

CPSC 311: Project Proposal (Team T3)

Name: Ashley Lee
Ugrad ID: k7y8
Student ID: 34959122
Email: alee238@hotmail.com

Name: Min Seok Ray Roh
Ugrad ID: l9x9a
Student ID: 33737123
Email: msroh94@gmail.com

Name: Michelle Wilson
Ugrad ID: a2d0b
Student ID: 60855087
Email: m888wilson@gmail.com

Name: Cecile Leung
Ugrad ID: o8r6
Student ID: 90600974
Email: cecilephl@gmail.com

Name: Norman Sue
Ugrad ID: h0e9
Student ID: 20396131
Email normansue3@gmail.com

1 Overview

Project Type: 3

Project Topic: TypeScript

Target Completion Level: 100%

Our group will research features of the TypeScript language and use the knowledge gained to build a substantial web application that generates “linkbait articles”.

Additionally, we will compare our TypeScript program against an analogous JavaScript version of the program, written from scratch, to showcase the advantages of using TypeScript over JavaScript.

We are coordinating online communication via [Slack](#) and documentation via [GitHub](#).

2 Background Research Report (project-background)

For our background research report, we plan to do further research based on the topics listed in Microsoft’s official TypeScript GitHub [wiki page](#) and have divvied up the topics as follows:

Michelle: Basic Types, Interfaces, Classes

Cecile: Generics, Mixins

Ashley: Namespaces and Modules, Functions

Ray: Type Inference, Writing Type Definition Files

Norman: Type Compatibility, How TypeScript preserves runtime behavior of all JavaScript code, How TS aligns with current and future ECMAScript proposals

After doing individual reading, we individually type up our findings in separate plaintext files, and then meet on November 13 to combine the content and format it to produce the background research report.

We'll include snippets of code executable TypeScript code to demonstrate our understanding of the language features that we're researching.

Starting point documents for our research:

- Tutorial: [Learn TypeScript in Y Minutes](#)
- Spec: [TypeScript Specification](#)
- Docs: [TypeScript Design Goals](#)
- Docs: [TypeScript GitHub wiki](#)
- Docs: [TypeScript Handbook](#)
- Libraries: [DefinitelyTyped](#)
- Blog: [Making .NET Developers Comfortable with JavaScript](#) by Shane Boyer
- Book: [TypeScript Essentials](#) by Christopher Nance
- Book: [Mastering TypeScript](#) by Nathan Rozentals
- Book: [Pro TypeScript: Application-Scale JavaScript Development](#) by Steve Fenton

3 Proof-of-Concept and Plan (project-plan-proof)

After writing the background research report, we will write and implement the necessary TypeScript modules and classes with the associated type definition files (`.d.ts` files) to showcase all the features of the language that we want to use in the 100% deliverable web application.

Simultaneously, we will also write the less-type-safe JavaScript analogue classes for the ones we write in TypeScript. Since TypeScript compiles to JavaScript, this means we will have two resulting sets of JavaScript files: one that is compiled from the TypeScript we will write, and another that is written from scratch without the type safeness and advanced features that TypeScript provides.

In order to know what classes and modules to build for the proof-of-concept, we will be deciding upon the required functionalities of the web application by:

1. Documenting user stories.
2. Breaking down user stories into individual tasks.
3. Researching any required external libraries and APIs needed.
4. Delegating tasks amongst ourselves.
5. Drawing any necessary UML class digrams.

We will also need to plan out the following:

- what browsers we will support
- what the visual layout of our web application will be
- server-side API routes needed
- server-side database schema
- deployment platform (e.g. [Amazon EC2](#) or [Linode](#))

4 Poster

Our poster will have snippets of code from our proof-of-concept classes comparing JavaScript and TypeScript, using the features that we've researched to show how they're beneficial to a potential TypeScript programmer.

5 Final Project (`project-final`)

Using the TypeScript modules and class files built for our proof-of-concept, we will need to flesh out the details required for implementing the following remaining parts of the web application for generating linkbait articles.

5.1 Client-side implementation

Tasks we will need to complete for the client-side implementation:

- writing the HTML files (most likely with the help of a templating library such as [Jade](#))
- styling the HTML with CSS (most likely with a CSS framework such as [Twitter Bootstrap's CSS](#) or [Pure.css](#))
- programming the dynamic interaction with the DOM elements using jQuery in conjunction with our modules written from the proof-of-concept

All of our client-side code will be written using TypeScript.

5.2 Server-side implementation

Our server-side code will be kept to a minimal, mainly to only serve static assets, the compiled client-side TypeScript application, and API routes for accessing database data.

We plan on using the following technologies (along with any necessary `npm` modules):

- [Node.js](#)
- [Express](#)
- [MySQL](#)

All of our server-side code will also be written using TypeScript.

We will determine more specific implementation requirements for the final project within the proof-of-concept and planning phase.