TabX is a typing assistant program. It is an accessibility tool that functions primarily as a Google Chrome extension. It's targeted at those whose native language is not English or other users who don't have much computer experience. The purpose of the product is to make the typing process easier.

Rather, we expect to see a reasonably realistic plan in this report, and we expect you to follow-up on whether you reached your goals or not.

**Overarching Goal:**

Central Idea: We wanted to provide create a writing assistance plugin that would be able to:

o provide the user with potential suggestions based on partial strings

o predict next likely word based on the partial text, usually an incomplete sentence

o abbreviation expansion in the text

o Word correction

o Adhering to Chrome API as extension

o Creating Clean Interface for seamless frontend and backend

o Optimize the backend models

**What the Front End currently achieved:**

* Understood and used the Chrome API
* created functionality needed for the mock UI
* Tested the mock front end system on simple html
  + Tested on general html pages
* Behavior Testing for Front End
* Integrated backend current word prediction model into frontend
  + Regression Testing in Place
* Investigating Selenium UI automation
  + Interactive HTML UI automation

**Next steps for Front End:**

* Then create second mock UI for next feature beyond word completion
* Create the Interface to define/differentiate front end from back-end
* Test the behaviors of the second mock UI via Jasmine
* Repeat the process for the remaining 3 Models
* Integrate additional features

**What the Back End currently achieved:**

* **BOTH MODELS FULFILL THE CURRENT WORD PREDICTION**
  + Created Trie from scratch that provides suggestions based on partial string
  + Created Levenshtein algorithm from scratch that can also do word prediction like trie
* **MODEL FOR NEXT WORD PREDICTION** 
  + Have an Keras/Tensorflow RNN
  + it’s currently unspecified beyond being trained on vectorized text data

**Next steps for Back End:**

* Create more backend predictive models based on additional features
  + Word Correction Backend models
  + Abbreviation Expansion Backend models
  + Additional models based on chosen features to implement

**Features**

* Profanity Filter setting (if someone types profanity, don’t suggest words for it)
* admin powers to adjust all these settings
* Customizable Word Set
* Option to forget Searches (for public computer usage)
* More UI-friendly autocorrect
* Abbreviation expansion model
* Changing display

**Competitors:**

* Name: iOS and Android keyboard word suggestions
* Con: it is only on phone
* Name: Google Autocomplete extensions
* Con: all of them autocomplete webforms, not actual typing
* Name: Co: Writer Universal
* Con: does the same thing we want to do and more. However, based on the reviews, it is not free, does not work well, also makes the browsing slow since it has a lot of features running real-time.
* ability to change font size, color, overall display, etc. in order to improve accessibility and user experience
* setting to not provide suggestions for profanity, given the nature of the library and its availability to all ages and different types of people
* setting to use a word suggestion set customized to library (frequently searched terms)
* ability to forget previous searches & not customize to a specific user, given that the computers are for public use by multiple individuals
* administrative abilities so that only authorized users can adjust these settings