkin terjadi, dan tindakan apa yang harus diambil sesuai dengan data yang ditampilkan. Dengan menggunakan visualisasi seperti tabel, grafik, dan bagan, orang lain yang tidak familiar dengan data dapat dengan cepat dan mudah memahami data dibandingkan dengan data yang ditampilkan dengan mentah yang biasanya digunakan dan dimengerti oleh pekerja internal[3][6].

Dashboard berbeda dengan laporan biasa, secara garis besar, laporan biasanya memiliki fokus yang lebih sempit. *Dashboard* berguna untuk memberikan pandangan mendalam ke dalam kumpulan data dan cenderung terkonsentrasi pada satu item atau peristiwa. Di sisi lain, *dashboard* cenderung memiliki tampilan dari

sejumlah besar data dan dibuat untuk menjawab satu pertanyaan. Pertanyaan itu bisa luas, seperti, "Bagaimana kinerja situs kami pada satu bulan terakhir?" Atau lebih spesifik, seperti, “Berapa unit yang telah berhasil dijual?” Atau mungkin sesuatu yang sedikit lebih sulit dilacak tanpa keahlian khusus, seperti, "apakah efisiensi kita secara keseluruhan meningkat?"[32].

Laporan visualisasi berguna untuk menyajikan data dalam bentuk visual agar lebih mudah dipahami dan dianalisis. Berbeda dengan *dashboard* yang tidak hanya memberi laporan visualisasi namun *dashboard* juga memberi ringkasan kumpulan data yang berbeda namun terkait, disajikan dengan cara yang membuat informasi terkait lebih mudah dipahami. *Dashboard* adalah jenis visualisasi data, dan sering kali menggunakan alat visualisasi umum seperti grafik, bagan, dan tabel[39].

Dashboard berfungsi dengan mengambil data dari berbagai sumber dan menggabungkannya sehingga pengguna awam dapat dengan lebih mudah membaca dan mengerti data yang disajikan. Dengan elemen interaktif, ini membantu pengguna awam yang menggunakan *dashboard* untuk lebih memahami poin – poin tertentu, dengan bantuan dari *dashboard* pengguna awam akan dapat mempelajari suatu topik khusus seperti angka penjualan, dan menjawab lebih banyak pertanyaan yang kemudian dapat digunakan untuk memberikan jawaban terhadap keputusan yang akan diambil[36].

Penggunaan utama *dashboard* adalah untuk menampilkan inti utama data dari berbagai sumber. *Dashboard* berguna untuk memantau, mengukur, dan menganalisa data yang relevan di area utama. Mereka mengambil data mentah dari banyak sumber dan menyajikannya sesuai dengan kebutuhan pengguna.

Dashboard berguna untuk mengukur banyak hal seperti: metrik pelanggan, informasi keuangan, informasi penjualan, analisis web, informasi manufaktur, data sumber daya manusia, Kinerja pemasaran, informasi *logistik*. *Dashboard* adalah alat pengumpul dan visualisasi data, *dashboard* digunakan oleh para profesional untuk menganalisis data kompleks atau pakar materi pelajaran untuk melacak atau menyajikan data pada pengguna awam. *Dashboard* juga digunakan dalam

presentasi untuk membantu memahami tantangan, peluang, di mana harus tumbuh dan membuat perubahan[34].

Dashboard penting karena menyediakan platform bagi orang-orang untuk membuat keputusan yang lebih baik dan informatif berdasarkan data yang disajikan. *Dashboard* bersifat dinamis, interaktif, dan dapat menampilkan data yang hampir *real-time*, *Dashboard* membantu pengguna untuk mendapatkan pemahaman yang lebih tepat pada saat itu juga[2].

Ada banyak manfaat menggunakan *dashboard*, jika pengguna memvisualisasikan dan memahami data. Pengguna akan mendapatkan pandangan yang lebih jelas tentang metrik data utama yang penting bagi organisasi. Manfaat utama lainnya juga termasuk:

1. Kejelasan data
2. Analitik data secara langsung
3. Peramalan yang lebih akurat
4. Presentasi yang lebih intuitif
5. Peningkatan aksesibilitas dan transparansi
6. Pengambilan keputusan dan pemecahan masalah yang lebih baik

2.5 *Framework*

Sebuah *framework* adalah kerangka kerja atau struktur yang telah dibuat sebelumnya yang menyediakan kerangka dasar untuk membangun atau menghasilkan perangkat lunak, aplikasi web, atau proyek lainnya. Sebuah *framework* selalu memiliki struktur dasar yang mencakup arsitektur aplikasi, struktur direktori, dan pola desain. *Framework* juga memiliki komponen siap pakai untuk mempermudah proses pengembangan aplikasi. Sebuah framework juga memiliki aturan untuk memastikan proses pengembangan dilakukan secara konsisten. . *Framework* membantu dalam pengembangan perangkat lunak dengan menyediakan seperangkat alat, komponen, aturan, dan standar yang dapat digunakan oleh pengembang untuk mempercepat proses pengembangan[31].

2.6 *DASH*

Dash adalah *framework* open source untuk membangun tampilan visualisasi data. Dirilis pada tahun 2017 sebagai *library Python*, dikembangkan untuk menyertakan implementasi untuk R, Julia, dan F#. *Dash* membantu data scientist membangun aplikasi web analitik tanpa memerlukan pengetahuan pengembangan web tingkat lanjut.

Lima fitur inti dari *framework Dash* adalah:

1. *Flask* yang berfungsi untuk menyediakan fungsionalitas server web
2. *React.js* yang berfungsi untuk membuat antarmuka pengguna halaman web
3. *Plotly.js* yang berfungsi untuk menampilkan *chart* yang digunakan dalam aplikasi
4. *Layouts* yang berfungsi untuk mengatur ukuran, posisi , warna, dsb tentang *graph* yang akan dimunculkan
5. *Callbacks* yang berfungsi untuk memungkinkan aplikasi untuk berinteraksi dengan pengguna

Dash dengan sendirinya akan mengintegrasikan ketiga teknologi yang sudah tercantum di atas jika program dibuat menggunakan *Python*, R, Julia, dsb. Jika aplikasi menggunakan *framework dash* seseorang dapat langsung mempelajari dash daripada harus mempelajari syntax kode lain yang akan memakan waktu lebih lama[30].

2.7 *React*

React adalah sebuah perpustakaan *JavaScript* yang sangat populer untuk membangun tampilan aplikasi interaktif dan dinamis untuk aplikasi web. *React* memungkinkan pengembang untuk membuat komponen UI yang dapat diatur secara hierarkis dan diubah secara efisien ketika halaman aplikasi berubah. *React* memiliki berbagai komponen terpisah yang dapat digunakan secara bersamaan untuk membangun tampilan aplikasi. *React* juga menyediakan penggunaan ekstensi JSX agar komponen react dapat mengimitasi penggunaan sintaks html. *React* juga

menggunakan fungsi *useState* yang berguna untuk mengecek variabel yang terdapat pada aplikasi dan melakukan tugas yang berbeda sesuai dengan variabel[31].

2.8 *Python*

Python adalah bahasa pemrograman *interpreted*, *object oriented* dengan semantik dinamis. Struktur data bawaan tingkat tinggi, dikombinasikan dengan pengetikan dinamis dan pengikatan dinamis, membuatnya sangat menarik untuk *Rapid Action Development* (*RAD*), serta untuk digunakan sebagai bahasa skrip atau lem untuk menghubungkan komponen yang ada bersama-sama. Sintaks *Python* yang sederhana dan mudah dipelajari menekankan kemudahan program *python* untuk dapat dibaca dan karenanya mengurangi biaya pemeliharaan program. *Python* mendukung modul dan paket, yang mendorong modularitas program dan penggunaan kembali kode. *Interpreter Python* dan perpustakaan standar yang luas tersedia dalam bentuk sumber atau biner tanpa biaya untuk semua platform utama, dan dapat didistribusikan secara bebas[22].

Seringkali, pengembang aplikasi menggunakan *Python* karena peningkatan produktivitas yang diberikannya. Sejak tidak ada langkah kompilasi, siklus edit-tes-debug sangat cepat. Melakukan *debug* program *Python* itu relatif mudah dibandingkan dengan compiler lain, bug atau input yang buruk tidak akan pernah menyebabkan kesalahan segmentasi. Sebaliknya, ketika juru bahasa menemukan kesalahan, itu menimbulkan *exception error*. *Debugger* tingkat *root* memungkinkan pemeriksaan variabel lokal dan *global*, evaluasi ekspresi arbitrer, menyetel breakpoint, menelusuri kode satu per satu, dan seterusnya. *Debugger* ditulis dengan *Python* itu sendiri, membuktikan kekuatan introspeksi *Python*. Di sisi lain, seringkali cara tercepat untuk melakukan *debug* program adalah menambahkan beberapa pernyataan cetak ke sumbernya: siklus edit-uji-*debug* cepat membuat pendekatan sederhana ini sangat efektif[29][30].

2.9 *Javascript*

Sebagian besar dari situs web modern menggunakan *javascript* untuk membuat halaman yang lebih interaktif besertakan untuk menangani fungsionalitas. Halaman web tradisional yang menggunakan html relatif lambat jika dibandingkan dengan

menggunakan *javascript*. Sistem *framework* akan menentukan proses pengerjaan sistem yang bertujuan untuk menentukan alur pengembangan suatu aplikasi besertakan juga mengurangi error yang dapat terjadi dalam pengembangan suatu aplikasi. Setiap *framework* memiliki kelebihan dan kekurangan masing – masing sehingga pemilihan *framework* yang tepat sangat vital dalam pengembangan aplikasi dengan baik.[31]

Tiga *framework* utama yang dapat digunakan dalam javascript adalah:

1. *Angular*: *Angular* dirancang untuk semua platform: web, web seluler, dan *desktop*. Kecepatan di web dan memungkinkan pengguna memiliki kendali atas skalabilitas sekaligus memenuhi kebutuhan data yang besar.
2. *React: React* adalah perpustakaan *JavaScript* untuk membangun antarmuka pengguna. *React* dirancang untuk meningkatkan pengembangan UI interaktif dengan mempermudah memperbarui tampilan ketika data berubah. Hal ini dilakukan dengan membagi tampilan menjadi komponen-komponen yang lebih kecil, yang dapat disusun untuk membuat UI yang kompleks.
3. *Vue: Vue* merupakan kerangka kerja javascript untuk membangun antarmuka pengguna. *Vue* dibangun di atas HTML, CSS, dan *JavaScript* untuk menyediakan model pemrograman berbasis komponen deklaratif yang membantu pengembangan antarmuka pengguna dengan kompleksitas apapun secara efisien.

2.10 *Black Box Testing*

Pengujian *black-box* bekerja dengan cara mengamati keluaran suatu aplikasi web ketika aplikasi web diberikan masukan tertentu.[27] Black box juga disebut sebagai pengujian fungsional berdasarkan spesifikasi dari klien dan penguji sistem tidak memiliki akses untuk ke kode program dari sistem tersebut. Metode *Blackbox Testing* merupakan salah satu metode yang mudah digunakan karena hanya memerlukan batas bawah dan batas atas dari data yang diharapkan[25][26].

2.11 Visualisasi *Graph*

Graph dapat digunakan untuk melakukan visualisasi dimana masing – masing *graph* berfungsi untuk menunjukkan data yang berbeda. Terdapat *line graph* yang berguna untuk menunjukkan *trend data* sesuai dengan waktu, terdapat juga *bar graph* yang berfungsi untuk menunjukkan data secara kategorik dengan nilai negatif. Terdapat juga *pie chart* yang dapat digunakan untuk menunjukkan data kategorik dimana *pie chart* akan menggunakan data dimana total persentasi harus berjumlah 100%[19][20].

2.12 Aplikasi berbasis web

Aplikasi web memberi pengguna kemampuan interaksi dan kolaborasi dalam berbagai cara sehingga membentuk komunitas virtual. Aplikasi web berfungsi secara dinamis dengan menghubungkan server besertakan dengan tampilan antarmuka aplikasi. Aplikasi web menggunakan koneksi dengan sisi server untuk dapat menjalankan fungsi aplikasi dengan baik, dimana jika tidak ada koneksi dengan server aplikasi tidak akan mendapatkan informasi untuk menjalankan fungsi aplikasi. Aplikasi berbasis web tidak memerlukan instalasi secara manual dikarenakan segala fungsionalitas dapat didapatkan dari server.[11][15]