ANOTODE

The Web Annotator .

LLD - Browser Extension

Revision 3.0

CS Group 1

Authors:
Avi (201451070)

Reviewed by:
Manohar (201451024)

Revision Table

Revision	Author	Reviewer	Revision Date	Revision Tracking Notes
1	Avi	Manohar	5-10-16	Increment 1. No changes needed
2	Avi	Manohar	17-10-16	Increment 2. Add technique for error handling
3	Avi	Manohar	1-11-16	Increment 3. No changes needed

Contents

1 Introduction

1.1 Purpose

This document holds the low level design specification for the browser extension component of the project Anotode. It is supposed to be followed during the coding phase and will be used as a reference for developing the extension part of the project.

1.2 Document Convention

This document uses the same standards as our all other documents and the specifications for the same has been defined in the Document Convention document. Apart from those defined there, in this document we are using some conventions which are as follows -

- Extension means Chrome Extension.

1.3 Intended Audience and Reading Suggestions

The audience of this document is assumed to be the development team and whosoever who is part of development of the browser extension and in some cases, the entire project. It is recommended that anyone reading this document must have prior knowledge of how browser extensions work. An experience in developing browser extensions would come really handy.

1.4 References

Various references were used in preparing this document. A non-exhaustive list of them can be listed as follows -

- Google Chrome Developer Reference
  https://developer.chrome.com/extensions
- Tutsplus Chrome Extension Getting Started Guide
  https://code.tutsplus.com/tutorials/developing-google-chrome-extensions–net-33076
- Sitepoint 10 minute tutorial on getting into browser extension
  https://www.sitepoint.com/create-chrome-extension-10-minutes-flat/
- Firefox Web extensions reference
  https://developer.mozilla.org/en-US/Add-ons/WebExtensions
- Firefox Addon SDK docs
  https://developer.mozilla.org/en-US/Add-ons/SDK

2 Design Overview

Browser extension consists of 3 parts .

- Content scripts
- Background pages
- Popup

Content scripts They are responsible for highlighting the webpage in case of highlight or re-highlighting in case of re-highlight.

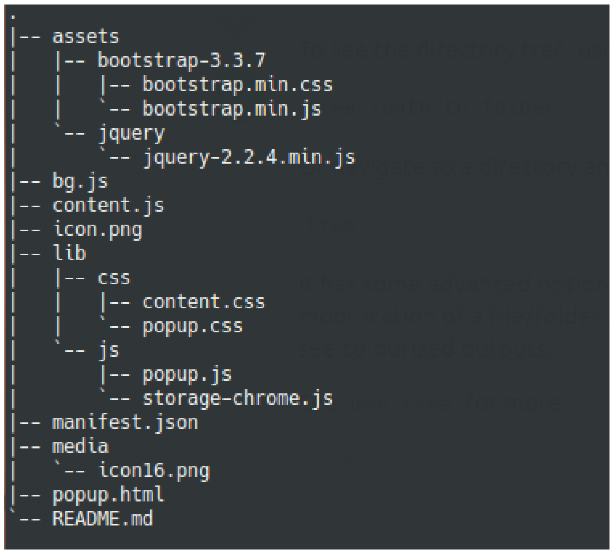
Background pages They are responsible for loading data from the server and saving it back on the server.

Popup It is responsible for showing the GUI that allows user to login/signup and change settings.

3 Design Description

3.1 Codebase Structure

Codebase should be organized as the following mockup tree structure.



```
.
|-- assets
|   |-- bootstrap-3.3.7
|   |   |-- bootstrap.min.css
|   |   `-- bootstrap.min.js
|   `-- jquery
|       `-- jquery-2.2.4.min.js
|-- bg.js
|-- content.js
|-- icon.png
|-- lib
|   |-- css
|   |   |-- content.css
|   |   `-- popup.css
|   `-- js
|       |-- popup.js
|       `-- storage-chrome.js
|-- manifest.json
|-- media
|   `-- icon16.png
|-- popup.html
`-- README.md
```

- Third party use-as-is libraries should be stored in the assets folder.
- bg.js and content.js are the background pages and content script respectively.

- popup.html is the Popup view.
- media folder should store all binary media files required by the extension to work.
- README.md should be created to include basic information about the project component.

3.2 User Interface

User Interface mockups for the extension can be seen in the prototype design documents. Please refer to it.

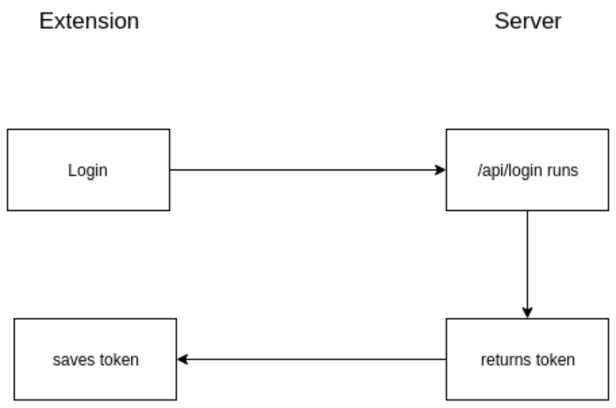
3.3 Class Diagrams

N/A
Browser Extension doesn't require classes or class-like entities. It is based on event handlers which has been discussed in the next section.

3.4 Event/Message flow diagrams

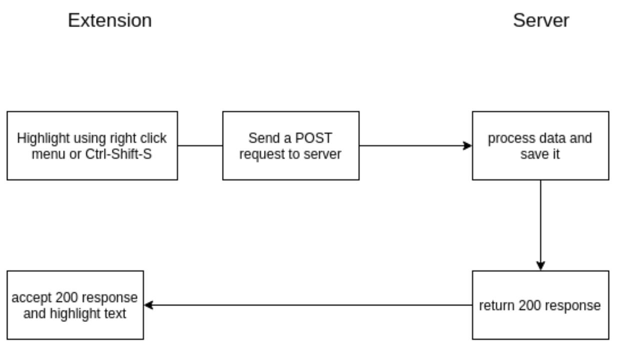
3.4.1 Login flow

email and password is sent to the server which returns with the authentication token in case the combination is valid.



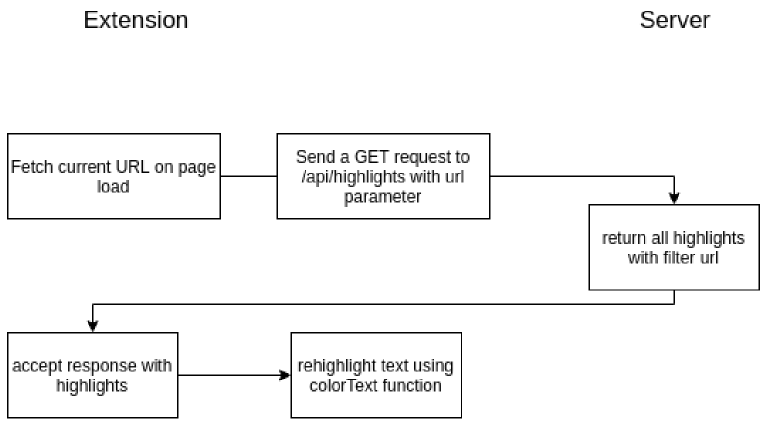
3.4.2 Highlight flow

User can highlight text using the right click menu or using shortcut $Ctrl - Shift - S$. The data is sent to the server which returns with 200 response in case of success.


Extension
Server
Highlight using right click menu or Ctrl-Shift-S
Send a POST request to server
process data and save it
return 200 response
accept 200 response and highlight text

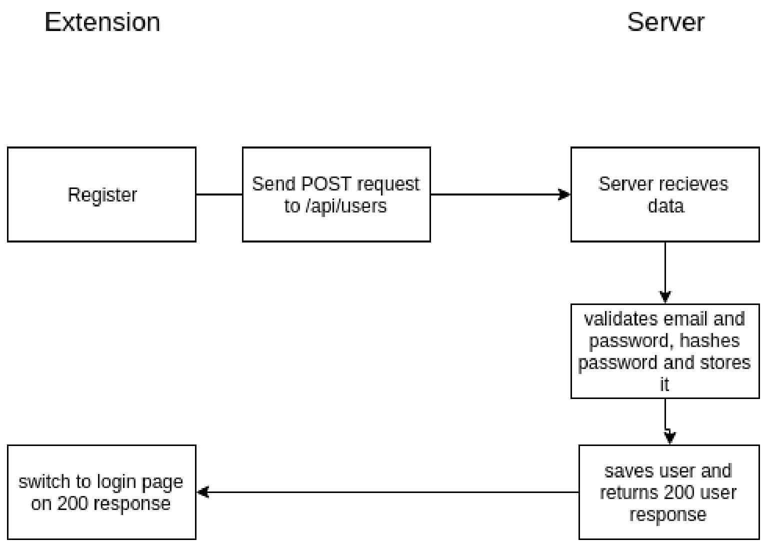
3.4.3 Re-highlighting

To rehighlight a page, extension sends a request to server to fetch highlights for that page. The server responds with the list of highlights in that page and the chrome extension highlights it again.


Extension
Server
Fetch current URL on page load
Send a GET request to /api/highlights with url parameter
return all highlights with filter url
accept response with highlights
rehighlight text using colorText function

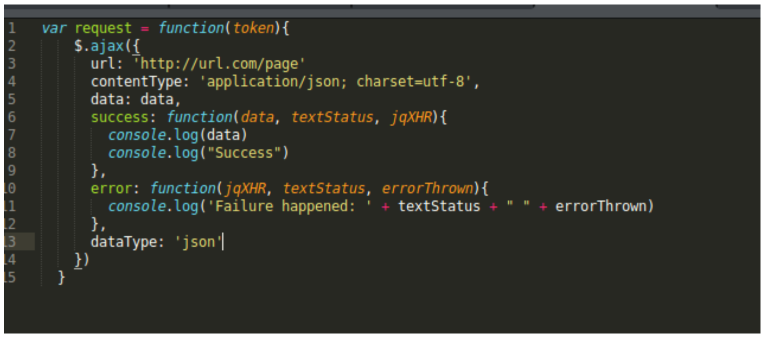
3.4.4 Register flow

email, password and username is sent to the server which returns with 200 response of user created if everything went fine.


Extension
Server
Register
Send POST request to /api/users
Server recieves data
validates email and password, hashes password and stores it
saves user and returns 200 user response
switch to login page on 200 response

3.5 Error/Exception Handling

Errors in browser extension can mainly happen in server requests. To handle such errors, we should use *success* and *error* attributes of JQuery. The operation will be aborted in case of error.



```
var request = function(token){
    $.ajax({
      url: 'http://url.com/page'
      contentType: 'application/json; charset=utf-8',
      data: data,
      success: function(data, textStatus, jqXHR){
        console.log(data)
        console.log("Success")
      },
      error: function(jqXHR, textStatus, errorThrown){
        console.log('Failure happened: ' + textStatus + " " + errorThrown)
      },
      dataType: 'json'
    })
  }
```

4 Configuration

This section lists the configurations that need to be done in order to get the system running.

4.1 Adapter / Connector Configuration

N/A

4.2 Application Configuration

4.2.1 BaseURL

The BaseURL of the backend server needs to configured to use the server. Right now, the url is $https : //anotode.herokuapp.com$.

4.3 Third Party tool/libraries Configuration

The following libraries are needed by the extension component of Anotode.

- JQuery v2.2.4 and above
- Bootstrap v3.3.7 and above

Keep them in the assets folder as already shown in the codebase structure section (3.1).

5 Customization

- Version should be made customizable from manifest.json by using a $version$ key in the file.
- Product icon should be customizable. For icon, use a icon.png in project root. This icon should be customizable as the icon can change when the product matures.
- Context menu icon needs to be changeable too. It can be changed from media/icon16.png file.

6 Assumptions

The following assumptions were made while writing this document.

- It is assumed that the developer/coder is experienced with JQuery AJAX requests as that have not been specifically specified in the document.
- The developer should have some prior experience of developing browser extensions as it is not a getting started level document.

7 Dependencies

The following are the dependencies for building the software mentioned in this document.

- Chrome API
- JQuery

- Bootstrap
- HTML5 storage API

8 Glossary

AJAX Asynchronous JavaScript and XML