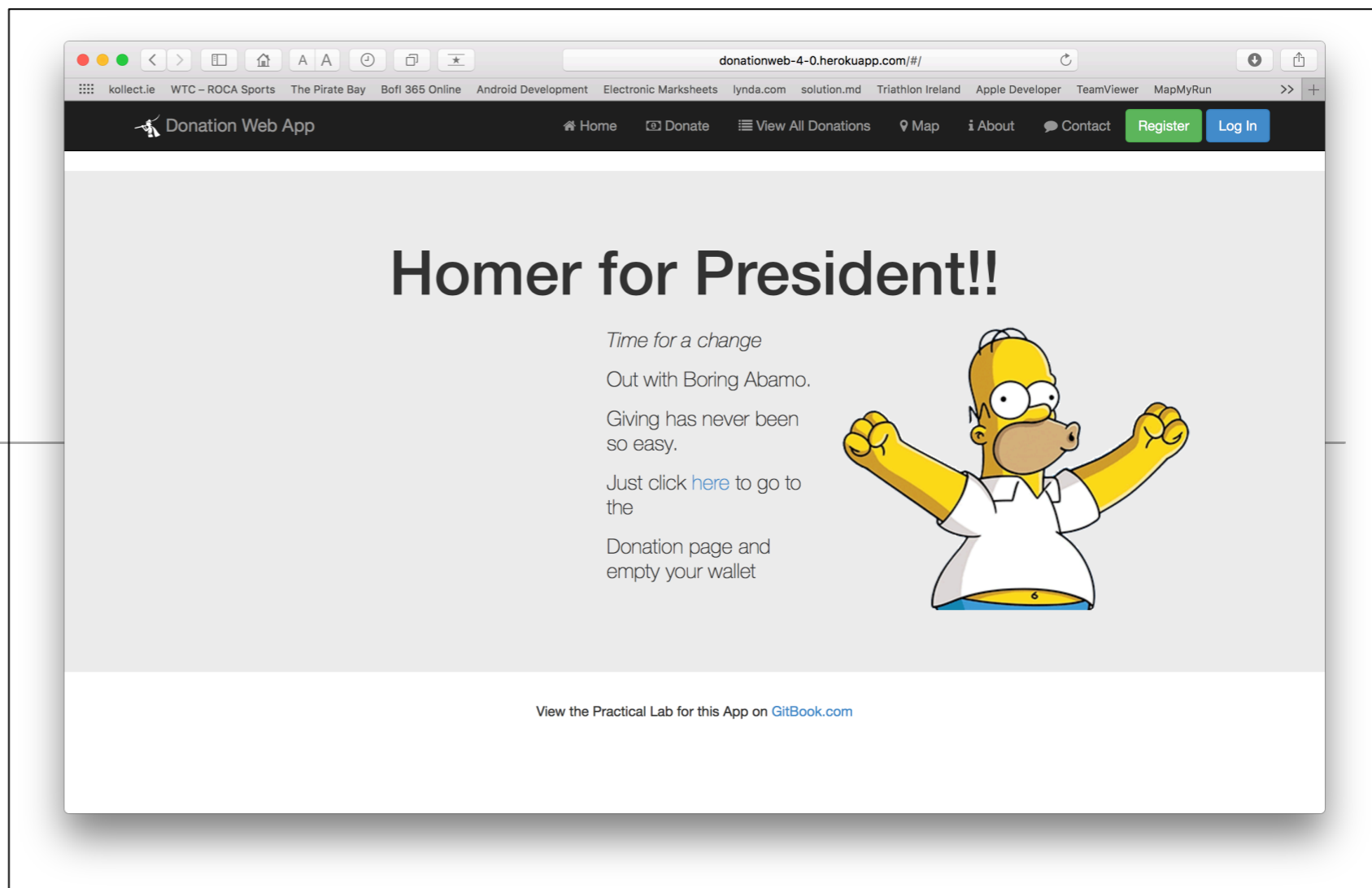# Assignment 2

70% of Overall Grade

Agenda

- Specification

- Grading Rubric

- Submission Guidelines

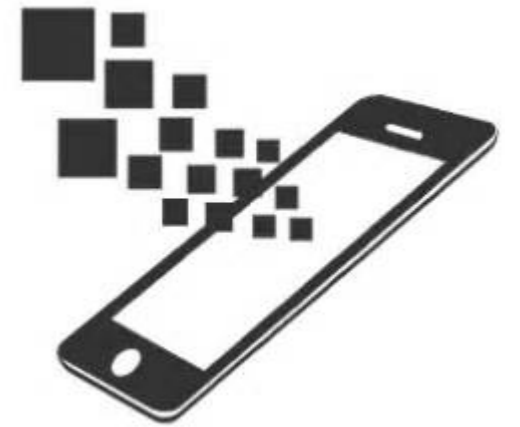- Presentation

Agenda

- **Specification**

- Grading Rubric

- Submission Guidelines

- Presentation

Assignment 2 – Options

Continue working on your own app, exhibiting similar level of complexity/feature density as covered in the 2nd half of the Semester.

Case Study RECAP – Donation (Assignment 1)

- A Node Web Server to manage donations made to 'Homers Presidential Campaign '.

- App Features (all via RESTful API)

  - POST a payment type and donation amount in JSON format

  - GET a list of donation amounts and types

  - GET an individual donation using an ID

  - DELETE an individual donation using and ID

 - Upvote a donation via PUT request

- Persistence via MongoDB

Case Study – Donation (Assignment 2)

- A FULL JS Web App with a Node Back-end and Angular front-end to manage donations made to 'Homers Presidential Campaign '.

- App Features

  - Make a Donation

  - List All Donations (and show the most 'upvoted')

  - Upvote an individual donation using an ID

  - DELETE an individual donation using and ID
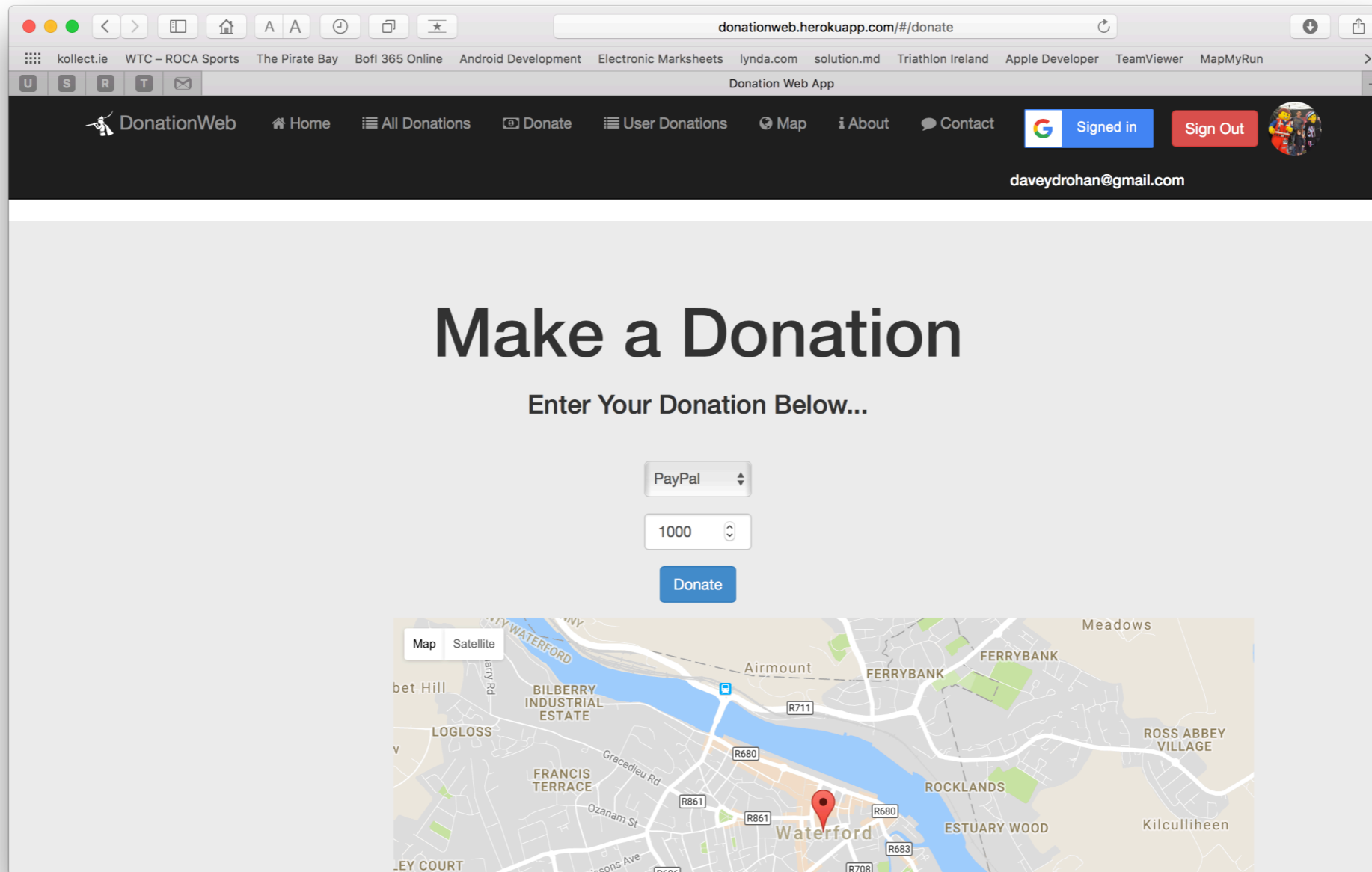
- Persistence via MongoDB

Sample Extra Features

1. Enable User Signup / Registration / Login.

2. The donations are persisted (in an SQLite database), and will be reloaded when a user logs in.

3. Support viewing/updating individual donations.

4. Allow a user to delete their own donations from the database.

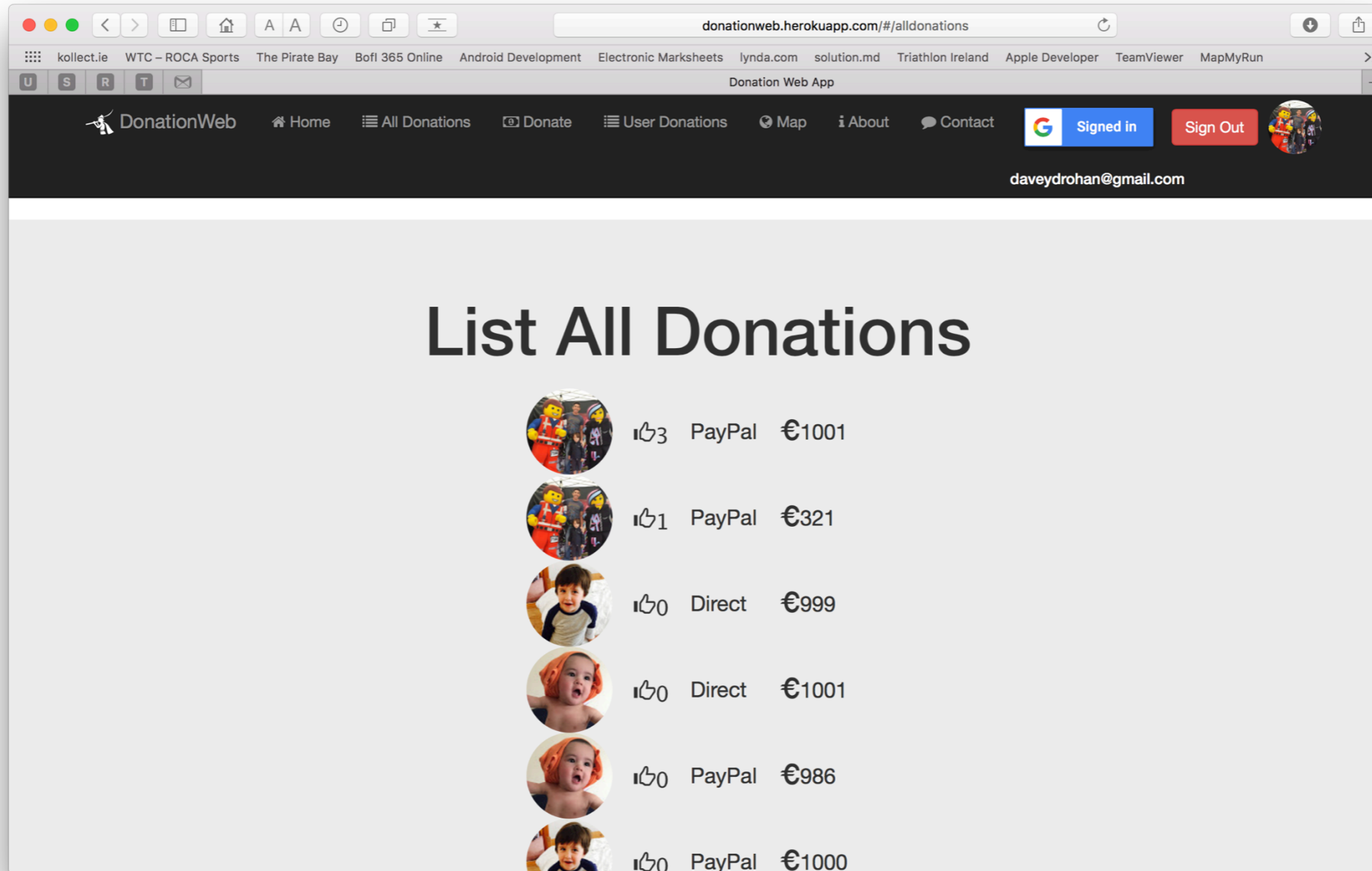5. Store a location with the donation and display on a Map, with donation info attached to marker.

Web App Features – Sign In
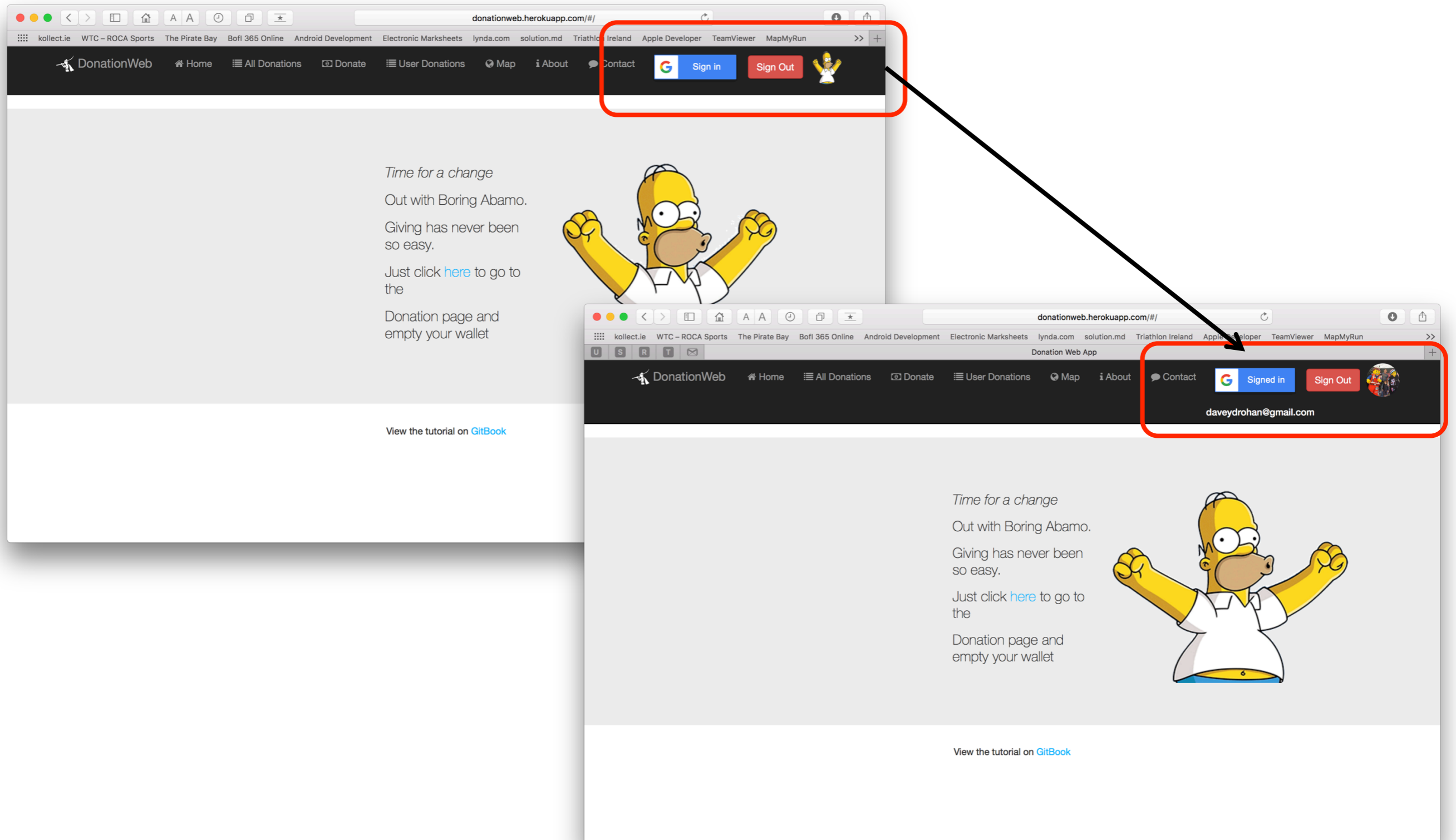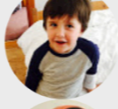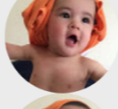
Web App Features – Make a Donation
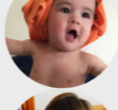
Web App Features – List All Donations

Web App Features – List User Donations
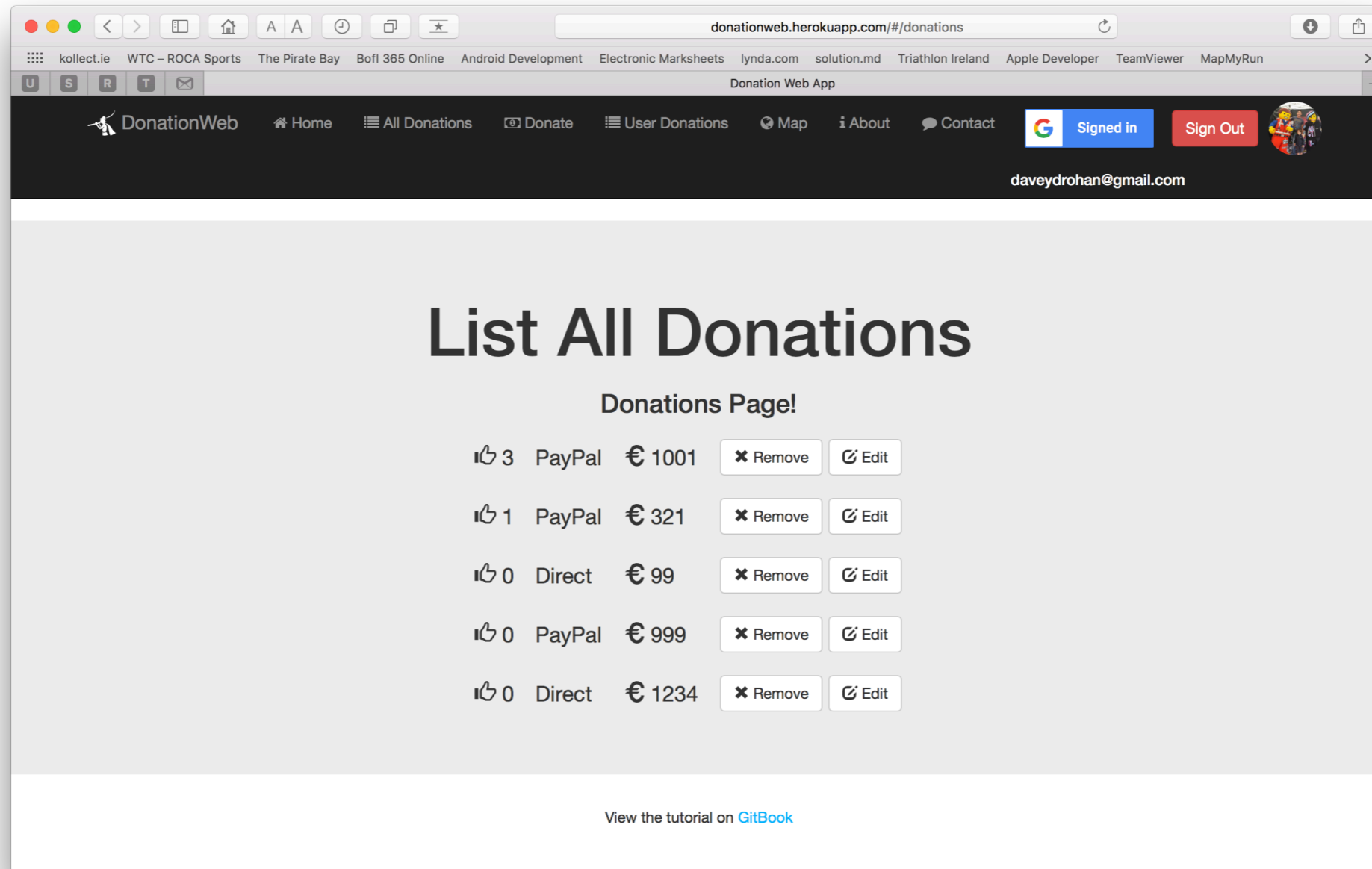
Web App Features – Edit a Donation
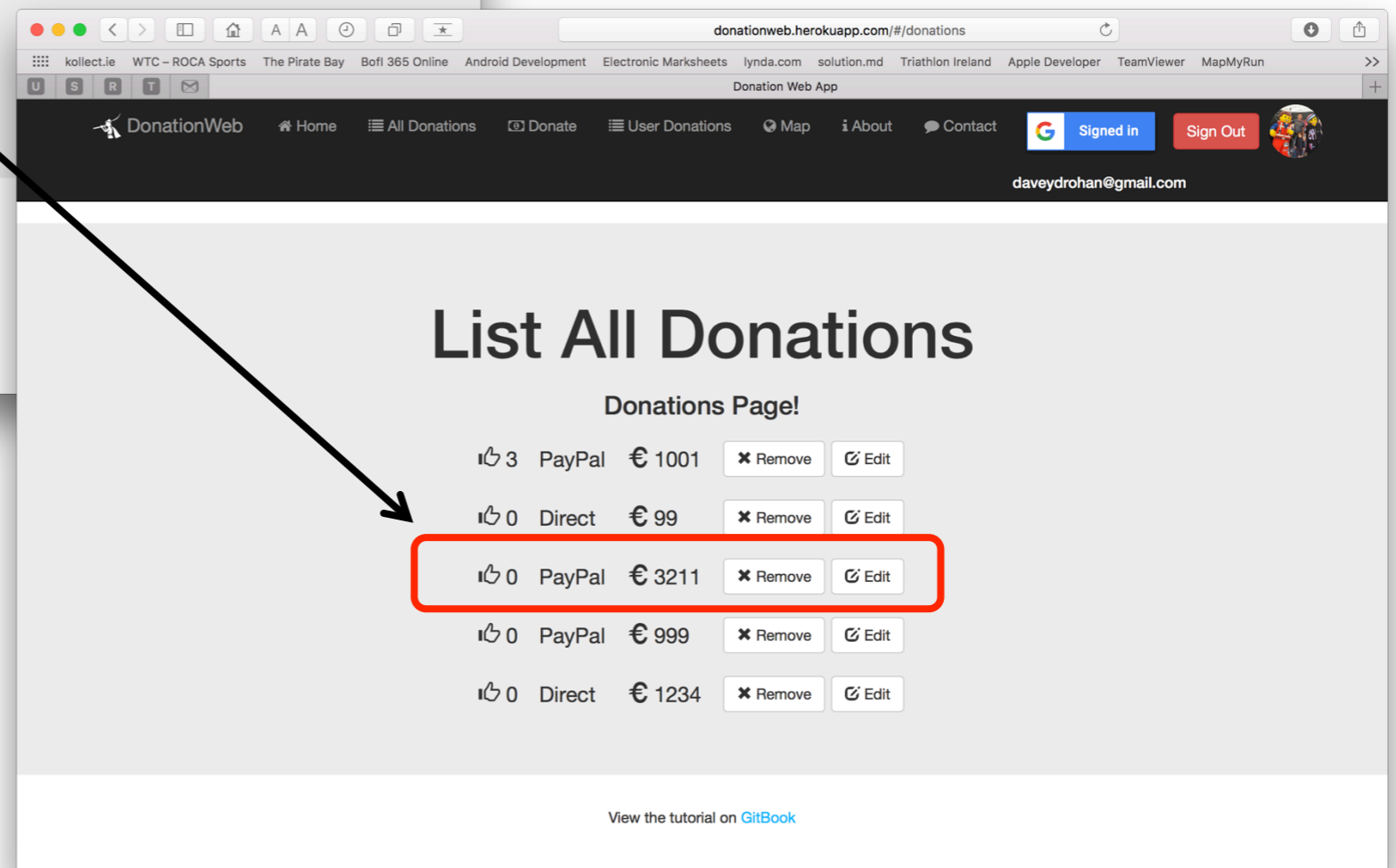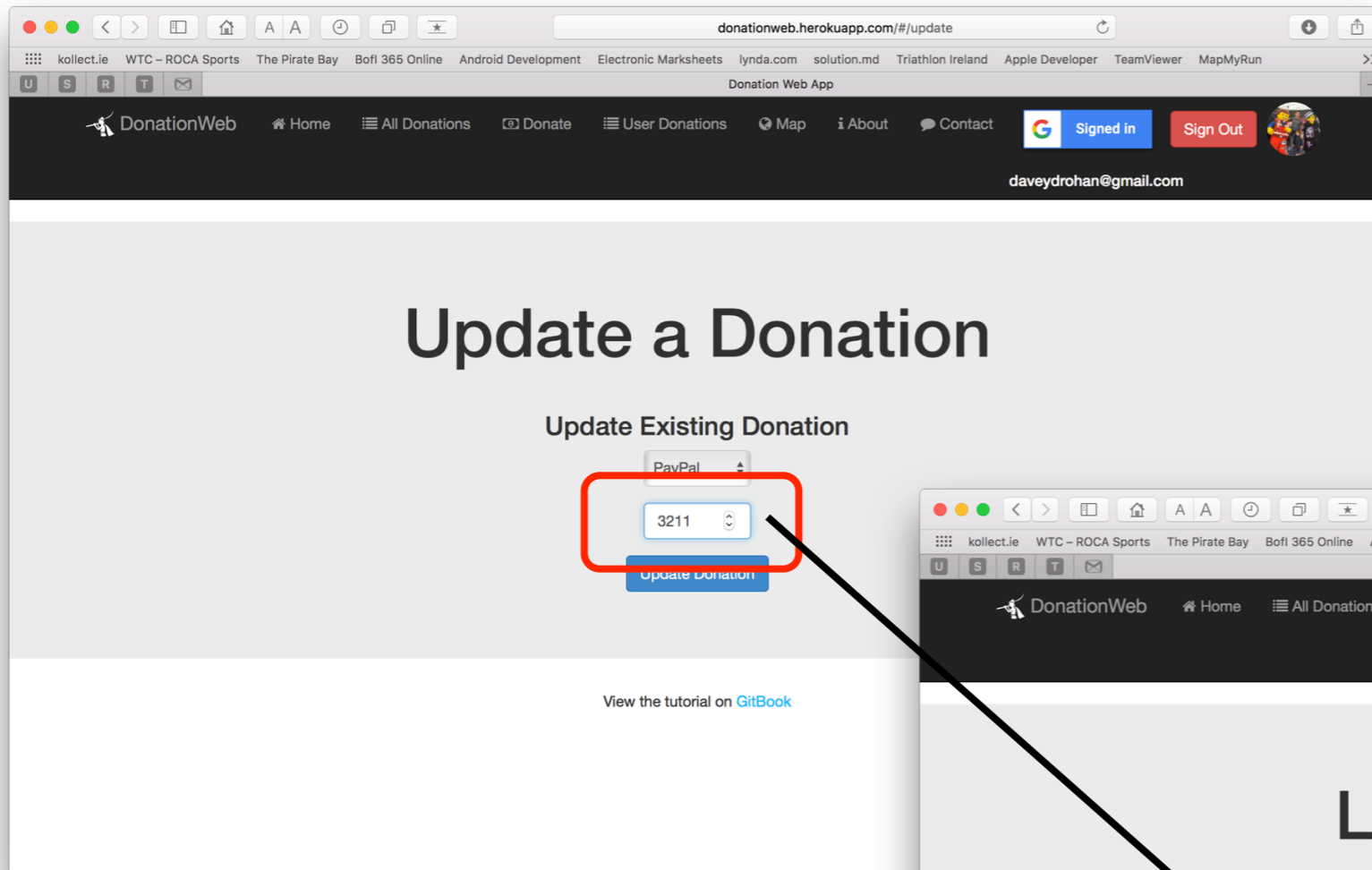
Web App Features – Delete a Donation

Web App Features – View Map
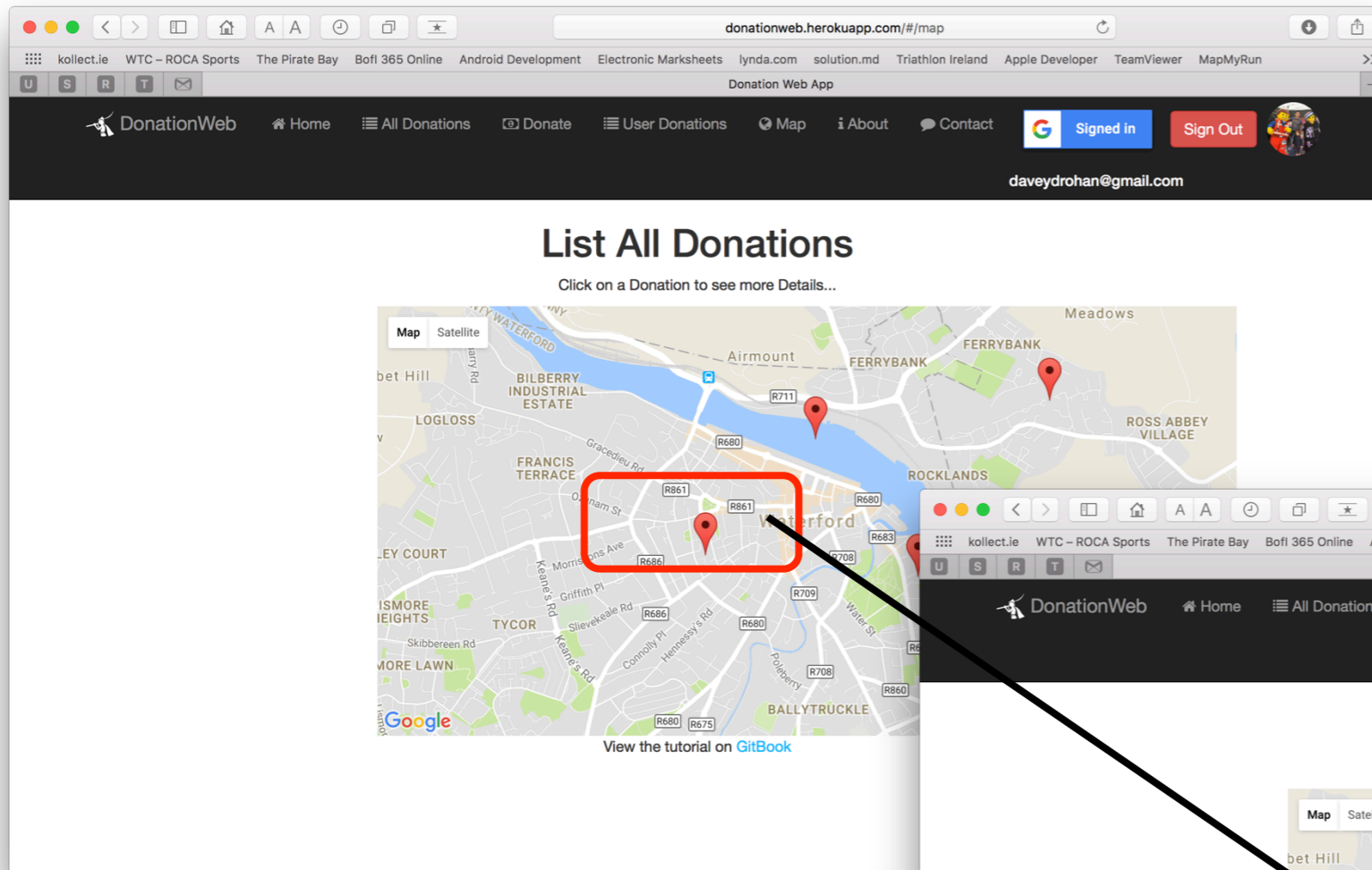
Agenda

- ~~Specification~~

- Grading Rubric

- Submission Guidelines

- Presentation

Assignment Rubric for Assignment 2

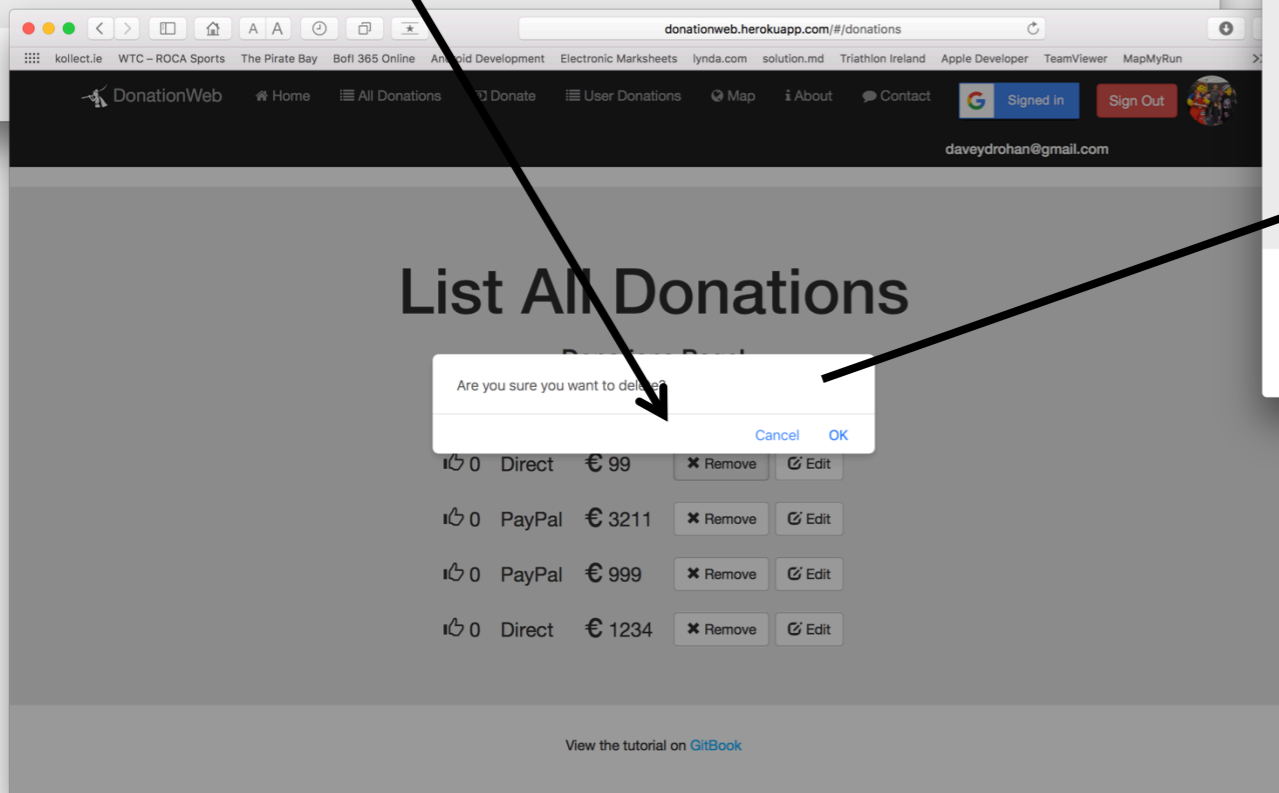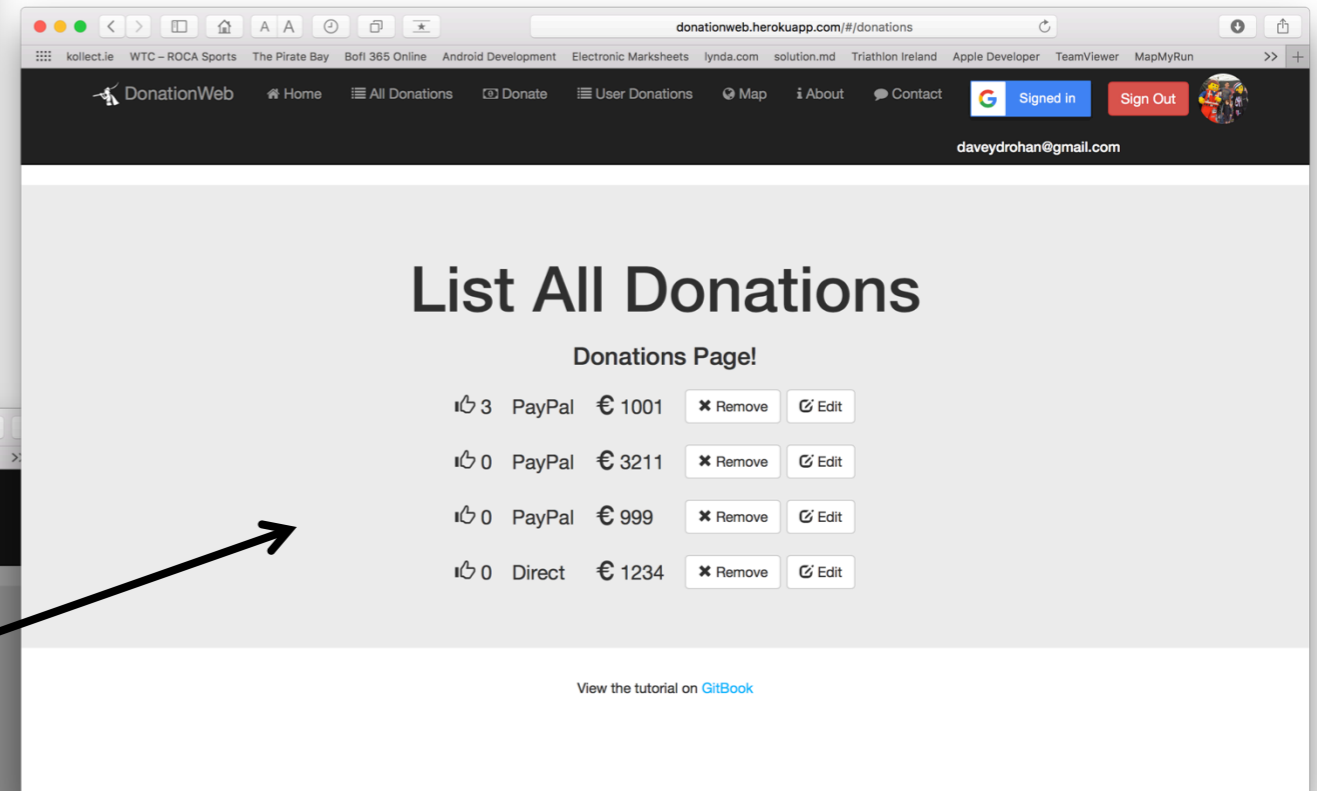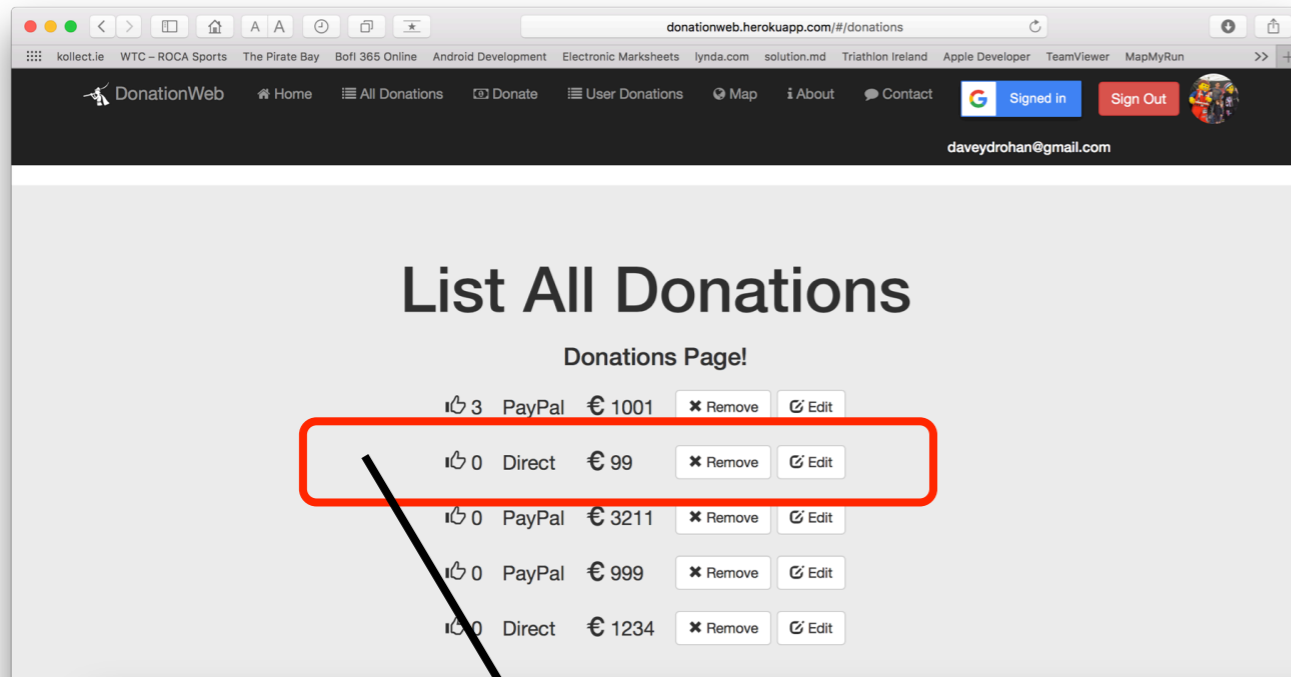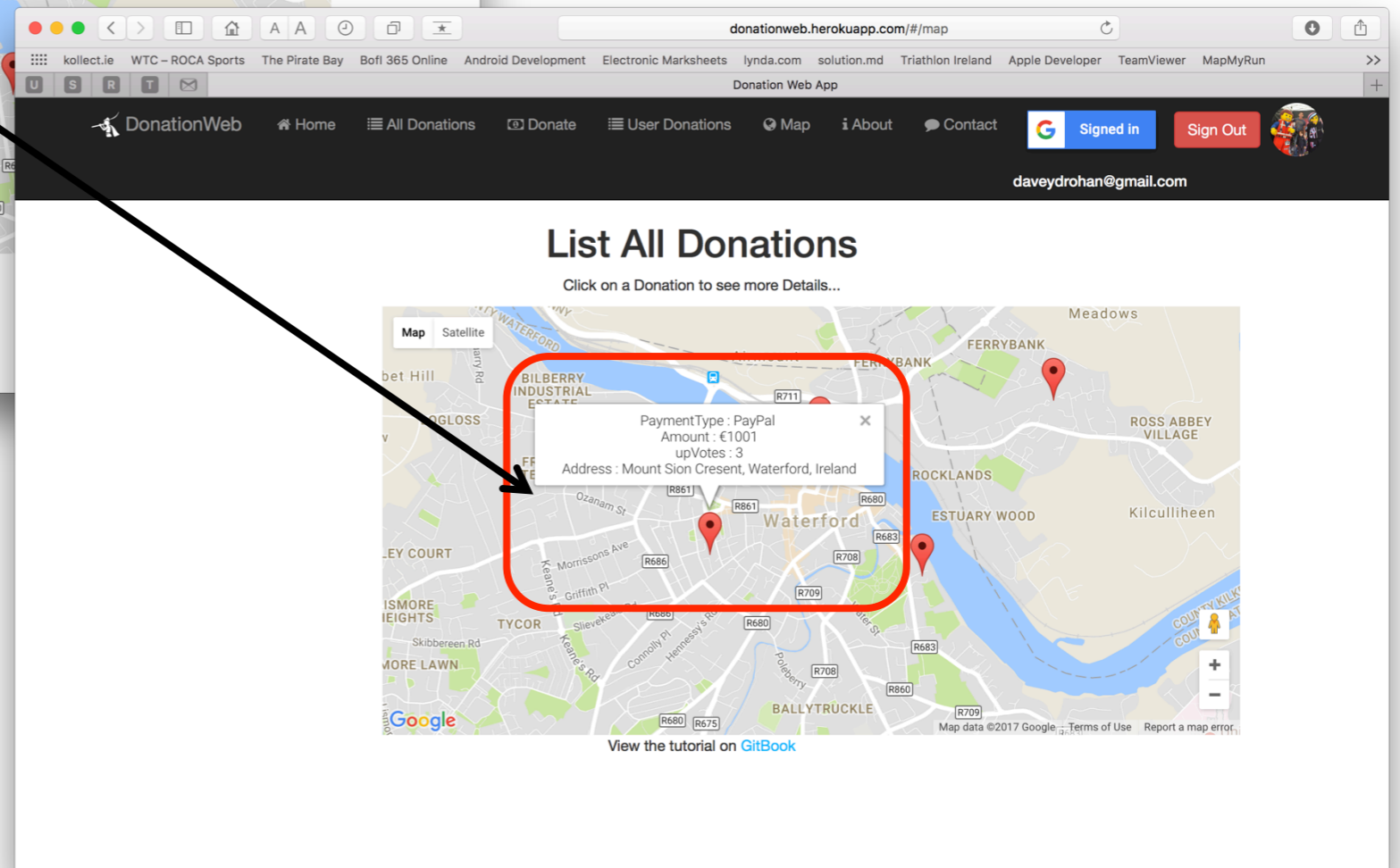| Standard	Client Functionality [60%]	Server Functionality [20%]	UX [10%]	DX [10%]
Baseline	Assignment 1 Functionality with Basic CRUD	MongoDB + Schema	App Navigation (via Menus)	Data Validation
Pass line Good	Additional Functionality as part of CRUD eg searching/filtering	> 2 Additional routes	Use of UI elements to complement UX eg DatePicker	Adherence to JS Best Practices eg SoC, Design
Very Good	Use of > 1 3rd Party API	> 3 Additional routes + Additional Models	UI Guidelines adhered to	Automated Testing
Excellent/ Outstanding (70%+)	Use of > 3 3rd Party APIs/ Google APIs	Cloud Support/acts as BaaS	Material Design Guidelines adhered to	Repo Usage, git etc.

Agenda

- ~~Specification~~

- ~~Grading Rubric~~

- Submission Guidelines

- Presentation

README file

Include a DESIGN Document file (max 20 pages):

- Name and Student ID.

- Full description of Web App functionality, including, Server & Client, specific Frameworks used and if any, 3rd party and/or Google APIs used.

- Appropriate UML Diagrams & Use Cases

- Database Schemas

- Git approach adopted and link to git project / access.

- UX/DX approach adopted.

- References

Submitting Project Code and APK

Submit zip of code via Moodle dropbox. This zip should also include:

- The Design Document file and

- full source of your web project

Give read access to your lecturer to your GitHub / BitBucket repos. GitHub and BitBucket ids are:

- **ddrohan.**

Agenda

- ~~Specification~~

- ~~Grading Rubric~~

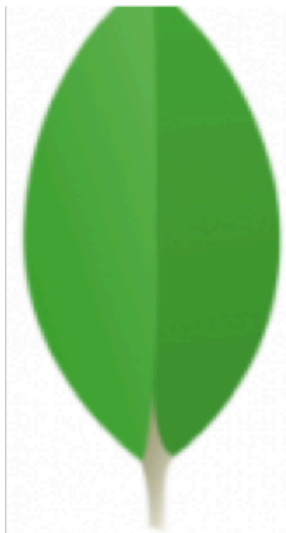- ~~Submission Guidelines~~

- Presentation

Presentation

You will be allocated a 15 minute slot in the week 12 practical labs to present your project.

- Attended by Tuition team only.

- 15 Minute to include demo + Q&A.

Note: I will be strict on the 15 minute allocation, so please arrive into the room with your Laptop ready to go with your app / code walkthrough.

Questions?