

Institiúid Teicneolaíochta Cheatharlach



At the Heart of South Leinster

Online Voting System

Project Report

Name: Mark Kelly

Student Number: C00198041

Course: Bachelor of Science (Honours) Software Development

Supervisor: Dr. Lei Shi

Date: 18th April 2018

Abstract

The purpose of this document is to provide a description of the submitted project. This document contains an introduction to the project, a detailed description of the project, conformance to specification and design, learning achievements, an overall review of the project and finally, acknowledgement to the people that helped along the way.

Table of Contents

1. Introduction	3
2. Project Description	3
2.1 Product description	3
2.2 Product screenshots	4
2.3 System description	9
2.4 Software description	10
2.4.1 Back end	10
2.4.2 Front end	10
2.4.3 Django libraries	10
2.4.4 Sending email notifications	11
2.4.5 Forcing HTTPS	11
3. Conformance to Specification and Design	11
3.1 Reminder of specification	11
3.2 Conformance	12
3.2.1 Changes in general	12
3.2.2 Changes in requirements	12
4. Learning Outcomes	13
4.1 Technical	13
4.2 Personal	13
5. Project Review	14
5.1 Things that went right	14
5.2 Things that went wrong	15
5.3 Things left to do	17
5.3.1 GDPR	17
5.4 What I would do differently if starting again	17
5.5 Advice for someone attempting a similar project	17
5.6 Implications of the technology choices	18
6. Acknowledgements	19
7.1 Bibliography	19

1. Introduction

This document will provide a detailed description of the suVote electronic voting application. The document will be composed of the following sections;

Project description

This section will provide a description of the finished project including software, the user interface for both the election administrator and the voter, and deployment of the project. Screenshots of the project will be provided.

Conformance to specification and design

This section will compare the submitted product with the initial specification and design. If it does not match, reasons and justifications as to why changes were necessary will be provided.

Learning outcomes

Learning outcomes will look at the technical and personal achievements. This section will look at new technologies learned, and what I learned about myself while working on this project over the past few months.

Project review

The project review will discuss the following;

- Things that went right
- Things that went wrong
- Things left to do
- Advice for someone attempting a similar project in the future
- Implications of the technology choices

Acknowledgements

This section is a hat tip to all the people that helped me over the past few months, and in some cases, years.

2. Project Description

This section provides a description of the finished project including software, the user interface for both the election administrator and the voter, and deployment of the project. Screenshots of the project will be provided.

2.1 Product description

suVote is an online voting platform for IT Carlow students to use to vote in IT Carlow Students Union elections and referenda. suVote provides the opportunity for students who are not on campus on the day of voting, for example students who are out on placement, to

register and vote online. There are two types of users able to interact with the web application, **election administrators** and **voters** (students).

Election administrators can login to the web application, create and manage elections, ballots and candidates. The election administrator can upload an Excel file with the email addresses of the allowed voters on it. The election administrator can also view past results of elections. Graphs and charts are provided to allow the election administrator to view and verify the results of past elections. Election administrators can print off a list of all the voters that have registered to vote online on the day of the election, so they can be removed from the paper ballot system, ensuring they can't vote twice.

Voters can register using their own college email address. Registration is a two step process, where the voter has to confirm their email address in order to complete the registration and login to the web application. On the day of an election, voters will receive a reminder email and be able to login and cast their vote. Voters have to register again for future elections.

2.2 Product screenshots

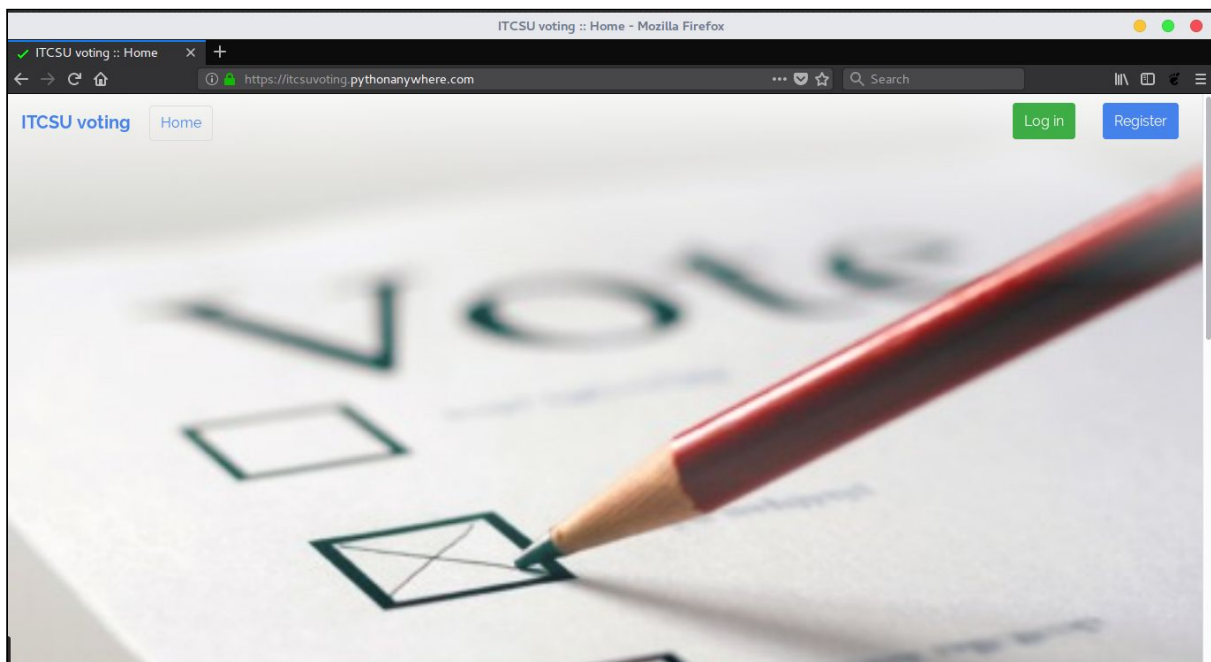


Fig. 2.1 Home page

Please Log In

Email address*
Enter Email

Password*
Enter Password

[Forgot Password?](#)

Remember me

Log in

Not registered to vote? [Register](#).

© Students' Union - Institute of Technology Carlow 2018 Connect with us on [f Facebook](#) or [t Twitter](#)

Fig. 2.2 Login page

Register to Vote

Email address*
Enter Email

Password*
Enter Password

Password confirmation*
Re-enter Password

Enter the same password as above, for verification.

Register

Already registered to vote? [Log in](#).

© Students' Union - Institute of Technology Carlow 2018 Connect with us on [f Facebook](#) or [t Twitter](#)

Fig. 2.3 Register page



Fig. 2.4 Voter logged in



Fig. 2.5 Election administrator logged in

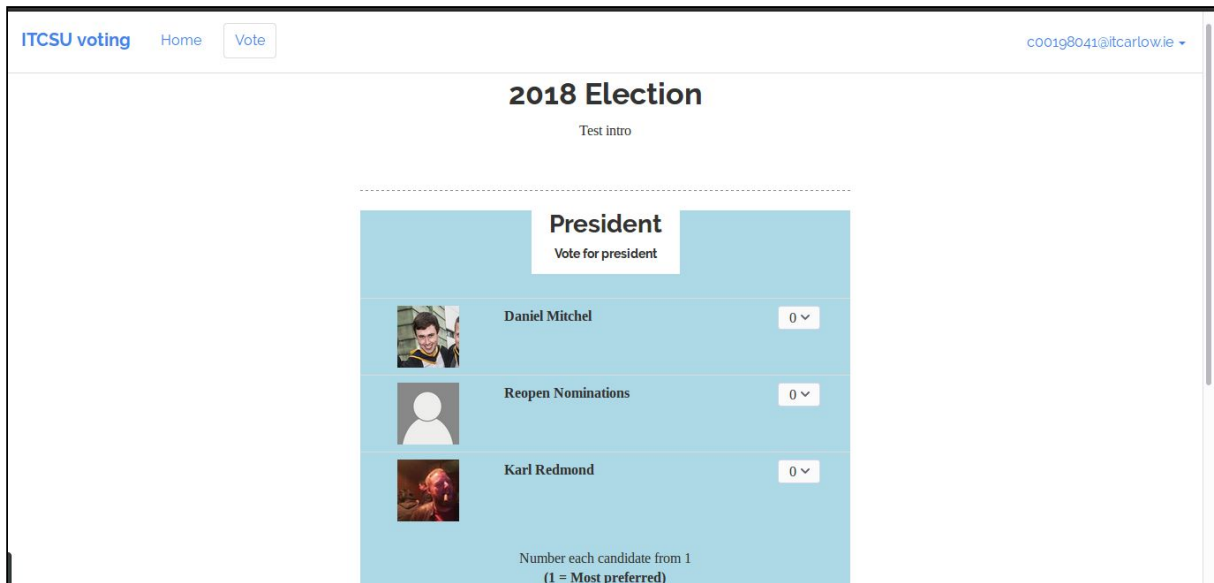


Fig. 2.6 Voting interface

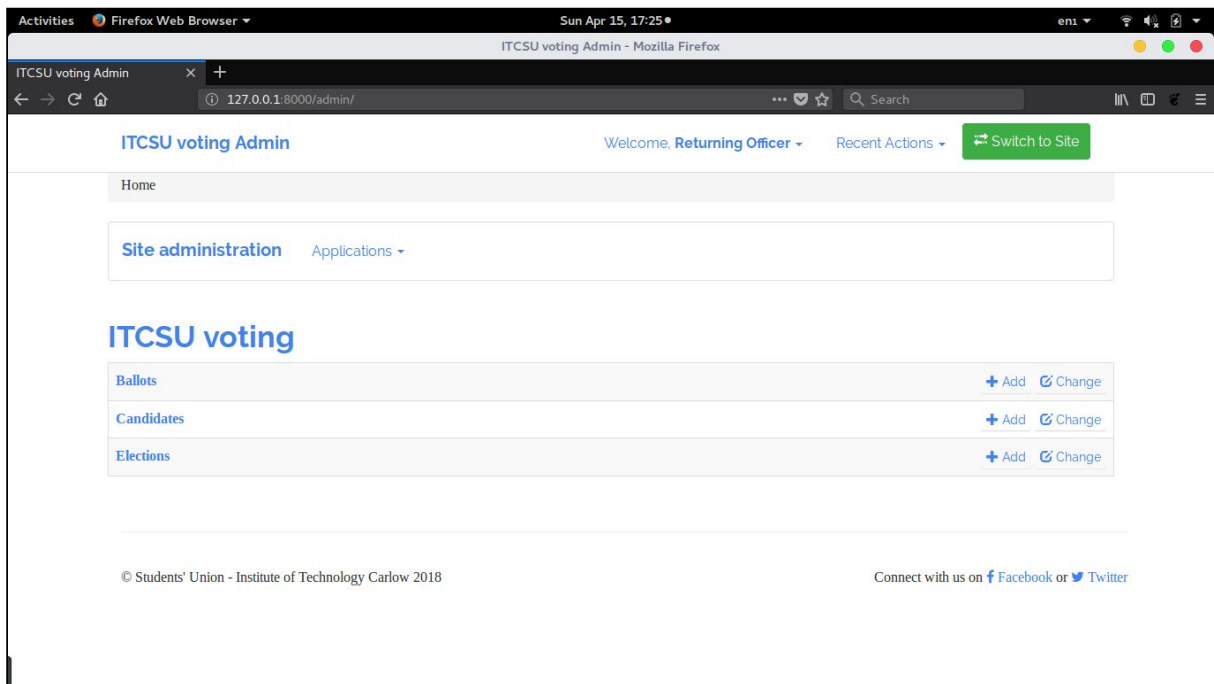


Fig. 2.7 Election administrator interface

The screenshot shows the 'Add election' page in the ITCSU voting Admin interface. At the top, there is a header with 'ITCSU voting Admin', a user greeting 'Welcome, Returning Officer', and a 'Switch to Site' button. Below the header is a breadcrumb trail: 'Home / Itcsu Voting / Elections / Add election'. A large blue button labeled 'Add election' is positioned at the top of the main content area. An orange banner below the button states 'Fields in bold are required.' The form includes a 'Name:' field with a text input and a tooltip that reads 'Used to uniquely identify elections. Will be shown with ' Election' appended to it on all publicly-visible pages.' Below this is an 'Introduction:' field with a larger text area. At the bottom of the page, there is a small note: 'This is entered at the top of the voting page below the header.'

Fig. 2.8 Add election page

The screenshot shows the 'Add ballot' page in the ITCSU voting Admin interface. The header and breadcrumb trail are similar to the previous page, but the breadcrumb trail is 'Home / Itcsu Voting / Ballots / Add ballot'. A large blue button labeled 'Add ballot' is at the top. An orange banner below the button states 'Fields in bold are required.' The form includes an 'Election:' field with a dropdown menu, a pencil icon, and a plus sign. Below this is a 'Position number:' field with a text input containing the number '1' and a tooltip that reads 'Change this if you want to customize the order in which ballots are shown for an election.' Below this are 'Description:' and 'Introduction:' fields with text inputs and a larger text area, respectively.

Fig. 2.9 Add ballot page

ITCSU voting Admin

Welcome, **Returning Officer** - Recent Actions - [Switch to Site](#)

Home / Itcsu Voting / Candidates / Add candidate

[Add candidate](#)

Fields in **bold** are required.

Ballot:

First name:

Last name:

Institution:

Profile picture: No file selected.

Fig. 2.10 Add candidate page

2.3 System description

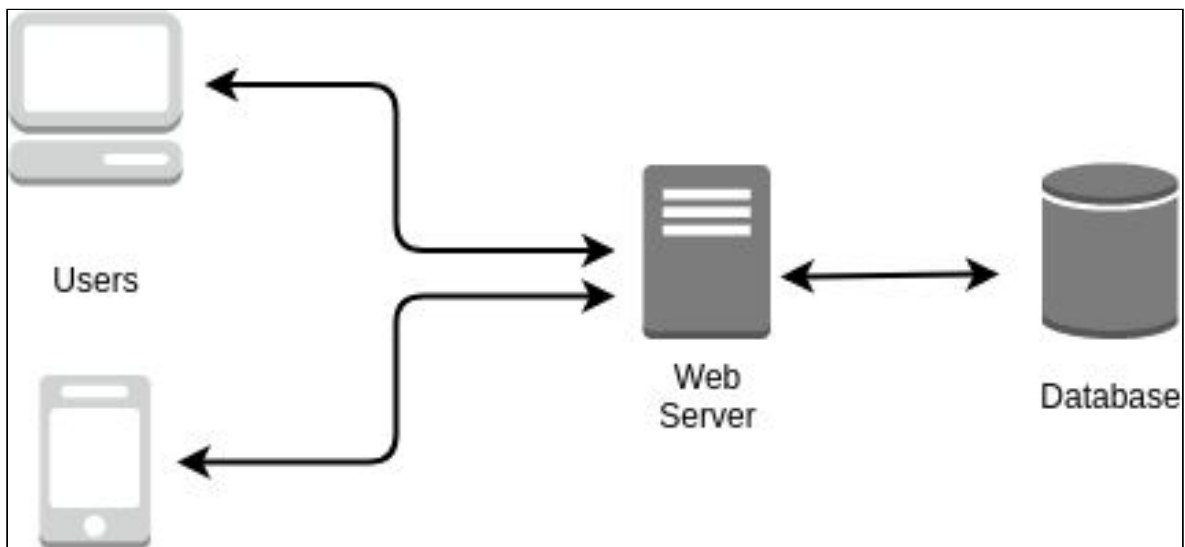


Fig. 2.1 System Architecture

The system architecture for this project was quite simple, as shown above in fig 2.1. The end **Users** have a fully responsive interface on any size device. Although, most election administrative tasks are best carried out on a desktop.

The **Web Server** is hosted on PythonAnywhere, a web hosting service based on the Python programming language.

The **Database** is an Amazon Relational Database Service (Amazon RDS) making it easy to set up, operate, and scale a relational database in the cloud. We can also pick the region where we want our database to live.

2.4 Software description

2.4.1 Back end

The web application was written in Python 2.7 and based on the Django 1.9 framework. The Django project structure is unlike previous frameworks I worked with, like Flask which has a much simpler structure. A Django web application is a project that contains a collection of apps. These apps can be slotted in and out of your project with relative ease. This allows for adding in apps to handle accounts, registration, media and other jobs that most web applications use.

2.4.2 Front end

The front end was developed using HTML, with Bootstrap to add a responsive design. jQuery was used for the timepicker on the election administrator interface for the start and end time of an election.

2.4.3 Django libraries

Below is the list of Django libraries and their versions, used to develop the web application.

```
confusable-homoglyphs==3.0.0
Django==1.9
django-admin-bootstrapped==2.5.7
django-authtools==1.6.0
django-autocomplete-light==3.3.0rc5
django-braces==1.12.0
django-crispy-forms==1.7.0
django-debug-toolbar==1.9.1
django-elect==0.1
django-environ==0.4.4
django-registration==2.4.1
easy-thumbnails==2.5
freezegun==0.3.9
mysqlclient==1.3.12
numpy==1.14.2
pandas==0.22.0
Pillow==5.0.0
python-dateutil==2.6.1
python-http-client==3.0.0
pytz==2018.3
sendgrid==3.6.5
sendgrid-django==4.2.0
six==1.11.0
sqlparse==0.2.4
Werkzeug==0.14.1
xlrd==1.1.0
```

2.4.4 Sending email notifications

The web application is deployed to PythonAnywhere on the free tier account. Free users are restricted to HTTP/HTTPS only, to a whitelist of sites. Because most email services work over SMTP, which is not HTTP or HTTPS, that means you cannot normally use SMTP on Free accounts. So in order to send emails from PythonAnywhere, I used a service called sendGrid, who allow you to send email using HTTP(S) requests, and their API endpoint are on the PythonAnywhere whitelist.

2.4.5 Forcing HTTPS

It is essential to encrypt our data in transit to make it harder for man in the middle attacks. With Django (1.8 or higher), this is a simple process. In settings.py, set:

```
SECURE_SSL_REDIRECT = True
```

Then make sure you have the Security Middleware enabled. This provides several security enhancements to the request/response cycle. Each one can be independently enabled or disabled with a setting.

3. Conformance to Specification and Design

This section will see if the submitted product matches the specification and design. Reasons and justifications as to why changes were necessary will be discussed.

3.1 Reminder of specification

The ITC Student's Union want to allow students to vote online in the student elections. The student's union are looking to develop an online voting platform for IT Carlow students to use to vote in IT Carlow Students Union elections and referenda.

The student's union wish to keep the traditional ballot paper on campus on the day of voting as it creates a great buzz on the campus during that day, but unfortunately, there is no opportunity for students who are not on campus on that day, for example students who are out on placement, etc. The student's union love the marketing opportunity for the union provided by the traditional elections, so for this reason a hybrid system is required.

The student's union would like the system to require students to register their intention to vote online in advance of the election (so they can remove them from the central register used in polling stations), and they then have to log in to the system during polling hours to cast their vote. When students have registered to vote online and log in during polling hours (they get an automated reminder email on the morning of the election), and can then log onto the system to cast their vote.

The student's union would also like to link the system to the student's blackboard account. The system must provide the following functionality:

- **Register for Voting**
 - Allow students to register online using their student ID
- **Cast Votes**
 - Students can place their vote and have it recorded correctly
- **Count Votes**
 - Give the correct results of the election
- **Display Results Graphically**
 - Display charts showing data visually
- **Full Security**
 - Fully test the system against cyber attacks
 - Show that an individual user's votes cannot be tracked while also showing traceability of the result.
- **Management UI**
 - An online based secure management console for setting up elections

3.2 Conformance

Overall, I believe the submitted project has achieved what was proposed in the functional specification and the design document. The core functionality of the application has been delivered and the product is functioning as designed.

3.2.1 Changes in general

After speaking with the product owner, the president of the students' union, Lorna Fitzpatrick, it was decided that we did not need to link the system to the student's blackboard account. This was decided because the system I proposed to develop would be a stand alone system that was not integrated into Blackboard. This had the benefit of making the system more modular so that it could be used for other colleges and institutions as opposed to being a niche product integrated into Blackboard.

3.2.2 Changes in requirements

One original requirement was for the ballots with only one candidate was to have an option to reopen nominations. This would allow voters to vote to reopen the nominations for that particular ballot. After one of my meetings with Lorna, she told me she wanted to have the option to reopen nominations for every ballot. This was not really an issue, as its as easy implement this for one candidate as it is for many candidates.

The next change of requirements came in a meeting with the election committee at the end of the second iteration. The original requirement was to display the results of the election to the screen, so that the election administrator can add on the results to their paper system and work out the winner. The proposed change was for the election administrator to input the results of the paper ballot into the web application and let it work out the winner. This

change came fairly late in the project and would require a lot of time and development to implement at this stage with limited time left on the last iteration.

4. Learning Outcomes

This section documents what my technical and personal achievements were and what I learned.

4.1 Technical

My biggest technical achievement was learning a new web framework called Django. I have lots of experience with the Flask web framework and I could of implemented this project a lot faster without having to learn new things, but where is the fun in that? I wanted to push myself out of my comfort zone and learn a new technology. I learned so much about Django, including the structure of a Django project, which is different to other project structures I worked with. I learned about working with URLs which involves using regex patterns in the views.

I also learned that deploying a Django project to PythonAnywhere is a lot more involved than deploying a Flask app. To serve static files in Django you need to run `$ python manage.py collectstatic`. This allows `django.contrib.staticfiles` to collect static files from each of your applications (and any other places you specify) into a single location that can easily be served in production.

I know I'm still a student with a lot to learn, but with every project I complete, I can feel my technical powers growing. I'm coming out of this project with the power to design, develop, and deploy production ready web applications to a global audience. I'm going to take the valuable technical knowledge I have aquired from this project and develop my own blog in Django over the summer (As soon as I recover from fourth year!).

4.2 Personal

When I first heard that I got the E-Voting project, I was so excited because it was for a real client and my product has the potential of being used by the student's union for upcoming elections. My excitement soon turned to nerves as I thought about the task ahead. I've always had an issue where I doubt myself and my skills. I was thinking things like "how am I going to be able to deliver this product, and on time?", "will I make a fool of myself when talking in the client meetings?", "will I be able to design a good UI?". As time went on, I started meeting the client and developing small bits of functionality in little byte sized projects. I was getting good feedback from the client and my confidence started to grow. I was starting to believe in myself and my abilities.

I'd day my biggest personal achievement was completing the project, and the course. I had to take a few weeks off just after Christmas to look after a family member. When I returned to college I was overwhelmed with the amount of work I had to catch up on. There was

deadlines and continuous assessment coming from everywhere. That's when I started to doubt myself. I started to think "I can't do this". I was looking at all this work not knowing where to start. But I decided, I can't give up! What kind of example would I set for my kids? I decided I would rather fight on and possibly fail, rather than taking the easy way out by giving up. I learned not to get overwhelmed by looking at this huge pile of work. I broke everything in to small task, and just chipped away at them.

I believe this project has helped me to believe in myself more on a personal level. I feel more confident when talking to clients and groups of people. My presentation skills also improved as I'm not even half as nervous as I used to be. I am so grateful for the experience I had working on a real world project. This project (and year) showed me I'm stronger than I thought, both technically and personally.

5. Project Review

When I started this project I had a clear plan in my head of developing a web application. After meeting my tutor, Dr. Lei Shi for the first time, he suggested using Blockchain technology to create a decentralized app (dapp). This overwhelmed me a little as I already had a lot of research gone into creating a traditional web app. I was worried about failing to deliver a dapp, when I was confident with my ability to deliver an app on time.

This section will review things that went right and wrong, things left to do, advice for someone attempting a similar project in the future, and the implications of the technology choices that I made.

5.1 Things that went right

I think I had a slight advantage doing this project, as I also done the third year project last year. The fourth year project is bigger, but the structure is much the same. This advantage gave me some breathing space and allowed me to concentrate on development. The only part of this project I was worried about was the deployment. I had to abandon my third year project, which was a Flask web app, because I couldn't get it deployed. I did have a problem deploying my fourth year project, but it was easy to resolve after a quick email to PythonAnywhere support. They even added me to their testimonials page, fig 5.1.

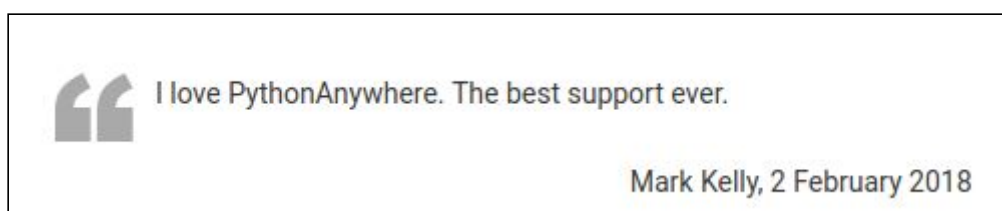
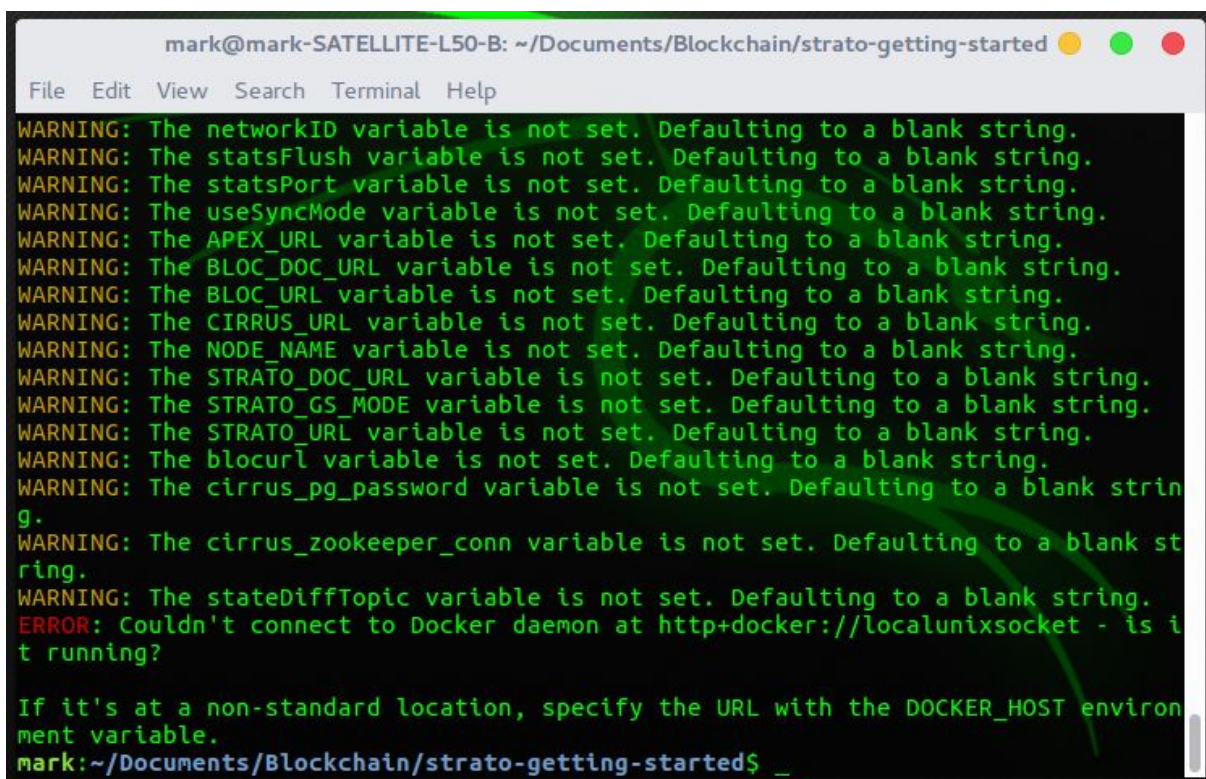


Fig. 5.1 PythonAnywhere testimonial page

5.2 Things that went wrong

After I had my first meeting with Dr. Lei Shi and he suggested using Blockchain technology to create a decentralized app (dapp), I decided to find some tutorials on Blockchain technology. After some research I went with BlockApps STRATO Platform for Blockchain Application Development. STRATO, BlockApps is a full-stack technology solution that allows you to build industry-specific Blockchain applications on top of your own customized permissioned private, consortium (semi-private) ledgers, public Blockchain ledgers, or on the permission-less public Ethereum platform.

I started to follow the tutorials, but I struggled to get the platform up and running. I recently changed my operating system to Linux and I had to set up nginx web server on my machine. This was a learning curve as I was trying to find my feet with a new operating system. I couldn't connect to the Docker daemon. Fig 5.2 below, shows the error I was getting. I eventually found a fix on the GitHub forums. I had to run `docker ps` in the terminal, which got me to the next step.

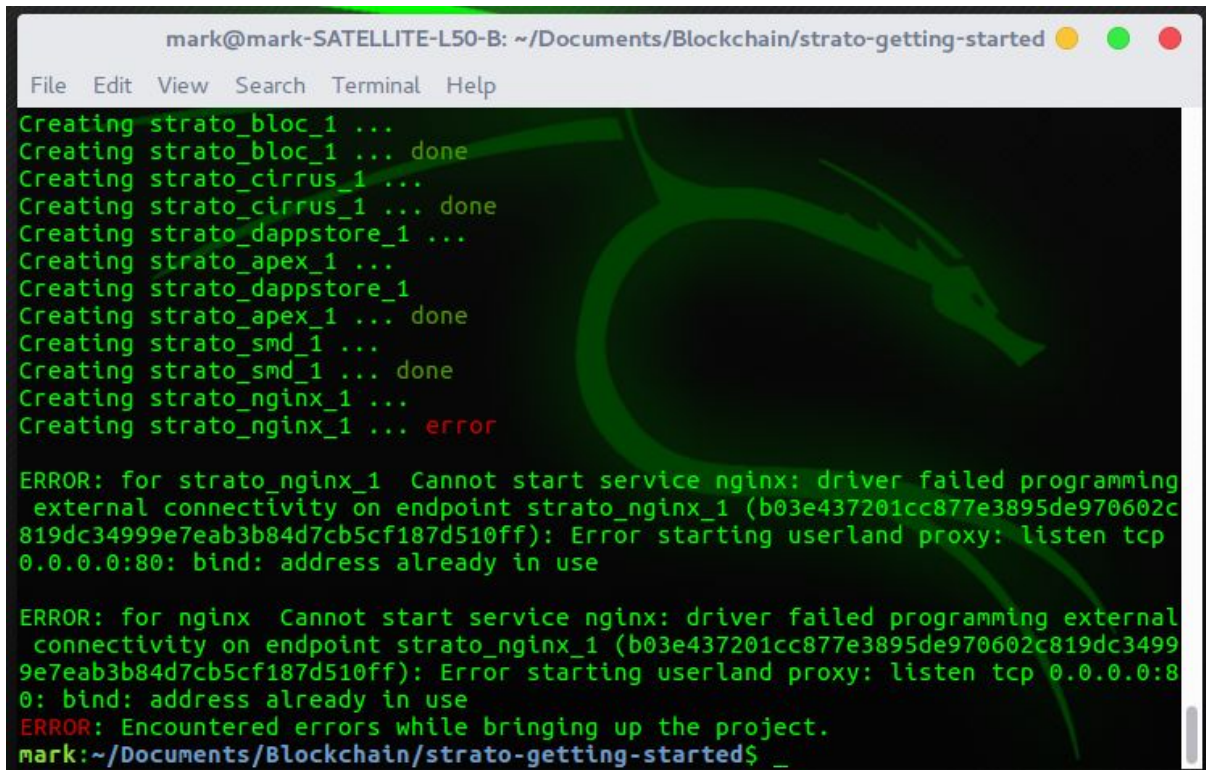
A terminal window screenshot showing a series of warning and error messages. The window title is 'mark@mark-SATELLITE-L50-B: ~/Documents/Blockchain/strato-getting-started'. The terminal output consists of 15 lines of warnings, each stating that a specific environment variable is not set and defaulting to a blank string. The variables listed are: networkID, statsFlush, statsPort, useSyncMode, APEX_URL, BLOC_DOC_URL, BLOC_URL, CIRRUS_URL, NODE_NAME, STRATO_DOC_URL, STRATO_GS_MODE, STRATO_URL, blocurl, cirrus_pg_password, and cirrus_zookeeper_conn. The 16th line is an error message: 'ERROR: Couldn't connect to Docker daemon at http+docker://localunixsocket - is it running?'. Below the error message, there is a line of text: 'If it's at a non-standard location, specify the URL with the DOCKER_HOST environment variable.' The terminal prompt is 'mark:~/Documents/Blockchain/strato-getting-started\$ _'.

```
mark@mark-SATELLITE-L50-B: ~/Documents/Blockchain/strato-getting-started
File Edit View Search Terminal Help
WARNING: The networkID variable is not set. Defaulting to a blank string.
WARNING: The statsFlush variable is not set. Defaulting to a blank string.
WARNING: The statsPort variable is not set. Defaulting to a blank string.
WARNING: The useSyncMode variable is not set. Defaulting to a blank string.
WARNING: The APEX_URL variable is not set. Defaulting to a blank string.
WARNING: The BLOC_DOC_URL variable is not set. Defaulting to a blank string.
WARNING: The BLOC_URL variable is not set. Defaulting to a blank string.
WARNING: The CIRRUS_URL variable is not set. Defaulting to a blank string.
WARNING: The NODE_NAME variable is not set. Defaulting to a blank string.
WARNING: The STRATO_DOC_URL variable is not set. Defaulting to a blank string.
WARNING: The STRATO_GS_MODE variable is not set. Defaulting to a blank string.
WARNING: The STRATO_URL variable is not set. Defaulting to a blank string.
WARNING: The blocurl variable is not set. Defaulting to a blank string.
WARNING: The cirrus_pg_password variable is not set. Defaulting to a blank string.
WARNING: The cirrus_zookeeper_conn variable is not set. Defaulting to a blank string.
WARNING: The stateDiffTopic variable is not set. Defaulting to a blank string.
ERROR: Couldn't connect to Docker daemon at http+docker://localunixsocket - is it running?

If it's at a non-standard location, specify the URL with the DOCKER_HOST environment variable.
mark:~/Documents/Blockchain/strato-getting-started$ _
```

Fig. 5.2 Error message

Then I was having problems trying to start the nginx service with STRATO. This was the last step to getting the Strato Management Dashboard up and running. Fig 5.3, on the next page shows the error message I was getting.

A terminal window screenshot showing the process of creating services. The terminal output lists several services being created successfully, such as strato_bloc_1, strato_cirrus_1, strato_dappstore_1, strato_apex_1, and strato_smd_1. The creation of strato_nginx_1 fails with an error. The error message states: "ERROR: for strato_nginx_1 Cannot start service nginx: driver failed programming external connectivity on endpoint strato_nginx_1 (b03e437201cc877e3895de970602c819dc34999e7eab3b84d7cb5cf187d510ff): Error starting userland proxy: listen tcp 0.0.0.0:80: bind: address already in use". A second, identical error message is shown for the 'nginx' service. The terminal ends with the prompt "mark:~/Documents/Blockchain/strato-getting-started\$ _".

```
mark@mark-SATELLITE-L50-B: ~/Documents/Blockchain/strato-getting-started
File Edit View Search Terminal Help
Creating strato_bloc_1 ...
Creating strato_bloc_1 ... done
Creating strato_cirrus_1 ...
Creating strato_cirrus_1 ... done
Creating strato_dappstore_1 ...
Creating strato_apex_1 ...
Creating strato_dappstore_1 ...
Creating strato_apex_1 ... done
Creating strato_smd_1 ...
Creating strato_smd_1 ... done
Creating strato_nginx_1 ...
Creating strato_nginx_1 ... error

ERROR: for strato_nginx_1 Cannot start service nginx: driver failed programming
external connectivity on endpoint strato_nginx_1 (b03e437201cc877e3895de970602c
819dc34999e7eab3b84d7cb5cf187d510ff): Error starting userland proxy: listen tcp
0.0.0.0:80: bind: address already in use

ERROR: for nginx Cannot start service nginx: driver failed programming external
connectivity on endpoint strato_nginx_1 (b03e437201cc877e3895de970602c819dc3499
9e7eab3b84d7cb5cf187d510ff): Error starting userland proxy: listen tcp 0.0.0.0:8
0: bind: address already in use
ERROR: Encountered errors while bringing up the project.
mark:~/Documents/Blockchain/strato-getting-started$ _
```

Fig. 5.3 Error message

I looked through countless forums including Stack Overflow and I couldn't find a fix. I had nearly a week invested into trying to get the Strato Management Dashboard up and running. At this stage I was starting to panic about having no product at all. So I decided to build a traditional web app, and try fall back later to implement Blockchain technology into the voting part of the web app.

The first iteration went really well, and I was proud of what I achieved. The plan was to finish the web app by the end of the second iteration, and spend the final iteration having another go at Blockchain technology. But the second iteration was tough going with all the other deadlines and continuous assessments, and I didn't have as much time on my project as I hoped to.

This problem was pointed out by Hisain Elshaafi, at the second iteration presentation. He pointed out that the EU provides strong protection for personal data. If data belonging to EU businesses or citizens is stored outside the EU, the transfer of that data needs to be secure with data protection requirements at the other end at least as strong as those in the EU. The initial system architecture design presents a problem when using the database on pythonanywhere. This is because the pythonanywhere servers are outside the EU. For this reason We shall use Amazon Web Services (AWS) to host the database. AWS allow you to pick the region where your data is stored.

5.3 Things left to do

I believe I managed to hit all the core functionalities on this project. But there was a couple of things left to do. The first thing was testing. Before the project could go into production, the code would have to be fully tested. I had plans of writing tests for all the views, forms, and models. The second thing left to do was penetration testing. It is very important to evaluate the security of the web app by safely trying to exploit any vulnerabilities. I would of liked to of tested a mock election with my fellow students, but due to time constraints, this was not possible.

5.3.1 GDPR

The EU General Data Protection Regulation (GDPR) replaces the Data Protection Directive 95/46/EC and was designed to harmonize data privacy laws across Europe, to protect and empower all EU citizens data privacy and to reshape the way organizations across the region approach data privacy [1].

As the project went on, I learned more about the GDPR, and how it affects me as a developer. I have the my data in transit encrypted over HTTPS, but I still need to encrypt my data at rest. MySQL Enterprise Encryption provides industry standard functionality for asymmetric encryption, but with an annual subscription cost of \$5000 for the MySQL Enterprise Edition [2], this is not an option. Given more time, the plan would be to use PyCrypto, a Python cryptography toolkit intended to provide a reliable and stable base for writing Python programs that require cryptographic functions [3].

5.4 What I would do differently if starting again

If I was to start this project again right now, I would of challenged myself more with the Blockchain technology and decentralized applications (dapps). I'm sorry I didn't spend more time trying to get the dapp working. I started to panic about how much time I was investing in it with little or no progress. I have since found tutorials for creating a dapp on the Ethereum Blockchain app platform using NodeJs and Solidity. Solidity is a contract-oriented programming language for writing smart contracts, its syntax is very like JavaScript. If I was starting again knowing what I know now, I would have stuck it out with the Blockchain technology and tried to develop a dapp on the Ethereum blockchain platform.

5.5 Advice for someone attempting a similar project

You should spend plenty of time researching your technology choices. Make sure the technology is right for the job before you make your mind up Don't be afraid to set up a sandbox to test out some technologies that you think you will use for your project. I would definitely advise some form of version control right from the start. If you drop your laptop, you will be happy in the knowledge that your code is save on GitHub.

If your building a web application in Django, or any other framework, use a boilerplate. A boilerplate is like a scaffold or skeleton template that provides most features that you would expect and help you focus on the actual problem. At the same time, it avoids the bloat of numerous dependencies or complex apps. I can't recommend this enough. I have watched a lot of my peers getting bogged down in mundane development tasks that nearly every web application needs. Use a boilerplate to get you started quickly and concentrate on the main functionality of your web application. Check out AWESOME DJANGO [4], which is the holy grail of Django resources to help you bootstrap your project.

As soon as you have a basic app working, deploy it straight away. When it comes to deploying a web application, there are so many things that can go wrong. I have had to abandon past projects because I couldn't get it deployed. Most of the time, It's not a quick process. Even with my project this year, I had problems deploying my Django app to PythonAnywhere. It took me close to a week to get the issue sorted. If you leave it until the eleventh hour and think you can just throw it up onto PythonAnywhere, you could find yourself in trouble with no time to fix it.

Try not look at the project as a whole. This can overwhelm you when you think of all the stuff you need to accomplish. Break the project down into small chunks of functionality, and work on each one. When you achieve each chunk, push to GitHub and move onto the next chunk. The trick is not to focus on fluffy stuff like logging in and out, leave all the fluffy stuff until the core functionality is achieved. There is no point having a shiny login system that logs into nothing. Pick the hardest piece of functionality and focus on that first. Always ask yourself, is this core functionality?

Some general advice for any project, would be to start your final report when you start the first iteration of coding. I'm not saying write the whole thing. Everytime you hit a problem, document and screenshot it straight away. If it's not in the final report, it didn't happen. It's also a good idea to document and screenshot everything when your deploying or installing your project, this will be very useful at the end of the project, when it comes to writing your administrative manual on how to install and use your system.

My last bit of advice would be, don't sit in a corner on your own. Get yourself into a small group of like minded people that are willing to put in the effort, and support each other throughout the year. Help each other out. In my opinion, this is very important, as you more than likely won't be sitting on your own in the corner when you start working for a company.

5.6 Implications of the technology choices

The most of my research time on technologies went on frameworks. I looked at Flask, Pyramid, Web2py, Bottle, and Django. I have experience with Flask and I knew it could do the job for my project. I should of used Flask instead of Django. I was biased in my decision for a web framework as I really wanted to get some experience with Django and learn a new technology. Django is more of a heavy duty framework and best suited to larger web applications. Flask is a more lightweight framework, and would have been better suited to a small web application such as the one in my project. I was happy with the rest of my

technology choice, but it was always in the back of my mind that I should have used the Flask framework.

6. Acknowledgements

I would like to thank my fellow students, especially Gerrard Dobbs, Karl Redmond and Ronan Timmons who gave me invaluable feedback day or night about my project over the course of the year.

I would like to thank the 4th year lectures who helped me to improve my presentation skills and also gave constructive criticism at both my sprint presentations. Thank you to Hisain Elshaafi for his comments on data storage in the EU. I took onboard all the questions and comments and applied them to my project. I would like to thank Dr. Joe Kehoe for all the organising and information he has given us in relation to the project. It always felt like Joe was on our side and he had our backs. I would like to give a special thanks to Paul Barry, who has inspired me so much both this year and last. Even though he wasn't my project tutor, he gave me his time and knowledge without hesitation. I shall never forget Paul, or any of the stuff he taught me over the past couple of years.

I would like to specifically thank my supervisor, Dr. Lei Shi. From day one, he was pushing me out of my technological comfort zone. He gave me great support and opened my eyes to different technologies. He was always there to provide feedback on my progress and presentations. He was very supportive throughout the process and always finished our meetings making me feel confident.

Nobody has been more important to me in the course of this project than the members of my family. I would like to thank my dad for buying me my laptop so I could pursue my dreams. my brother, whose support and guidance are with me in whatever I pursue. They are the ultimate role models. Most importantly, I wish to thank my loving and supportive wife, Ruth, who always believed in me, and my four amazing children, Cillian, Jamie, Callum and Lyla, who provide endless inspiration.

7.1 Bibliography

[1] EU GDPR Portal. 2018. Home Page of EU GDPR. [ONLINE] Available at: <https://www.eugdpr.org/>. [Accessed 10 April 2018].

[2] MySQL :: MySQL Editions. 2018. MySQL :: MySQL Editions. [ONLINE] Available at: <https://www.mysql.com/products/>. [Accessed 10 April 2018].

[3] PyCrypto - The Python Cryptography Toolkit. 2018. PyCrypto - The Python Cryptography Toolkit. [ONLINE] Available at: <https://www.dlitz.net/software/pycrypto/>. [Accessed 10 April 2018].

[4] Awesome Django. 2018. Awesome Django. [ONLINE] Available at: <http://awesome-django.com/>. [Accessed 10 April 2018].