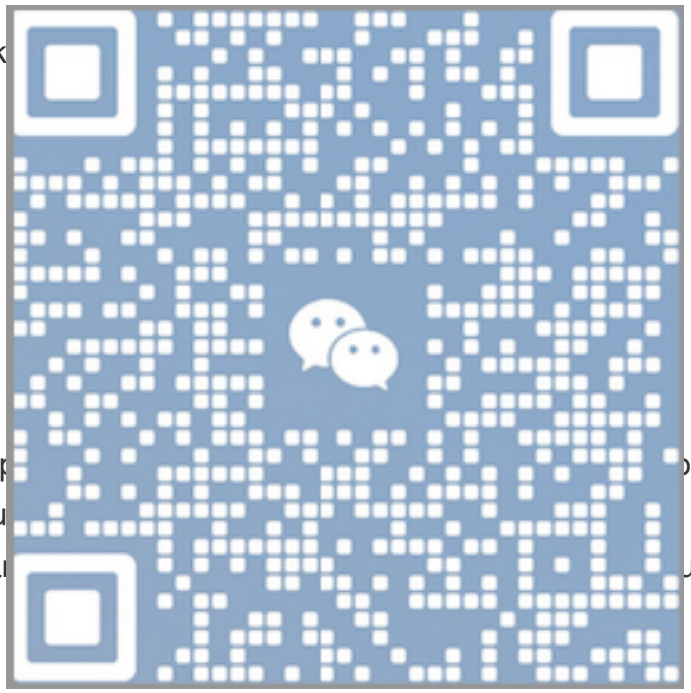
FIT2102 Programming Paradigms 2024 - Assignment 2: Markdown to HTML

Due Date

- Friday, 18th October, 11:55 pm

Weighting

- 30% of your final mark



Interview

- SWOTVAC + Week 13

Overview

- Students will create a p on using functional programming techniqu
- Demonstrate understa umentation of design decisions.

Submission Instructions

- Submit a zipped file named `<studentNo>_<name>.zip` which extracts to a folder named `<studentNo>_<name>` .
  - Must contain all code, report (named `<studentNo>_<name>.pdf`).
  - No additional Haskell libraries (except for testing).
  - Run `stack clean --full` before zipping.
  - Don't submit `node_modules` or `.git` folder.
  - Ensure code executes properly.
- Marking process:
  - Extract zip.
  - Copy submission folder contents.

- Execute `stack build`, `stack test`, `stack exec main/npm run dev` for front end.

Late Submissions

- Penalised at 5% per calendar day, rounded up.
- After seven days, receive zero marks and no feedback.

Table of Contents

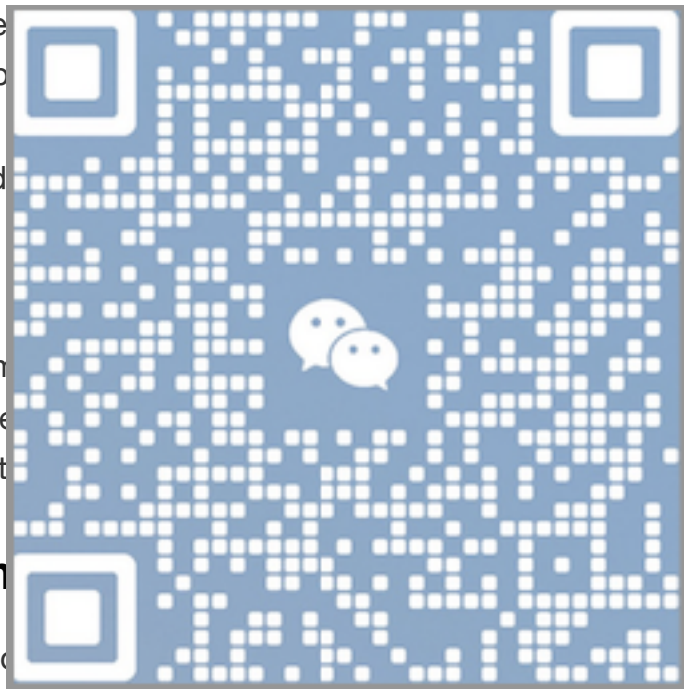
8. Correctness
9. Changelog

Minimum Requirements

- Parse up to and including code blocks (with high code quality and good report) for a passing grade.
- Higher mark requires parsing more difficult data structures and modifying the HTML page.

Introduction

- Develop a transpiler to convert Markdown to HTML in Haskell.
- Web page sends Markdown to Haskell backend server, which converts and returns HTML.
- Use materials from pre
- Assignment split into p doing Part A and B in tandem.
- Parse based on Markd



Goals / Learning

- Use functional progran
- Understand and use ke
- Apply Haskell and FP t

Scope of assignm

- Parse expression into
- Not required to render Markdown or HTML strings.

Exercises (24 marks)

Part A: (12 marks): Parsing Markdown

- Parse markdown string into Algebraic Data Type (ADT).
- Define own ADT and functions to parse requirements.
- ADT should have enough info for HTML conversion.
- Add `deriving Show` to ADT and custom types.
- Export `markdownParser :: Parser ADT` and `convertADTHTML :: ADT -> String`.
- Example scripts:
  - Run `stack test` to parse Markdown and save output.

- Use `npm run dev` with `stack run main` to test in real-time.

Part A - Details

Aside - Text Modifiers (2 marks)

- Italic Text: `_italics_`
- Bold Text: `**bold**`
- Strikethrough: `~~strikethrough~~`
- Link: `[link text](URL)`
- Inline Code: `` `code` ``
- Footnotes: `[^ℤ+]` (e.g., `[^1]`, `[^2]`)
  - No whitespace inside `[` and `]`.
  - No nested modifiers.
  - Text inside modifie

Images (0.5 marks)

- `![Alt Text](URL "Ca`
  - Alt Text, URL, Cap
  - At least one non-n ext.
  - No spaces after `!`

Footnote References (0.5

- `[^2-^]: text` (ignore

Free Text (1 mark)

- Any text not following

Headings (1 mark)

- `# Heading 1` to `###### Heading 6`
- Alternative syntax: `Heading 1` followed by `======` for level 1, `Heading 2` followed by `------` for level 2.
- Headings may have modifiers.

Blockquotes (1 mark)

- Start with `>` at the beginning of a line.
- Ignore leading whitespace after `>` and before text.
- Text inside can have modifiers.
- No nested block quotes.

Code (1 mark)

- Starts and ends with three backticks (```), with optional language identifier.
- Code block should not consider text modifiers.

Ordered Lists (2 marks)

- Starts with number 1, followed by `.` and at least one whitespace.
- Can have sublists with 4 spaces before each item.
- Items may contain text modifiers.
- Don't handle unordered lists.

Tables (3 marks)

- Use pipes `|` to separate columns and dashes `-` to separate header and content rows.
- Compulsory beginning
- Each row must have th
- Cells may contain text space.

Part B: (6 marks): HT

- Convert ADT to HTML
- Indent HTML with 4 sp
- HTML must be a self-c

Part B - Details

Text Modifiers (1 mark)

- Italics: `<em>italics</`
- Bold: `<strong>bold</strong>`
- Strikethrough: `<del>strikethrough</del>`
- Link: `<a href="URL">link text</a>`
- Inline Code: `<code>code</code>`
- Footnotes: `<sup><a id="fn1ref" href="#fn1">1</a></sup>`

Images (0.5 marks)

- `<img src="URL" alt="Alt Text" title="Caption Text">`

Footnote References (0.5 marks)

- `<p id="fn1">My reference.</p>`