## *[Ninjabot]*

#### *[Derwin Lim Chin Wee]*

#### *[2427221L]*

## Proposal

### Motivation

Technological advancements have allowed more devices to access the Web page on behalf of its users to obtain information. However, identification techniques that can properly identify Web Widgets are rather inaccurate if the developer of the Web page does not use any WAI-ARIA. Most of the identification methods are usually looking out for keywords. Therefore, by tracing from the source of the Web Page, the accuracy of identification of the Web Widget could be improved.

### Aims

This project will develop a Web robot (bot) to automatically scrape the Web page so that Web Widgets can be identified using Machine Learning techniques. The bot is configurable to scrape pages that intentionally orchestrated content for specific user-agent in a discrete manner. The effectiveness of the project in successful identification of Web Widget will be experimentally validated based on the amount of characteristic spotted in a widget.

## Progress

* The project will be implemented in Python and JavaScript.
* Software architecture outlined in the appendix Figure 1.0
* Background research conducted on:
  + Definition of the Web, Web site, Web page, Web Widget and Web application
  + The intent of the Web model Creator (Tim Berners-Lee)
  + Purpose of the WCAG 1.0 and 2.0
  + Web UI Design patterns
  + Definition and Framework of ECMAScript
* Initial version of bookmarklet, which access the webpage and analyze the Document Object Modem (DOM)

## Problems and risks

### Problems

* Implemented Scraper to replicating Web pages as near identical as possible.
* Attempts to bypass Browser Cross-Origin Resource Sharing (CORS) to replicate the scrape site as close as possible.
* Implemented bookmarklet to inject JavaScript to obtain the required resource from website.
* Understanding the ECMAScript Architecture to retrace existing of Web Widget in a Web Page. Still looking into how ECMAScript functions can be correlated back to framework scripts.

### Risks

* Many widget component to identify. **Mitigation:** will narrow down to one to two widgets.
* Web page Sign-in authentication. **Mitigation:** Any form of Sign-in authentication will be avoided in this project.
* Unclear how to evaluate success of Widget Identification. **Mitigation**: will do background research to investigate how success of Widget Identification has been performed in the research literature review.
* How many signs must the code have in order to ensure that the Widget is clearly identify? **Mitigation**: will take roughly about 70% to 75% of the signs.
* Retracing of code and ensuring the accuracy of the Identified component. No clear mitigation available.

## Plan

**Semester 2**

* Week 1-2: Research tracing JavaScript to identify Web Widgets.
* Week 2-3: Integrate WAI-Aria requirements to project once Widget is identified.
* Week 3-5: Integrate tracing with Machine learning.
* Week 5-6: Merge the Integration with scraper.
* Week 6: Research on how to evaluate the system performance.
* Week 7-9: Machine Learning Training on Identification of Web Widget components
* Week 9: Evaluate test experiments run
* Week 8-10: Write up on report and poster

## Appendix

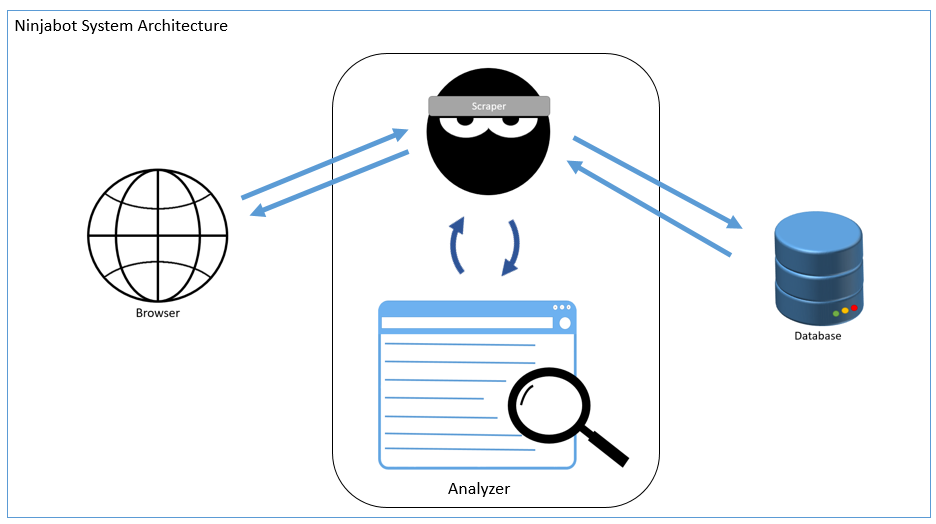


Figure 1.0

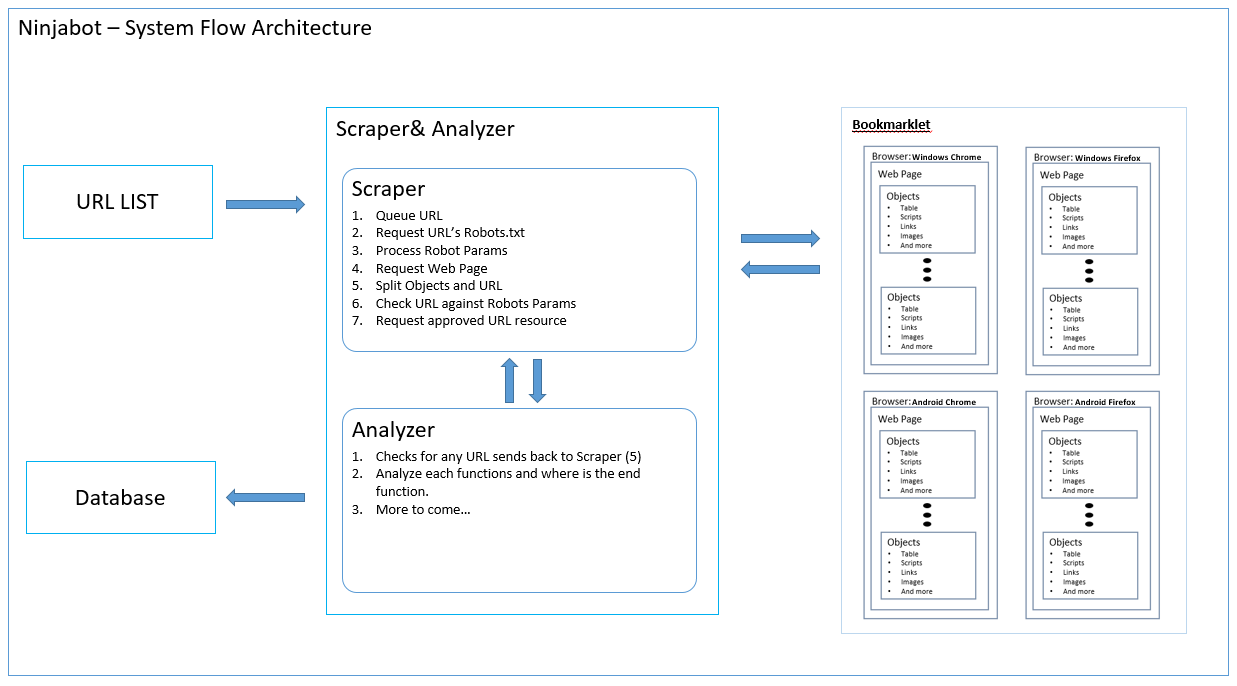


Figure 2.0