Scripting Demo Prototype Coursework Report

Prototype Functionality Description

With the dramatic increase of different devices tracking fitness and other daily activities, comes an associated need to analyze the produced data in order to understand the good or bad progress in factors like physical activities, sleep, sports performance and even nutritional habits.

This supported the idea within the group to create an application that shows basic personal statistics for the logged in user, getting data specifically for fitness activities recorded in Google Fit. This application is called **TrackStats**.

From the end-user perspective, the website initially shows a welcome page explaining very quickly the site propose and allowing the user sign-in using its Google credentials. This is a prerequisite to allow it seeing its own data coming from Google Fit.

For logged-in users, TrackStats shows initially a dashboard with a summary of fitness related statistics for the most common factors like total distance, burned calories, average speed, etc., aggregated for the last month. In the same page there's a calendar view showing a condensed list of activities (sessions) per day, and allowing the end-user selecting one to see more details.

The session details page shows a summary for the particular activity. Additionally it shows a map displaying activity location tracking along with a graph showing complementary measures like speed evolution and distance.

The previously described functionalities are the designed ones within the scope of this project. However, the architecture allows a consistent and organized application grow, offering more data insights, integration with a mobile web version or even extending the interface to mobile apps.

TrackStats

URL: http://trackstatsk.appspot.com

GitHub: https://github.com/miballeuk/trackstats

The goal with the prototype is building a set of basic functionalities that allows any user to see its own fitness data from the most common perspectives, to later grow according to identified needs ensuring a high usage level.

In the end, easy adapting is one of the most important features an application should have, and this one was designed with this capacity in its core.

Team Members

Name	E-Mail	GitHub user
Ekrem Yurdakul	ey2g13@soton.ac.uk	ekremyurdakul
Manos Milanos	ep1g15@soton.ac.uk	emmpets
Miguel Ballesteros	mabm1e15@soton.ac.uk	miballesuk
Napoleon Koskinas	nk5g15@soton.ac.uk	koskinap
Nick Perrakis	np4g15@soton.ac.uk	Iolaum
Sandeep Vyas	sv2g12@soton.ac.uk	viazerd
Umar Anwar	ua1g13@soton.ac.uk	umar786

Tools and Techniques

Coordinating a group of seven contributors is a challenging task due to the many activities each person has, lectures, deadlines, as well as different working styles.

From mid-October the team defined a weekly slot to have a sync meeting. Following basic Project Management guidelines, the meeting focused on getting a current status of all project related activities, discuss about priorities and concerns, and define the actions to follow for the next 7 days.

There's a record of all meeting minutes in a shared notebook as well as additional information about the initially defined project phases: Planning, Development, Test, Deployment and Presentation.

By the end of October the final idea was chosen, assigning at the same time the roles each person will play according to its strengths and interests. Three people focused on the application layer development, three in the front end development and one in project management and supporting the development process of both parts.

The first step was creating a simple demo application in Google App Engine to understand its features and limitations. This along with publicly available examples allowed us to have a more clear idea about what is reasonable to do and the potential project dimensions.

Later, GitHub was introduced to the team for which it was important to explain the concepts it is based on. Thus, with basic concepts understood and a small practice, the whole team was ready to start the developing phase.

No particular IDE was requested to be used. Nobody in the team had project experience with Python so this was left open for experimentation. In the practice, some members used an IDE like PyCharm, while others simply used a text editor with language highlight like Notepad++.

Communication was held by different means, depending if it was one-to-one or one-to-all. In general it was always possible to have a fluent conversation, and synchronizing key aspects like data sharing between the services and front-end layers.

Tools

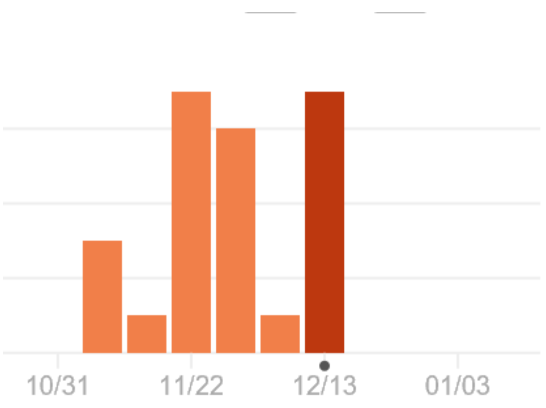
Team Collaboration	
OneNote	Shared notebook for meeting minutes and documentation.
Facebook	Group for sharing ideas and general communication
Skype	Virtual meetings
E-Mail	General communication
Development	
Notepad++	Text editor that recognizes many programming languages.
PyCharm	IDE for Python
Sublime Text 2	Text editor with programming language highlighting.
Google App Engine Launcher	Application that emulate the GAE locally for development proposes.
GitHub	Repository for the app code, deliverables and other general information.
Deployment	
Google App Engine	Hosting platform managed by Google as part of its cloud offer.

Relevant Statistics

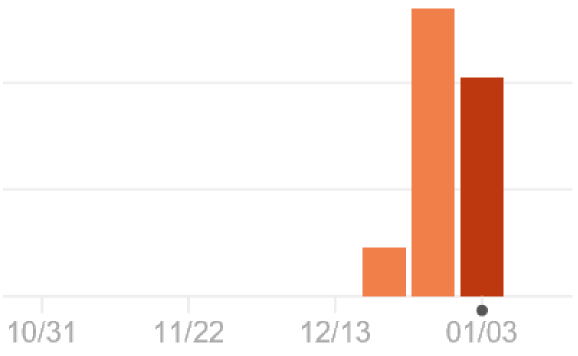
As per the dynamic the team had, it is hard to have consistent statistics for the whole process.

After creating an initial project in which all of the team members worked during some weeks, the whole architecture was changed when adopting the Django pattern. This change was delivered using a new GitHub repository.

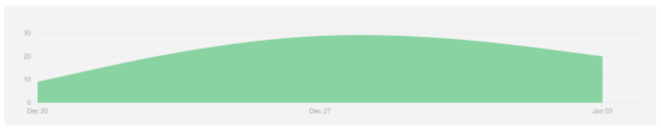
Previous GitHub repository

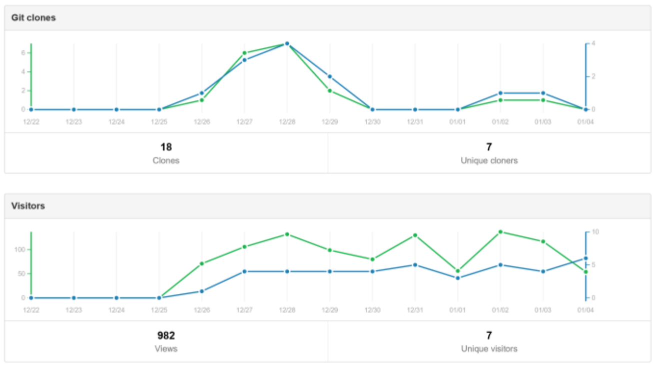


Current GitHub repository



This is the contribution for the last weeks for the new repository.





In general terms, there is an approximation in the code lines number for the ones created by the team from scratch for each language.

JavaScript	178
Python	540

Design and Implementation

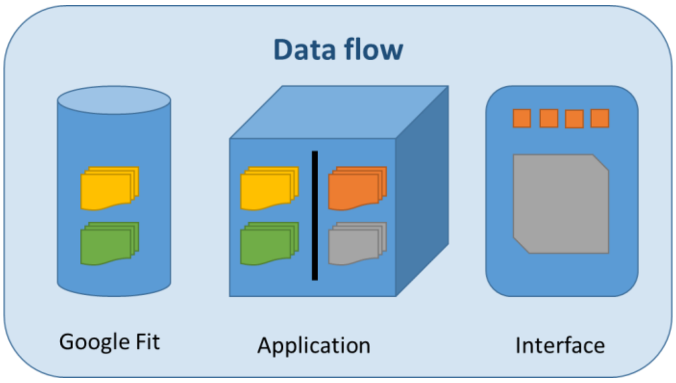
TrackStats as a starting point uses an existing service that stores and models logging data from different fitness and health sensors, called Google Fit. This service is originally designed to work as the backend for a broad range of devices requiring to store the collected data from a sensor like GPS location, heartbeat, distance, speed, consumed calories, weight, amongst others.

At the same time, Google Fit makes this data available to mobile applications or websites with the aim to let its users analyze it and provide insights to improve their wellbeing. This was the identified opportunity within the project to develop the exposed application.

In a way, Google Fit is a low level repository offering a very generic and noisy data schema requiring data consumers to group, aggregate or combine it at its own convenience. **TrackStats** then uses its own approach to read and interpret the data to show it in different views and applying its own business rules.

TrackStats is therefore a three layer application. It uses Google API as the data provider for both, fitness and user authentication. In the application layer it has a set of REST services built on Python that on demand pull data from Google Fit, transform it and return more meaningful datasets to the front end. This last layer is a web application running on top of Django that at the same time runs in a Google Application Engine environment. Its main role is to render HTML pages that when delivered to the browser, its associated JavaScripts allow the end-user interact with the data by triggering request to the application layer depending on the used features.

The implementation of the website and REST services uses the Django Framework. This Mode-View-Controller application structure allows organizing all application components according to their nature and at the same time setting the foundations for easily extensibility.



Because all data resides in Google Fit or Google+ user profile, the application does not use a local repository and therefore the model part of the architecture is not used at this stage. However, the model makes sense when considering future features like advanced statistics or analytics that require processed data which is not provided by Google Fit as raw source.

Critical Evaluation

As described previously, making a team of seven people get to an agreement in some aspects is a challenging task. The first evidence of this was defining what the project will be about.

The whole team had several meetings with brainstorming sessions to define a scenario all team members find interesting and feasible within the coursework scope and requirements. It took almost 1 month to make a decision and define a draft architecture. This was because there were many different points of views and perceptions about what would be a good idea to develop.

Team K is a mixed group residing amongst undergraduate and master's students. The starting point for each team member was different considering that some of them have no computer science backgrounds or programming experience. Writing code, defining the application architecture and handling some terms and concepts were aspects needing a special treatment and eventually causing some delays.

This skills mix caused some misunderstandings. For example, if a member with high programming skills requested something to one with not high programming skills, there were some implicit assumptions making the situation look good immediately. When the expectations started to differ from the provided results both parties realized about the existing misunderstanding. This repeated situation several times during the project development caused a significant productivity impact.

A project plan and a strategy were defined as part of the basic project management activities. Those were initially followed giving the impression that the project was in good track and status. However, as the deadline was closer, many of these project activities were not aligned with the expectations and therefore had to be tackled at the same time with other module assignments.

By the mid project life, some tasks were assigned to have early small functional pieces of the project. These tasks not always were completed and its later execution was not enforced, creating a limbo zone that later required quick actions to solve them.

Project Management approach initially relied on trusting team members to do their assigned tasks with a broad flexibility margin. As the deadline got closer, the approach changed to enforce tasks and demand quick results to avoid blocking other people tasks. Even if the used approach avoids conflicts between members, it is not good for getting the tasks done at the expected time. The lesson learned in this sense is to use a mixed approach in a future opportunity.

In general terms, the team got the expected synergy in a very advanced stage, increasing dramatically the productivity only during the last two weeks.