$\int$n2gr8 - Development Plan

Group 11

Anthony Guirguis (guirguia)
Hoda Mohammadi (mohamh8)
Mikolaj Hrycko (hryckom)

September 29, 2017

Team meeting plan

The team will be meeting every Monday at 7PM at Thode Library and the role of the meeting leader and note taker will rotate between members during each meeting. At each meeting the progress of the tasks that have been started by each member will be discussed to assure the deadlines can be met and the project is moving in the right direction. Meeting minutes will be used to keep track of the meeting and assign roles to each member.

Team communication plan

The team will be using Git to share and commit changes to codes and documents, as well as submitting each milestone. A milestone will be considered complete when all git issues related to that milestone have been closed. A facebook group chat has been made to communicate ideas and discuss issues and changes in a faster, more efficient way. Some of the documentations will also be edited in Google Docs to allow all members to edit the document simultaneously.

Team member roles

The roles and responsibilities of the members will alternate every week to ensure equal contribution from each member.

Member	Role	Expertise
Anthony	Leader	Technology, Documentation
Hoda	Scribe	Latex, Technology
Mikolaj	Contributor	Git, Documentation

Git work flow plan

Branching Scheme

For this project, the git workflow will follow a feature-release-master branch scheme. As many features are not dependant on one another, there will be many feature-branches that will be worked on at once. After a feature-branch is tested and completed it will be pushed to a release branch. Once a prescribed set of features are on the release branch, the release-branch will be tested as a whole and then pushed onto the master branch.

Issue Tracking

Gits issue tracker will be used to track completed and yet to be completed features. The issues will be linked to specific milestones, which will indicate when a release branch is ready to be merged with master.

Proof of concept demonstration plan

Most Significant Risk

The most significant risk of the project is whether we are able to implement certain parts of the code itself (mentioned below) and linking the code to the website and properly getting all of the inputs to be used, as well as outputting correctly.

Will a part of the implementation be difficult?

The most difficult part of implementation will be whether we will find a way to be able to implement code that can identify the particular solving method that needs to be applied to the users input. Along with this, we will have to be able to parse the users mathematical equation inputs in a way that will make it easy for other functions to use the data to solve the problem at hand. If these parts of the implementation are met then no other major roadblocks are seen in project implementation.

Will testing be difficult?

Testing will not be too difficult, we just need to familiarize ourselves with the testing framework. The Mocha testing framework is easy to use, and using techniques we learned in previous classes such as creating unit tests can be used to test our code.

Is a required library difficult to install?

No, for now no external library is required for project implementation.

Will portability be a concern?

Portability will not be a concern as this is a browser based application. It will be able to be accessed through any platform that has a web browser.

Technology

Programming Language/IDE

The project will be using Javascript as the main programming language. React, which is a Javascript library will be used for building the user interface on the website. Eclipse Neon with will be used as the IDE where we will run our javascript code.

Testing Framework

For the testing framework, we will use Mocha which runs on Node.js. With this, we can make unit tests and run them in the terminal using npm.

Document generation

The documentation will all be written in latex and pdflatex will be used to generate the pdf.

Coding style

General

- camelCase for identifier names (variables and functions)
- All names start with a letter
- Put spaces around operators (= + - * /), and after commas
- Tab for indentation of code blocks
- End a simple statement with a semicolon
- line length less than 100
- Constants (like PI) written in UPPERCASE

For complex statements

- Put the opening bracket at the end of the first line
- Put the closing bracket on a new line, without leading spaces

Project schedule

A detailed schedule of the project (Gantt Project) can be found here.

Project review