DEWANTARA. J. Tech., Vol. 02, No. 01 (Mei 2021), Hal. 21-25 E ISSN : 2776-6764
P ISSN : 2774-2032

Aplikasi Dekstop Multi Platform untuk Redis Client Framework Electron JS dan React JS

Syamsumar Bustamin

Prodi Teknik Mesin, Akademi Teknologi Industri Dewantara Palopo,
Jalan K.H. Ahmad Razak 2 No. 7, Kota Palopo, Indonesia

**Email : syamsumarb@atidewantara.ac.id*

Abstrak

Aplikasi desktop adalah perangkat lunak yang berdiri sendiri yang dapat dijalankan di PC desktop atau laptop. Dikalangan pengguna, perangkat lunak aplikasi desktop ini sangat populer, karena memiliki beberapa kelebihan diantaranya: (1) aplikasi desktop dapat diakses secara offline, (2) aplikasi desktop memiliki respon yang cepat, (3) aplikasi desktop kaya akan user experience[1]. Perangkat lunak multi platform adalah perangkat lunak yang dapat berjalan pada lebih dari sistem operasi. Pengembangan aplikasi desktop multi platform memerlukan sumber daya yang banyak baik itu waktu, human resource maupun cost, karena setiap sistem operasi memiliki API yang berbeda untuk menjalankan sebuah aplikasi diatasnya, sehingga developer harus menggunakan tools dan code base yang berbeda untuk membuat aplikasi yang berjalan pada multi platform [5] Penelitian ini bertujuan untuk mengebangkan aplikasi desktop multi platform yang memiliki user experience yag sama dengan native desktop application menggunakan tools dan code base yang sama untuk setiap platform sehingga dapat mengurangi sumber daya yang dibutuhkan baik waktu, human resource dan cost dengan menggunakan ElectronJS dan ReactJS dengan studi kasus aplikasi Redis Client.

Kata Kunci : *Aplikasi Desktop, Multi Platform, Electronjs, Reactjs*

1. Latar Belakang

Aplikasi desktop adalah perangkat lunak yang berdiri sendiri yang dapat dijalankan di PC desktop atau laptop. Biasanya aplikasi desktop ini di kembangkan dengan menggunakan bahasa pemrograman seperti VB, C#, Java dan sebagianya. Dikalangan pengguna, perangkat lunak aplikasi desktop ini sangat populer, karena memeliki beberapa kelebihan diantaranya: (1) aplikasi desktop dapat diakses secara offline, (2) aplikasi desktop memiliki respon yang cepat, (3) aplikasi desktop kaya akan user experience [1]

Perangkat lunak multi platform adalah perangkat lunak yang dapat berjalan pada lebih dari arsitektur komputer dan sistem operasi. Mengembagkan aplikasi multi pltaform adalah sebuah tugas yang dapat memakan waktu yang sangat lama, karena setiap sistem operasi yang berbeda memiliki *Application Programming Interface (API) yang berbeda* [5]

2. Metodologi

Aplikasi berbasis desktop adalah perangkat lunak yang di-install pada satu komputer yang akan melakukan fungsi dan tugas tertentu yang dirancang untuknya. Namun, perangkat yang sama dapat mengakomodasi banyak pengguna dengan bantuan jaringan. Contohnya

adalah media player, word processor, dll [2]. Aplikasi desktop adalah perangkat lunak yang berdiri sendiri yang dapat dijalankan di PC desktop atau laptop. Biasanya aplikasi desktop ini di kembangkan dengan menggunakan bahasa pemrograman seperti VB, C#, Java dan sebagianya [1]. Aplikasi desktop memiliki beberapa kelebihan, diantaranya [6]

a. Data Security: Tidak seperti aplikasi web, semua data disimpan dalam sistem komputer pengguna, jadi tidak perlu khawatir akan diretas. Pengguna memiliki kontrol penuh atas aplikasi yang berdiri sendiri dan oleh karena itu memungkinkan perlindungan dari berbagai kerentanan.
b. Available Controls: Jika dibandingkan dengan aplikasi berbasis web, aplikasi desktop dilengkapi dengan sejumlah native control. Native control memungkinkan aplikasi mengakses perangkat keras dan komponen OS yang mendasarinya.
c. Flexibilty: aplikasi desktop dapat menggunakan perangkat keras komputer pengguna seperti port serial, kamera, port jaringan
d. Performance: Aplikasi desktop jauh lebih cepat dan lebih responsif jika dibandingkan dengan aplikasi web. Ini karena aplikasi web secara inheren

Harus berkomunikasi degan *web server* melalui akses *internet*. Di sisi lain, aplikasi *desktop*, jika dirancang dengan benar hanya akan memuat apa yang dibutuhkan saja. Jadi, aplikasi *desktop* menggunakan lebih sedikit memori dan lebih sedikit sumber daya, sehingga meningkatkan kinerja dan meningkatkan efisiensi aplikasi.

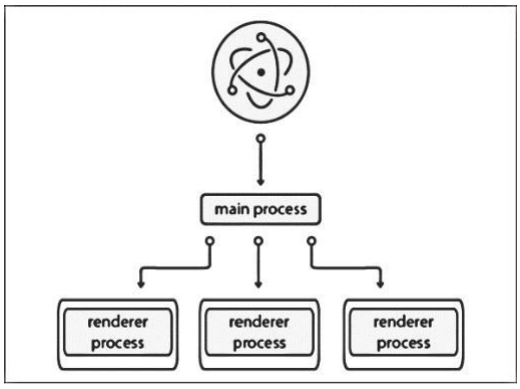
ReactJS

ReactJS adalah *view rendering JavaScript library* yang sangat bagus untuk menangani *JavaScript view component* yang berbasis *single page application* dengan menggunakan *data-driven UI* dengan cara yang efisien, dinamis, dan menarik secara visual. *ReactJS* akan secara otomatis mengelola pembaruan *UI* ketika ada perubahan data. ReactJS cukup fleksibel sehingga dapat digunakan dalam berbagai kapasitas dan konteks terlepas dari teknologi yang digunakan [4].

Sejak awal rilis pada tahun 2013, *ReactJS* telah diterima dengan baik oleh komunitas *JavasSript* diseluruh dunia, dan sejak itu sampai saat ini popularitasnya terus meningkat. *ReactJS* memiliki beberapa kelebihan dalam hal *performance*, *scalability*, *simplicity* dalam proses pengembangan baik dalam ukuran maupun kompleksitas data yang selalu berubah setiap waktu. ReactJS akan secara otomatis mengelola semua pembaruan UI Anda ketika data yang mendasarinya berubah, dan itu akan dilakukan dengan cepat. Itu adalah salah satu dari banyak hal yang disukai oleh para developer [1].

ElectronJS

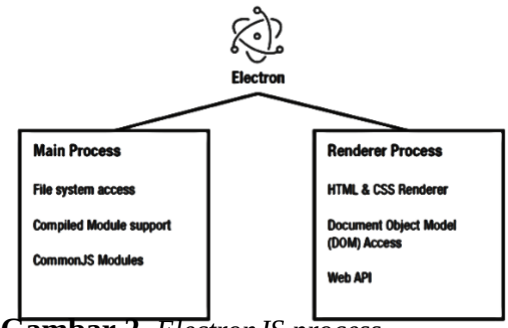
ElectronJS adalah *open source library* yang dikembangkan oleh *Github* untuk membangun aplikasi desktopmulti *platform* menggunakan *HTML*, *CSS* dan *JavaScript*. *ElectronJS* bekerja dengan cara menggabungkan *Chromium* dan *NodeJS* menjadi *single runtime* sehingga aplikasi dapat di-*package* menjadi *installer* untuk *Mac OS*, *Windows* dan *Linux*. Proyek *ElectronJS* dimulai pada tahun 2013 untuk membangun aplikasi *text editor* yang bernama *Atom*, kemudian pada tahun 2014 menjadi proyek *Open Source* [4].



Gambar 1 ElectronJS Procces

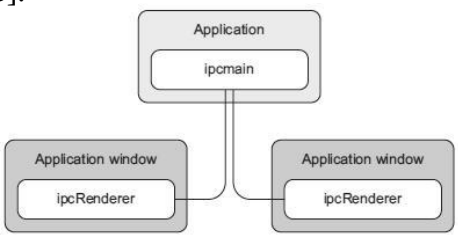
Aplikasi yang berbasis *ElectronJS* berjalan dalam dua proses yang berbeda, yaitu *Main Process* dan *Renderer Process*. Masing-masing dari dua *process* tersebut memiliki tanggung jawab yang spesifik dalam menjalankan aplikasi [3]

ElectronJS menyediakan koleksi *NodeJS Modules* yang bisa digunakan dalam aplikasi, modul-modul tersebut berjalan di proses tertentu. Sebagai contoh, akses untuk *API* dari sistem operasi hanya tersedia di *Main Process*, sedangkan akses ke *system's clipboard* tersedia di *Main* dan *Renderer Process* [4]. *Main Process* menangani *OS integration* (*file system access*, *compiled module support*, *commonJS module*). Sedangkan *Renderer Process* menangai *user interface* dan merespon *user event (HTML & CSS rednderer*, *Document Object Model Access*, *web API)* [2]



Gambar 2. *ElectronJS process responsibility*

Walaupun kedua proses tesebut tepisah dan terisolasi, namun keduanya bisa saling berkomikasi mengunakan *IpcMain* dan *IpcRenderer* (lihat Gbr. 2)[3].



Gambar 3. Komunikasi antar *process* pada *framework ElectronJS*

Sistem yang akan dikembangkan sebagai studi kasus dalam peneilitian ini adalah aplikasi *Redis Client* berbasis *desktop*. *Redis* adalah *NoSQL* database berbasis *key-value* dalam menyimpan data dan memiliki popularitas yang tinggi di komunitas pengembangan perangkat lunak [2]. Sistem yang akan dikembangkan ini memiliki beberapa fitur *basic CRUD (Create*, *Read*, *Update*, *Delete)* diantaranya:

- Autentikasi menggunakan *password*
- Memilih *database*
- Melihat list *key* yang sudah disimpan
- Melihat *value* dari *key* yang dipilih
- Menambah *key* dan *value* baru
- Mengubah *key* dan *value*
- Menghapus *key* dan *value*
- Menampilkan *key* dalam bentuk *raw text* atau format *JSON*.

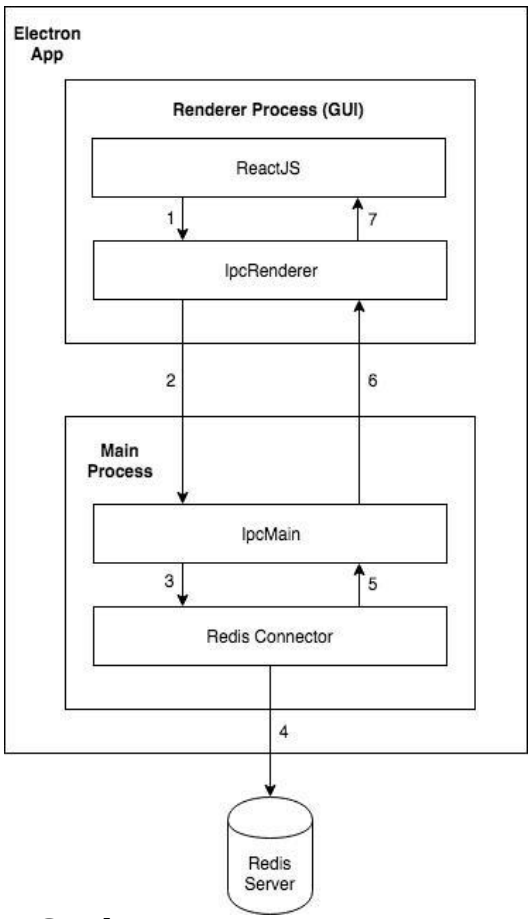
Arsitektur Sistem

Membangun sistem *Redis Client*, arsitektur perangkat lunak nya dibangun berdsarakan dua proses yang berbeda pada *ElectronJS* yaitu *Main Process* dan *Renderer Process*. Dibawah ini adalah arsitektur perangkat lunak dari sistem yang akan dibangun.

Dibawah ini penjelesan mengenai arsitektur sistem yang akan dikembang seperti pada Gambar 2:

- *GUI ReactJS* pada *renderer process* akan mengirimkan *request* ke *main process* melalui *IpcRenderer*.
- *IpcRenderer* akan meneruskan *request* dari *GUI* ke *main process* melalui *IpcMain*.
- *IpcMain* akan memanggil fungsi terkait pada *Redis Connector* sesuai dengan *request* yang dikirim dari *GUI*.

- *Redis Connector* berkomunikasi dengan *Redis Server* untuk melakukan *I/O process*.
- *Redis Connector* akan mengembalikan data respon sesuai dengan *request* yang dikirim dari *GUI*.
- *IpcMain* akan meneruskan data respon dari *Redis Connector* ke *GUI* melalui *IpcRenderer*.
- *IpcRenderer* akan mengirimkan data untuk di-*render* ke halaman tertentu yang mengirimkan *request* untuk diolah dan ditampilkan di *GUI*.



Gambar 4. Arsitektur aplikasi yang akan dikembangkan

Sistem ini dikembangkan berdasarkan pada *framework ElectronJS* dan *ReactJS* sehingga bisa berjalan di multi *platform*. Bahasa pemrograman yang digunakan untuk mengembangkan sistem adalah *HTML*, *CSS*, dan *JavaScript*. Pada dasaranya pembuatan atau pengembangan perangkat lunak dapat dilakukan di sistem operasi mana saja, baik *MacOS*, *Windows* ataupun *Linux*.

3. Hasil dan Pembahasan

Hasil dan pengujian sistem dilakukan pada tiga sistem operasi yang berbeda, yaitu MacOS High Sierra, Windows 10 dan Ubuntu 17.10. Aplikasi yang di-install dan dijalankan diatas sistem operasi tersebut dapat berjalan sesuai dengan fungsinya masing-masing. Dibawah ini adalah hasil pengujian Black Box dari sistem yang telah dikembangkan (aplikasi Redis Client).

Tabel 1. Hasil pengujian *Blackbox* dari aplikasi yang telah dikembangkan

Fitur	*MacOS High Sierra*	*Windows 10*	*Ubuntu 18.04*
Autentikasi menggunakan *password*	*OK*	*OK*	*OK*
Memilih *database*	*OK*	*OK*	*OK*
Melihat list *key* yang sudah disimpan	*OK*	*OK*	*OK*
Melihat *value* dari *key* yang dipilih	*OK*	*OK*	*OK*
Menambah *key* dan *value* baru	*OK*	*OK*	*OK*
Mengubah *key* dan *value*	*OK*	*OK*	*OK*
Menghapus *key* dan *value*	*OK*	*OK*	*OK*
Menampilkan *key* dalam bentuk *raw text* atau format *JSON*	*OK*	*OK*	*OK*

4. Kesimpulan

Dibawah ini adalah beberapa kesimpulan yang dapat diambil dari penelitian yang telah dilakukan:

1. Aplikasi desktop multi platform yang memiliki user experience yag sama dengan native desktop application dapat dikembangkan dengan menggunakan framework ElectronJS dan ReactJS yang berbasis teknologi web.
2. Pengembangan aplikasi desktop multi platform dengan menggunakan ElectronJS dan ReactJS dapat mengurangi sumber daya yang dibutuhkan karena developer hanya perlu membuat code base satu kali untuk bisa berjalan di multi platform.

DaftarPustaka

[1] G. Pankaj, K. Piyush, Sandeep, W. Saksham, Y. Vishal. Operating System. International Journal of Computer Science and Information Technology Research, vol. 2, 2014.
[2] Maxwell, D. D. S., Hugo, L. T.. Redis Essentials, Packt Publishing, 2015.
[3] Julie, C. M.. Sams Teach Yourself HTML, CSS, and Java Script All in One, Pearson Education, 2012
[4] ElectronJS, About Electron, ElectronJS.com, https:/electronjs.org/docs/tutorial/about (diakses: 07 Maret 2021)
[5] Wikipedia, Cross Platform, https://en.wikipedia.org/wiki/Cross-platform, (diakses: 01 Maret 2021)
[6] A. D. Midhun, Using Electron for Cross Platform Desktop Application Development : An Introduction, Cabotsolutions.com, https://www.cabotsolutions.com/2017/11/using-electron-for-cross-platform-desktop-application-development-an-introduction, (diakses: 01 Maret 2021).