**IFB299 – Application Design and Development**

Zaine Ashe Portfolio 09469010

Unit: IFB299 - Project Design and Development

Group 114

Tutor: Ahmed Shoeb Talukder

Tutorial: 6pm Thursday

Version: 1.3

Date 22/09/2017

Contents

[Artefact 1 – Sprint 1 Burndown Chart 3](#_Toc493877676)

[Artefact 2 – Database Implementation 4](#_Toc493877677)

[Artefact 3 – Justification of System Architecture 6](#_Toc493877678)

[Artefact 4 – Page implementation 7](#_Toc493877679)

[Artefact 5 – Data Creation Testing 10](#_Toc493877680)

[Reference 11](#_Toc493877681)

Links index:

Artefact 1 Sprint 1 burndown chart: <https://bitbucket.org/leexsen/ifb299/src/2ae4beb720be9c6472b0a978473abaffdad7350a/n9469010/Artefact_1_Sprint_1_burndown_chart/?at=master>

Artefact 2 Database Implementation: <https://bitbucket.org/leexsen/ifb299/src/2ae4beb720be9c6472b0a978473abaffdad7350a/n9469010/Artefact_2_Database_implementation/?at=master>

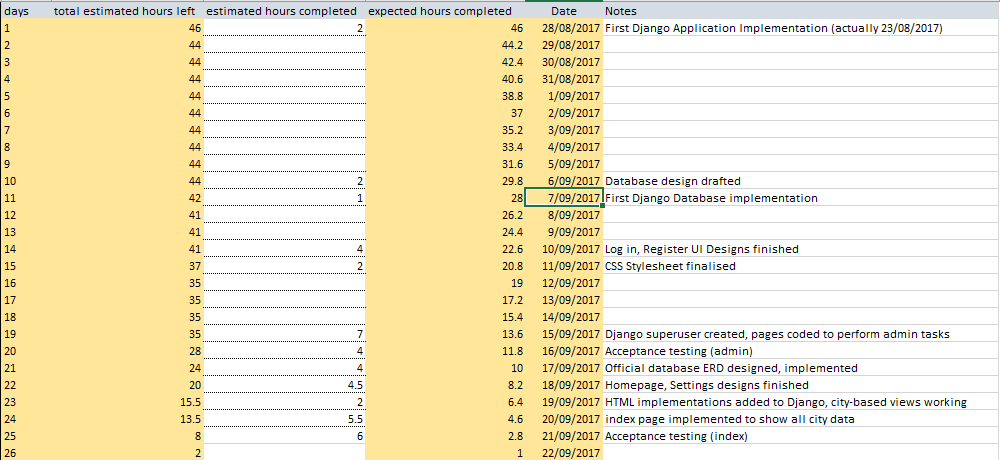
Artefact 3 Justification of System Architecture: <https://bitbucket.org/leexsen/ifb299/src/2ae4beb720be9c6472b0a978473abaffdad7350a/n9469010/Artefact_3_Justification_of_System_Architecture/?at=master>

Artefact 4 Page Implementation: <https://bitbucket.org/leexsen/ifb299/src/2ae4beb720be9c6472b0a978473abaffdad7350a/n9469010/Artefact_4_Page_Implementation/?at=master>

Artefact 5 Data Creation Testing: <https://bitbucket.org/leexsen/ifb299/src/2ae4beb720be9c6472b0a978473abaffdad7350a/n9469010/Artefact_5_Data_Creation_Testing/?at=master>

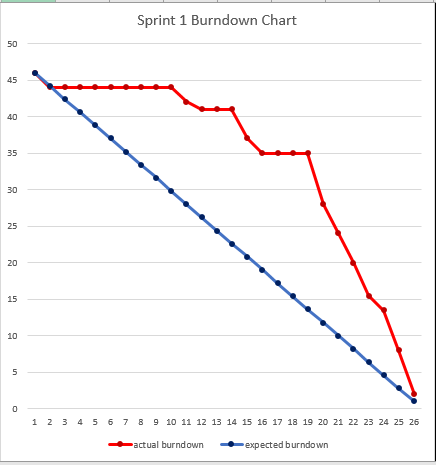
# Artefact 1 – Sprint 1 Burndown Chart

Bitbucket link (n9469010/Artefact\_1\_Sprint\_1\_Burndown\_Chart): <https://bitbucket.org/leexsen/ifb299/src/2ae4beb720be9c6472b0a978473abaffdad7350a/n9469010/Artefact_1_Sprint_1_burndown_chart/?at=master>



*Figure 1 – Sprint 1 burndown day by day.*

The burndown chart for sprint 1 measures hours worked against each day. The estimated amount of time required to finish all involved tasks was approx. 46 hours, interspersed throughout the tasks detailed in the release plan.



*Figure 2 – Sprint 1 burndown chart.*

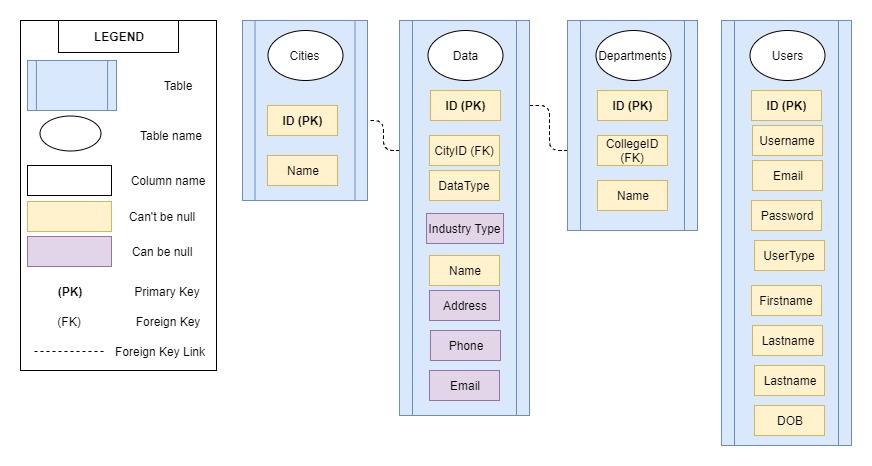
To complete all tasks, a production velocity of approx. 10h per week was estimated. Results of the sprint showed that members stagnated during the first two weeks of the sprint, but recovered lost productivity throughout the latter half.

At the end of the sprint, two tasks remained uncompleted (map API integration and acceptance testing for Create/Modify Cities). This left 2-4hrs of estimated work unfinished, as shown in figure 2.

# Artefact 2 – Database Implementation

Bitbucket link (n9469010/Artefact\_2\_Database\_Implementation): <https://bitbucket.org/leexsen/ifb299/src/2ae4beb720be9c6472b0a978473abaffdad7350a/n9469010/Artefact_2_Database_implementation/?at=master>

City data was normalised to reduce the number of redundant tables. Figure 3 shows a clear data correlation within assessment specification. Each piece of data comes with its own name, data type and contact information. The two edge cases of this normalisation are ‘Departments’ in colleges and ‘Industry type’ in industries. These edge cases can be mitigated by utilising ‘null’ data in the entries that don’t accommodate them.

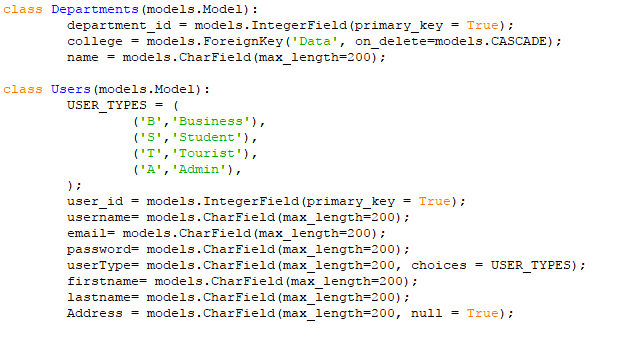


*Figure 3 – First draft database design.*

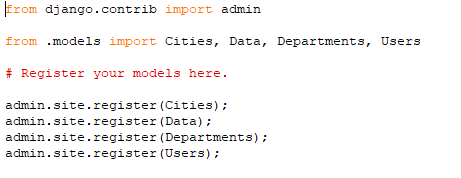
C:\Users\ZainePC\AppData\Local\Microsoft\Windows\INetCache\Content.Word\data_structure_flow.png

*Figure 4 – First draft data diagram.*

This design was implemented through Python script in the Django Framework. Each of the four individual tables were defined by their own separate Python Class. Figure 5 details the models.py and admins.py files involved on the following page.







*Figure 5 – models.py (first two images) and admins.py (third image) files from Project directory of group 114 repository.*

# Artefact 3 – Justification of System Architecture

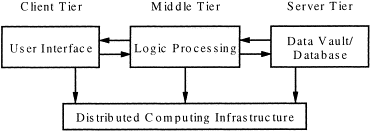
Bitbucket link (n9469010/Artefact\_3\_Justification\_of\_System\_Architecture): <https://bitbucket.org/leexsen/ifb299/src/2ae4beb720be9c6472b0a978473abaffdad7350a/n9469010/Artefact_3_Justification_of_System_Architecture/?at=master>

IFB299 Group 114’s Smart City solution will follow a Three-Tiered computational architecture. The project will require the use of database & Django functionality to accomplish set user goals. The final product will be able to interface with a server-stored database, adding and modifying data entries at the whim of the administrator user.

The product will also need to be able to perform complicated logic processing independent of database interface, while still having access to server-side security.

According to D. Tony Liu and William Xu’s paper, *A review of web-based product data management systems*, this kind of functionality can only be accommodated by an architecture with three or more tiered layers. Xu and Liu explain how ‘logic processing’ is an individual component which allows user requests to be personalised [2001, 2.2 Evolution of PDM methodology. Paras. 17-19].

The Smart City project will benefit greatly from a faster, more personalised experience, given the large audience scope implied by the assignment specification.

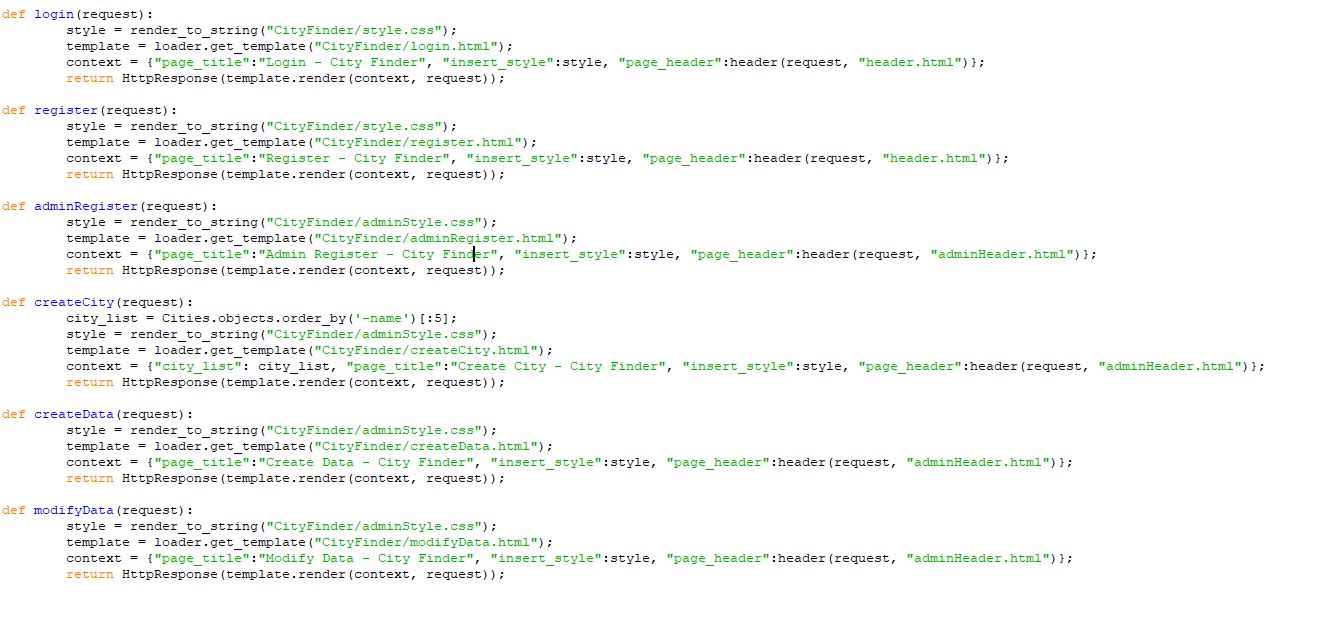


*Figure 6. Simple design of a three-tiered architecture* [adapted from *Computers in Industry* by Liu, D. T. & Xu, X. W, figure 3].

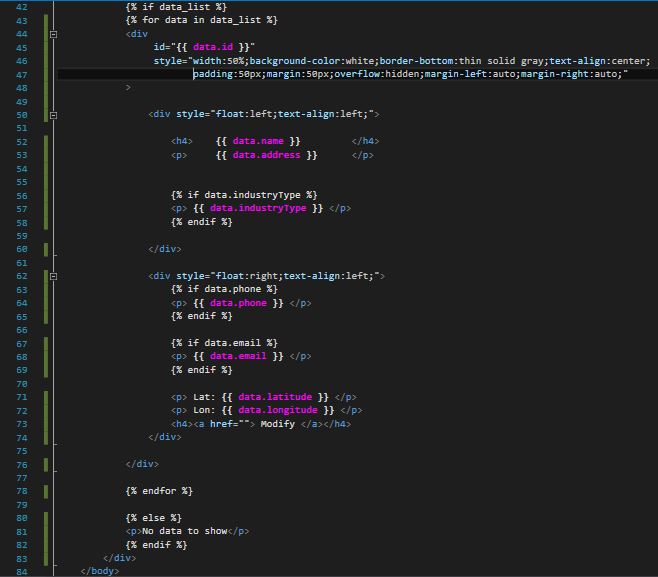
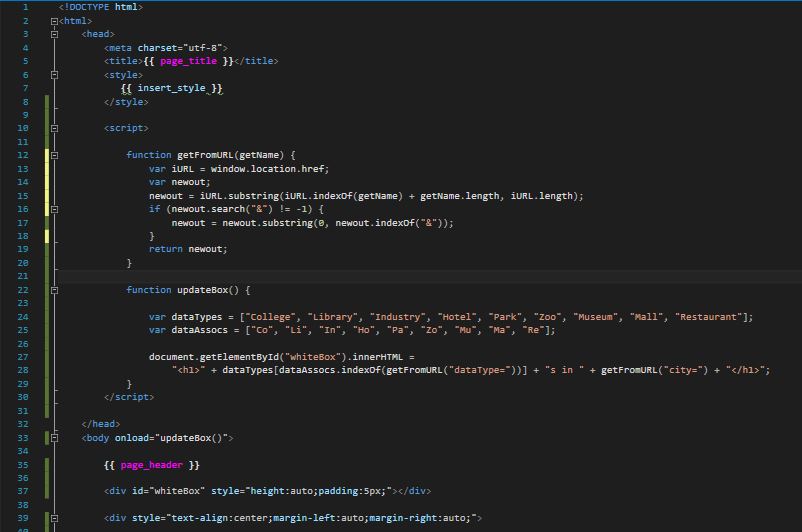
# Artefact 4 – Page implementation

Bitbucket link (n9469010/Artefact\_4\_Page\_Implementation): <https://bitbucket.org/leexsen/ifb299/src/2ae4beb720be9c6472b0a978473abaffdad7350a/n9469010/Artefact_4_Page_Implementation/?at=master>



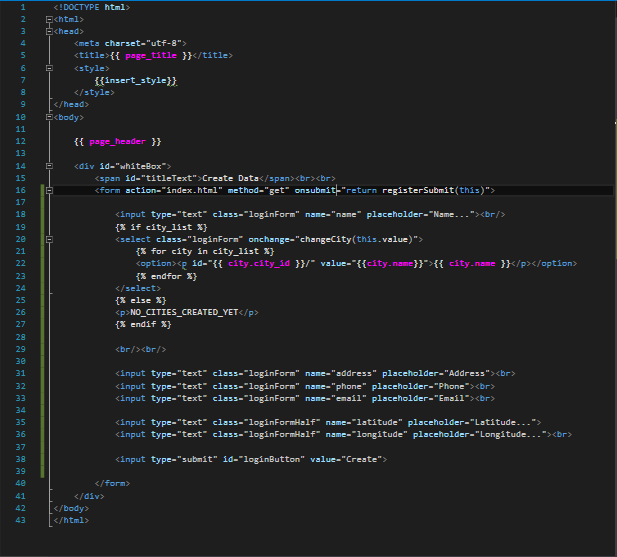


*Figure 7 – views.py file used for implementation in Project/CityFinder directory.*

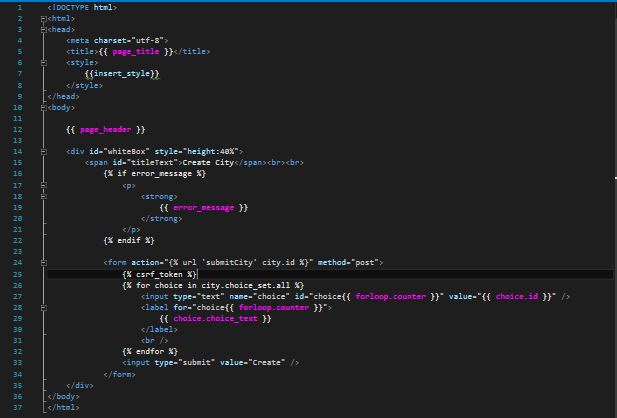


*Figure 8 – Index.html template used in Project implementation. This page displays all data stored in the Data table of the ifb299\_cityfinder MySQL schema.*

Django’s views.py file was used to interface HTML templates with MySQL storage. Each page was defined as separate methods within the views.py file, which parse request information and return http responses.



*Figure 9 – createData.html file used in Project implementation, which is used as a replacement for Django’s in-built schema management tool.*



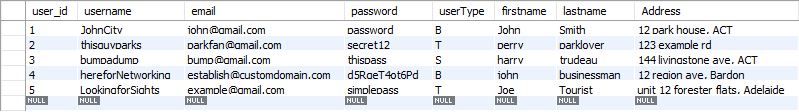
*Figure 10 – createCity.html file used as a template in the Project/CityFinder/templates/CityFinder directory.*

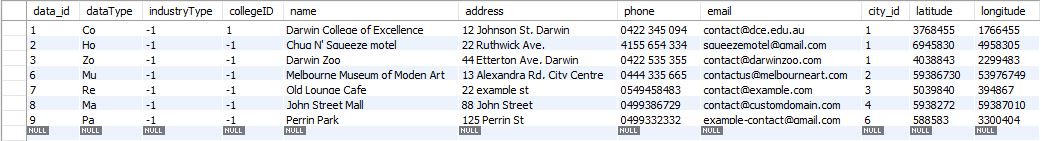
In the .html template files, specialised Django notation was used to parse data from the templates Python-based context. Any section of a HTML template which utilises the {{}} symbols is making use of the ‘context’ variable within its corresponding views.py method.

Figures 8 through 10 detail the three HTML templates developed by team member 9469010. Namely: ‘Index’, ‘createData’ and ‘createCity’. These files were built using the prior design and CSS work of other group 114 members.

# Artefact 5 – Data Creation Testing

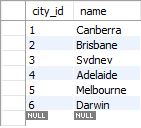
Bitbucket link (n9469010/Artefact\_5\_Data\_Creation\_Testing): <https://bitbucket.org/leexsen/ifb299/src/2ae4beb720be9c6472b0a978473abaffdad7350a/n9469010/Artefact_5_Data_Creation_Testing/?at=master>





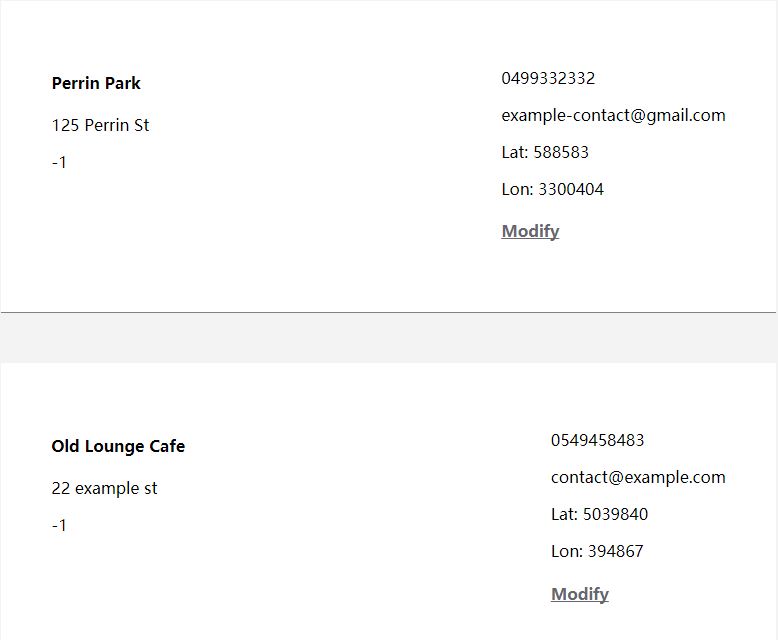
*Figure 11 – Data and Users tables in the ifb299\_cityfinder schema, filled with example data.*

Example data was entered using Django’s built in admin page to test index page and header implementation. The data entered in these examples aren’t based on real users or data entries, and are only used for prototyping.



*Figure 12 – Cities table of ifb\_299 schema filled with Australian capital cities as example data.*

The data was retrieved through MySQL Workbench 6.3 CE, and stored in separate .sql files for artefacts.



*Figure 13 – contents of the Data table on display in the index.html template. This screenshot displays two example entries, and the data they contain. (IDs:7 and 9).*

# Reference

Liu, D. T., & Xu, X. W. (2001). A review of web-based product data management systems. *Computers in industry*, *44*(3), 251-262.