Personal Portfolio Template

(Group 6 – Sem1 2018)
(Nick Pettigrew – n9654488)
(Scrumbois)

Table of Contents

PERSONAL PORTFOLIO TEMPLATE **1**

(GROUP 6 – SEM1 2018) **1**

(NICK PETTIGREW – N9654488) **1**

(SCRUMBOIS) **1**

ARTEFACT 1 ..... **2**

DRAT, AND FINAL USER STORIES REPOSITORIES: SCRUM MASTER/PRODUCT OWNER ARTEFACT 2

ARTEFACT 2 ..... **4**

USER INTERFACE DESIGN & TEMPLATING: SCRUM MASTER/DEVELOPMENT TEAM ARTEFACT 4

ARTEFACT 3 ..... **6**

DJANGO PROJECT APP/PAGE TEMPLATE: DEVELOPMENT TEAM ARTEFACT 6

ARTEFACT 4 ..... **8**

PYTHON CODING WITHIN DJANGO FRAMEWORK: DEVELOPMENT TEAM & SCRUM MASTER ARTEFACT 8

ARTEFACT 5 ..... **10**

SPRINT & RELEASE PLANNING/ESTIMATION TEMPLATE: SCRUM MASTER ARTEFACT 10

ARTEFACT 6 ..... **11**

RELEASE & SPRINT PLAN: SCRUM MASTER/PRODUCT OWNER ARTEFACT 11

ARTEFACT 7 ..... **13**

GROUP ORGANISATION, SCHEDULING, COMMUNICATION: SCRUM MASTER ARTEFACT 13

Artefact 1

Drat, and Final User Stories Repositories: SCRUM Master/product owner Artefact

(Description about what the artefact is in general)
The User Story Repositories (in the form of a power-point pack) combined all user stories collected in a single google docs page and collated them in a single document. A colour coding system was implemented by the scrum master to distinctly differentiate the ownership and responsibility of each user story within the repository. These user stories were then categorised, and presented in a meaningful order in the required template.
Multiple versions of this document (drafting with version control, and FINAL) were formulated based on feedback received from the group members, to ensure consistency and that the final submission was reflective of what the group believed was a representation of the final product.

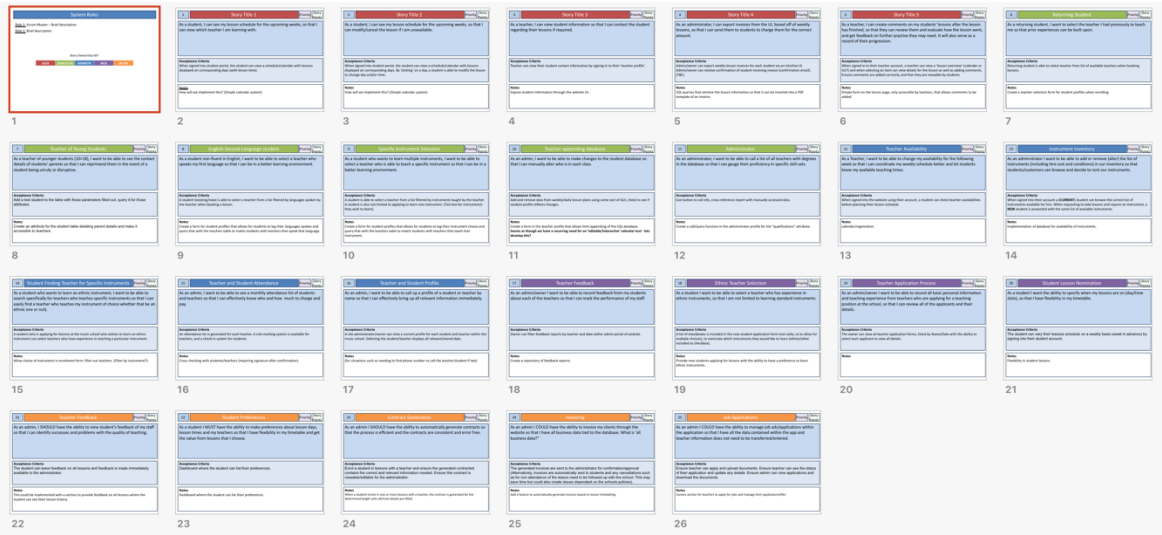
(Brief description of how it was used/contribution to the project)
The user stories formed the foundation of what the project is to become during both sprints. In referring to these stories, the group is able to ensure that what is being developed is in line with the functionality and acceptance criteria as depicted in each story.
In addition to this, the user stories also provided a sense of how much effort is to be spent to achieve the acceptance criteria for each story.

(Screen capture of where it's used – see next page)

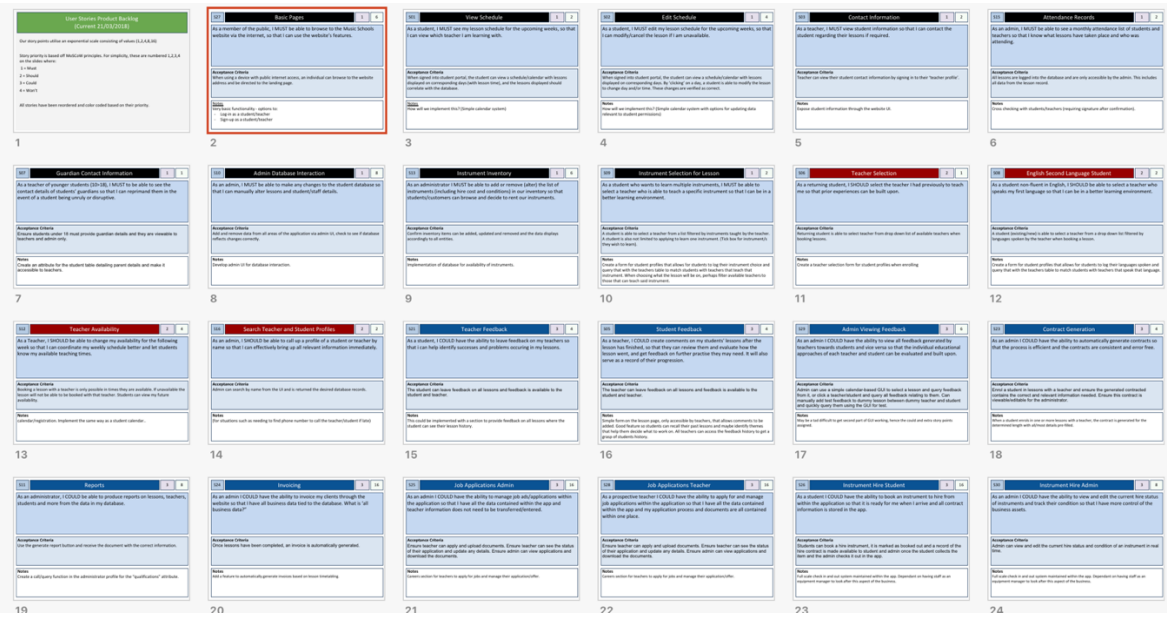
Draft Versioning and ownership

Story Ownership KEY

ALEX	SEBASTION	KENNETH	NICK	JACOB



Final User Story Repository



Artefact 2

User Interface Design & templating: SCRUM Master/development team Artefact

(Description about what the artefact is in general)
The User Interface Design artefact was created as a template (and early iteration mock-up) which incorporated each user requirement as described in the User Stories repository.

(Brief description of how it was used/contribution to the project)
In creating both a template page, and subsequent mock-up UI pages (based on this template) creation of the actual html Django template (and pages based on this template) were able to be coded in html5 and css3 with ease.
With the non-functional template and mock-up's created, the draft UI was presented to the design team for feedback, at which point slight modifications were made to further improve upon.

(Screen capture of where it's used – see next page)

User Interface Design and Template

HEADER [width: 95vw, height: 5vw, clear: both]

Sub-Header [width: 95vw, height: 5vw, clear: both]

Left Nav [width: 18vw, height: 80vh, float: left, padding: 3px]

Main Content [width: 55.5vw, height: 80vh, float: left, padding: 3px]

Right Nav [width height: 80vh, flo padding: 3p

Footer [width: 95ve, height: 5vh, clear: both]

Welcome to Scrumbois' Music School

Music School

Username

enter username here

Password

enter password here

Login

Register

Page Produced by Scrumbois.

Welcome to Scrumbois' Music School

MUSIC SCHOOL

WELCOME, {dynamic firstName}

Scheduale

Profile

Equiptment

Lesson Contract

Payment

My Equiptment

Instrument	Picked Up	Due Back	Condition	Hire Cost
Saxaphone	dynamic	dynamic	dynamic	dynamic
Flute	dynamic	dynamic	dynamic	dynamic
Bass Guitar	dynamic	dynamic	dynamic	dynamic

Hire Equiptment

Instrument:	dropdown
Pickup Date:	dropdown
Return Date:	dropdown
Cost:	calculated

Hire

Page Produced by Scrumbois.

Welcome to Scrumbois' Music School

MUSIC SCHOOL

WELCOME, {dynamic fi

Scheduale

Profile

Equiptment

Lesson Contract

Payment

My Scheduale

Instrumen	Lesson Dat	Time	Room	Teache
Flute	20-Apr	10:30	A	Mr John
Saxaphone	22-Apr	12:00	C	Miss Jane

Book Lesson

Instrument:	dropdown
Date:	dropdown
Time:	dropdown
Teacher:	dropdown

Confirm

Page Produced by Scrumbois.

Welcome to Scrumbois' Music School

MUSIC SCHOOL

WELCOME, {dynamic firstName}

Scheduale

Profile

Equiptment

Lesson Contract

Payment

My Profile

NAME:	dynamic content
ADDRESS:	dynamic content
PHONE NUM:	dynamic content
DOB:	dynamic content

Page Produced by Scrumbois.

Artefact 3

Django Project App/Page Template: Development Team Artefact

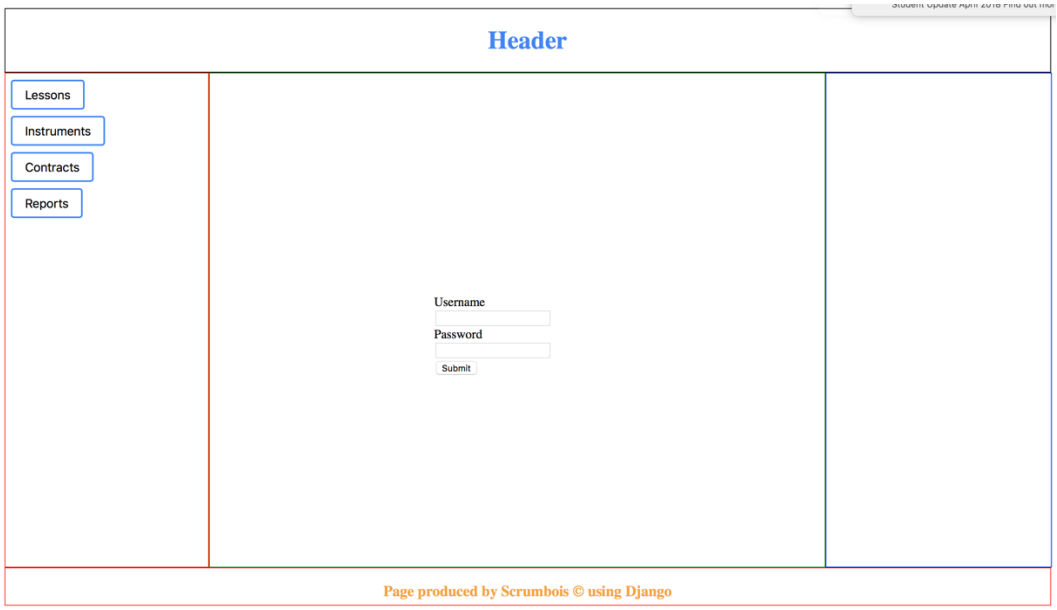
(Description about what the artefact is in general)
The Django project app/page template allows any subsequent pages that are required to deliver the functional requirements of the Music School page to be created with minimal additional html5 code. Additionally, it means that each page created within the Django project is guaranteed to have the same 'look and feel' as each page.

(Brief description of how it was used/contribution to the project)
After creating the application (schedule) within the Django Project (v3, mma_3) which acts as the front-end (user facing) of the web-app a html template was created. To assist in simplicity within the template file, a static external ccs3 style sheet was also created to provide the html template with consistent styling and a basic yet functional structure (through floating divs). After combining these two files to formulate the basic template file, an index page was created which doubles as the landing/login page. For improved usability, this page was hard-coded as the default redirected landing page within the root ulrs.py file.

Despite not being used in the final implementation of the web-apps sites, these html and css 'templates' framework was key during the early development stages of the project.
There group identified a requirement to have an initial basic, yet functional framework as a front-end so that the major back-end developed by the Computer Science students had a suitable testing environment. It was decided that as the scrum master, it was appropriate that I be responsible for developing the foundational/test environment so as to lay out the direction that the group stick to (avoiding conflicting development strategies). Further development was (largely) a responsibility of the development team (CS team members).

(Screen capture of where it's used– see next page)

Django base page template + HTML5 & CSS3 code.



styles2.css UNREGISTERED

```
/*Page structure here*/
#header {
  width: 95vw;
  height: 10vh
  clear: both;
  border: 1px solid black;
}

#left-nav {
  width: 18vw;
  height: 80vh;
  float: left;
  border: 1px solid red;
  padding: 3px;
}

#main-contnet {
  width: 55.5vw;
  height: 80vh;
  float: left;
  border: 1px solid green;
  padding: 3px;
}

#right-nav {
  width: 20vw;
  height: 80vh;
  float: left;
  border: 1px solid blue;
  padding: 3px;
}

#footer {
  width: 95vw;
  height: 6vh;
  clear: both;
```

Line 1, Column 1 Spaces: 2 CSS

index.html UNREGISTERED

```
{% extends "general_template.html" %}

{% block left_nav %}
<table align="centre">
        <tr>
            <td>
            <button class="button button1">Lessons</
            button>
            </td>
        </tr>
        <tr>
            <td>
            <button class="button button1">
            Instruments</button>
            </td>
        </tr>
        <tr>
            <td>
            <button class="button button1">Contracts
            </button>
            </td>
        </tr>
        <tr>
            <td>
            <button class="button button1">Reports</
            button>
            </td>
        </tr>
    </table>

{% endblock %}

{% block body_content %}
<table style="padding-left: 35%; padding-top: 35%">
    <tr>
        <td></td>
```

Line 1, Column 1 Tab Size: 4 HTML

general_template.html UNREGISTERED

```
<div id="header">
    {% block body_title %}
    <h1>Header</h1>
    {% endblock %}
</div>

<div id="left-nav">
    {% block left_nav %}
    {% load static %}
    <img src="{% static 'images/instruments.jpg' %}"
     height="190" width="200">

    <table align="centre">
        <tr>
            <td>
            <button class="button button1">Button 1
            </button>
            </td>
        </tr>
        <tr>
            <td>
            <button class="button button1">Button 2
            </button>
            </td>
        </tr>
        <tr>
            <td>
            <button class="button button1">Button 3
            </button>
            </td>
        </tr>
        <tr>
            <td>
            <button class="button button1">Button 4
            </button>
            </td>
```

Line 1, Column 1 Tab Size: 4 HTML

Artefact 4

Python Coding within Django Framework: Development Team & SCRUM Master Artefact

(Description about what the artefact is in general)
To progress the Music School web-app past the point of the initial framework (atomically generated upon starting the Django project, mma_3), a number of applications were created within the project.

To enable the intended functionality of the website (namely having each page 'mapped' to their respective 'views' using URL's), the automatically generated views.py and urls.py python files had to be modified.

By customising these python files for each application within the project, each page generated from the template (discussed in artefact 3) could be viewed via links, or by browsing to the address identified by its URL pattern.

(Brief description of how it was used/contribution to the project)
The following steps were followed after creating and including a new HTML web-page within each of the projects apps based on the template created by myself. As an Information Student (and SCRUM master) this (task) was my contribution to this aspect of the Django development component of the project:

- A view was defined within the respective views.py file, and named according to its html page.
- A URL pattern was then specified based on the path of the template for the page.
- Within the root site folder, the default landing page was set as the index.html page within the urls.py file. This was done so that when browsing (locally) to the page, that the landing page at the time would be the default page loaded.

Despite not being used in the final implementation of the web-app (and sites) this framework was key during the early development stages of the project. It was decided that (as a team) there was a requirement to have an initial basic, yet functional framework as a front-end so that the major back-end being developed by the Computer Science students has a suitable testing environment.

Further site iterations were refinements based on this framework.

(Screen capture of where it's used – see next page)

Python source-code to enable page linking & functionality

urls.py

```python
"""mysite URL Configuration

The `urlpatterns` list routes URLs to views. For more information please see:
    https://docs.djangoproject.com/en/2.0/topics/http/urls/
Examples:
Function views
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  path('', views.home, name='home')
Class-based views
    1. Add an import:  from other_app.views import Home
    2. Add a URL to urlpatterns:  path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
"""
from django.contrib import admin
from django.urls import include, path
from django.views.generic import RedirectView
from django.conf import settings
from django.conf.urls.static import static

urlpatterns = [

    path('', include('schedule.urls')),
    path('admin/', admin.site.urls, name='admin'),
    path('accounts/', include('accounts.urls'), name='accounts'),
    path('', RedirectView.as_view(url='/schedule/')),

] + static(settings.STATIC_URL, document_root=settings.STATIC_ROOT)
```

Line 27, Column 5 Spaces: 4

views.py

```python
from django.shortcuts import render
from django.views.generic import TemplateView
from django.http import HttpResponse
from .models import Lesson
from django.contrib.auth.models import User

#defining index page view
def index(request):
    lessons = Lesson.objects.all()
    return render(request, 'schedule/index.html', context={'lessons':lessons})

def instruments(request):
    return render(request, 'schedule/instruments.html')

def profile(request):
    return render(request, 'schedule/profile.html')

# Landing Page View, from generic_template.html
class HomePageView(TemplateView):
    def get(self, request, **kwargs):
        return render(request, 'index.html', context=None)
```

urls.py

```python
from django.urls import path
from django.conf.urls import url
from . import views

urlpatterns = [
    path('', views.index, name='index'),
    path('schedule', views.index, name='index'),
    path('instruments', views.instruments, name='instruments'),
    path('profile', views.profile, name='profile'),
    url(r'^$', views.HomePageView.as_view())
]
```

Line 1, Column 1 Tab Size: 4 Python

models.py

```python
class Lesson(models.Model):

    # lesson time
    # date and time is simply entered currently with no checks for double ups
    # fix this later!!!

    lesson_time = models.DateTimeField('Lesson Time') #Lesson Time


    # lesson duration
    # stored as CHAR as it takes far less room than INT and they are not used for arithmeic

    LESSON_DURATION_CHOICES = (
        ("30", '30 Minutes'),
        ("60", '1 Hour'),
        ("90", '1.5 Hours'),
        ("120", '2 Hours')
    )

    lesson_duration = models.CharField(
        max_length = 3,
        choices = LESSON_DURATION_CHOICES,
        default = "60"
    )


    # lesson location
    LESSON_LOCATION_CHOICES = (
        ("A0", 'Room A0'),
        ("A1", 'Room A1'),
        ("A2", 'Room A2'),
        ("A3", 'Room A3'),
        ("A4", 'Room A4')
    )

    lesson_location = models.CharField(
        max_length = 2,
        choices = LESSON_LOCATION_CHOICES,
        default = "A0"
    )

    #add foreign keys for teacher and student involved in lesson

    lesson_student = models.ForeignKey("accounts.User", limit_choices_to={'groups__name': "Stu

    lesson_teacher = models.ForeignKey("accounts.User", limit_choices_to={'groups__name': "Tea
```

Line 1, Column 1 Spaces: 4

Artefact 5

Sprint & Release Planning/estimation Template: SCRUM Master Artefact

(Description about what the artefact is in general)
The Sprint & Release Planning Spreadsheet was formulated as a means to outline which stories would be assigned to each sprint, and detail how much effort would be required (in terms of story points) to deliver each.

(Brief description of how it was used/contribution to the project)
The spreadsheet was largely used as a planning tool to ensure the group had distributed the required user stories throughout the duration of both sprints in a way that would align with the teams likely velocity.
The spreadsheet was broken down into the number of stories in each sprint, for each release.

(Screen capture of where it's used)
Planning & Estimation Template

Stories	R1 Sprint 1	R1 Sprint 2	Story Points		
27	Landing Page		6		
1	View Schedule		2		
2		Edit Schedule	4		
3	Contact Info		2		
TOTAL	10	4	14		
15	Attendance Records		2		Application Core
10	Admin DB interaction		8		Administrator Features
16	Search Teacher & Stu Profiles		2		Student Features
21		Teacher Feedback	4		Teacher Features
23		Contract Generation	4		Additional Features
	END SPRINT ONE (R1)	END SPRINT ONE (R2)			
TOTAL	12	8	20		
6	Teacher Selection		1		
8		ENG 2nd Language	2		
9	Instrument Selection Lesson		2		
26		Instrument Hire	16		
TOTAL	3	18	21		
12	Teacher Availability		4		
5	Student Feedback		4		
7	Guard Contact Info		1		
	END SPRINT TWO (R1)				
TOTAL	9	-	9		
11		Report export	8		
24		Invoicing	16		
		END SPRINT TWO (R2)			
25		Job Applications	16		
13		Instrument Inventory	6		
		END SPRINT ONE (R3)			
TOTAL	-	46	46		
SPRINT TOTALS	34	76			
		GRAND TOTAL	110		

Artefact 6

Release & Sprint Plan: SCRUM Master/product owner Artefact

(Description about what the artefact is in general)
The release & sprint plan was a culmination of many sprint planning activities that the group completed together. In completing a number of planning activities (story card poker, negation of tasks and priorities, and task delegation) the team was able to agree on what needed to be included within each release and respective sprints.

Each release within the release plan provides a high-level overview of the functionality the tasks delivered via each release feature set. These were categorised according to either where the functionality lay (front or back end) according to the system architecture, or who the intended end-users were.

(Brief description of how it was used/contribution to the project)
The Sprint and Release Plan was used throughout the entirety of sprint one as a point of reference for deliverables due within agreed upon time periods. In terms of the development team, when discussions were being made around what functionality would be delivered, the release plan was referenced to ensure the project source code was in-line with the agreed scope.

My personal contribution to the project via this artefact included collaborating each team members user stories (as per the 'Final User Story Repository') within this document. Once included, the 'Sprint & Release planning spreadsheet' was consulted to ensure that the story points assigned to each story were accurate and appropriate (team consulted when changes required). A brief summary of the feature sets included within each sprint was written in-line with the provided template to keep the document well presented and professional.

After breaking down the sprint plan into the six feature set deliverables, tasks were assigned and delegated to the group members according to individual strengths. As the scrum master, this was communicated clearly and regularly in order to ensure clarity and an understanding of expectations within the group.

(Screen capture of where it's used – see next page)

Sprint and Release Plan (screenshot)

Release and Sprint 1 Plan

Team Number: 6

Student Number	Team Member Name
9823204	Jacob Bullock
9709223	Sebastian Peeler
9987168	Alex Butler
9700374	Kenneth Cheung
9854488	Nick Petligrew

Tutor: Tara Capel
Workshop: Wednesday 11:00 – 13:00

Version: 1.5 - Final

Date: 21/03/2018

1

Table of Contents

Release 1 3
- Application Core 3
- Administrator Features 3
- Student Features 3
- Teacher Features 3

Release 2 4
- Application Core 4
- Administrator Features 4
- Student Features 4
- Teacher Features 4
- Additional Features 5

Release 3 5
- Job Applications 5
- Instrument Hire 5

Delivery Schedule 6

Sprint 1 7
- Story 27: Basic Pages 7
- Story 01: View Schedule 7
- Story 03: Contact Information 7
- Story 10: Administrator Database Interaction 7
- Story 16: Search Teacher and Student Profiles 8
- Story 07: Guardian Contact Information 8

2

Release Plan

Release 1

Delivery date: **01/06/2018** Total Story Points: **32**

Develop a database to function as the core of the application. Via a basic landing page, Students & Teachers will be able to view and edit their schedule and will have a profile page displaying all relevant information. The Admin will be able to edit the schedule, view lesson history and view the profiles of staff and students. There will be an instrument inventory which is editable by the admin.

Application Core

Develop the database and core app features including landing page, page profiles and a lesson schedule for all entities (No profile for administrator).

Story ID	Story Title	Story Points
27	Basic Pages 1	4
1	View Schedule 1	2
3	Contact Information 1	2
	Story Point Sub-Total:	**8**

Administrator Features

Develop the features which allow a site administrator, and product owner (Music School Owner) to make changes to the data stored within the database.

Story ID	Story Title	Story Points
15	Attendance Records	2
10	Administrator Database Interaction 1	4
16	Search Teacher & Student Profiles 1	2
	Story Point Sub-Total:	**8**

Student Features

Develop features unique to the student facing portion of the application.

Story ID	Story Title	Story Points
6	Teacher Selection	1
9	Instrument Selection for Lesson	2
21	Teacher Feedback	4
	Story Point Sub-Total:	**7**

Teacher Features

Develop the database and core app features including page profiles and a lesson schedule.

Story ID	Story Title	Story Points
12	Teacher Availability	4
5	Student Feedback	4
7	Guardian Contact Information 1	1
	Story Point Sub-Total:	**9**

3

Release 2

Delivery date: 27/07/2018 Total Story Points: **40**

The purpose of the second release is to build upon the foundation formed by completing the tasks associated with each story in the first release. This will be achieved by implementing an enriched feature set, largely containing stories that were prioritised as ones that SHOULD be available to users of the (web) application.

Application Core

Further develop the database and introduce additional features that have a direct impact on the usability and client-facing interface.

Story ID	Story Title	Story Points
2	Edit Schedule	4
	Story Point Sub-Total:	4

Administrator Features

Introduce features which allow a site administrator, and product owner (Music School Owner) to view the feedback on a given lesson left by the student and teacher. Also introduce the ability to generate contracts based on information stored within the Database.

Story ID	Story Title	Story Points
22	Admin Viewing Feedback	4
23	Contract Generation	4
	Story Point Sub-Total:	8

Student Features

Introduce features which allow a student (new, or existing) to nominate English as their second language to view and provide an interface which provides students with the ability to hire an instrument for future lessons.

Story ID	Story Title	Story Points
8	English Second Language Student	2
	Story Point Sub-Total:	2

Teacher Features

Develop feature that allows a teacher to leave feedback for the student.

Story ID	Story Title	Story Points
5	Student Feedback	4
	Story Point Sub-Total:	**4**

4

Additional Features

Additional features have been categorically prioritised as being desirable, but not pivotal to the operation and success of the product. These additional features provide the owner (and admins) to produce and export reports on lessons, teachers, and students; based on recorded and stored data. In addition, the owner will have the ability to automatically generate, and deliver invoices to their students (or guardians).

Story ID	Story Title	Story Points
11	Reports	8
24	Invoicing	16
	Story Point Sub-Total:	24

Release 3

Delivery date: 21/09/2018 Total Story Points: 40

Release 3 extends the application to include Job Applications & a real time Instrument Hire feature.

Job Applications

This feature offers the ability for prospective teachers to apply for positions and for the admin of the app to view/download applications.

Story ID	Story Title	Story Points
28	Job Application Teacher	16
25	Job Application Admin	8
	Story Point Sub-Total:	24

Instrument Hire

The instrument Hire feature allows students to book instruments to loan, check them in and out. The admin will have the ability to monitor the instruments hire status & condition.

Story ID	Story Title	Story Points
26	Instrument Hire Student	8
30	Instrument Hire Admin	8
	Story Point Sub-Total:	16

5

Delivery Schedule

Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13
Sprint 1				Sprint 2			
Release 1				Release 1			

Week 14	Week 15	Week 16	Week 17	Week 18	Week 19	Week 20	Week 21
Sprint 3				Sprint 4			
Release 2							

Week 22	Week 23	Week 24	Week 25	Week 26	Week 27	Week 28	Week 29
Sprint 5				Sprint 6			
Release 3							

6

Sprint Plan

Sprint 1

Total Story Points: 15 Total Hours: 30

Estimated Velocity: 15

Story 27: Basic Pages

Task ID	Task Description	Estimate	Taken
T01	Implement landing page	1	
T02	Implement login system	2	
T03	Implement register system	2	
T04	Implement Stylesheet and Scripts	2	
T05	Verify story is complete	1	
	Story Points: 4 Total Hours	8	

Story 01: View Schedule

Task ID	Task Description	Estimate	Taken
T06	Develop empty calendar	1	
T07	Write SQL read queries	1	
T08	Display information in calendar	1	
T09	Verify story is complete	1	
	Story Points: 2 Total Hours	4	

Story 03: Contact Information

Task ID	Task Description	Estimate	Taken
T10	Develop blank profile view	2	
T11	Display contact information on profile via SQL query	1	
T12	Verify story is complete	1	
	Story Points: 2 Total Hours	4	

Story 10: Administrator Database Interaction

Task ID	Task Description	Estimate	Taken
T13	Develop view for editing database	2	
T14	Write SQL read queries	1	
T15	Write SQL update queries	1	
T16	Write SQL delete queries	1	
T17	Verify story is complete	1	
	Story Points: 4 Total Hours	8	

Story 16: Search Teacher and Student Profiles

Task ID	Task Description	Estimate	Taken
T18	Develop view for the search form	1	
T19	Write SQL queries to search database	1	
T20	Display results in a table as links to profile	1	
T21	Verify story is complete	1	
	Story Points: 2 Total Hours	4	

Story 07: Guardian Contact Information

Task ID	Task Description	Estimate	Taken
T22	Display guardian information for users under 18 on student profile	1	
T23	Verify story is complete	1	
	Story Points: 1 Total Hours	2	

Artefact 7

Group organisation, scheduling, communication: SCRUM Master Artefact

(Description about what the artefact is in general)
As the assigned SCRUM master for the group, a concerted effort was made to ensure that, from the get-go, the team was on the same page. To achieve this outcome a communication channel was formed (via a facebook group chat) where the group members were encouraged to communicate any opinions, changes, or general notices to all group members.

This additionally included collecting meeting minutes (as a summary) of each workshop.

(Brief description of how it was used/contribution to the project)
In the two-weeks leading up to, and throughout the first sprint, the scrum master sent daily status messages to the group to make sure that the group was staying on top of the required workload and agreed upon deadlines. The below list summarises the roles and responsibilities the scrum master took upon himself throughout the project duration (to date):

- Assign tasks to team members based on identified strengths
- Negotiated work-load to ensure fairness across all team members
- Communicate openly and transparently to the group chat when making contributions to group tasks
- Request updates (daily, when required) from the team members to gain an understanding of sprint/task progress
- Create initial/version_1 files and templates for development (code-based) tasks as a way of providing a foundation and direction for the developers to follow
- Seek feedback from each team member on each others individual contributions to tasks (developmental + deliverable).
- Address team challenges as they presented themselves
    - Lack of communication in early weeks (before and after beginning of sprint) lead to group members working individually, and according to varied coding frame-works

(Screen capture of where it's used)
See following pages for group/project management efforts.

Team Group Message

Scrumbois

APR 24TH, 8:05AM

I'm going to further develop the template page, to include a little more detail (similar to our design/concept pages created in last weeks work shop)

WED 1:41PM

Hi guys, progress update from me. Template is now complete. Still looks basic, but it can now be filled with whatever content we want for each page/app 🙂

SUN 8:35PM

Kenneth

g'day guys

just an update

I am working on a revised component and flow diagram as well as a BPMN diagram which will show the entire business process model of our site

for the tute we missed last week, we also needed to have a class diagram finished and uploaded on our github too

😮 1

I'm not as sure about that stuff as you guys so some assistance in that will be great

finally, please remember to update me if yall have knocked out any other tasks

I found out that the burndown chart needs to be submitted too

TUE 10:42AM

Sorry @Kenneth Cheung I forgot to update yesterday, but I submitted 4/5 of my artefacts to the git repository. These were: Sprint&Release planning document, user stories repository file, User Interface design document, initial App/Page template within the Django project)

As a team, what do we want to have done within the next 24/48 hours? What do we think we need to hav e completed?

Type a message, @name...

Scrumbois

Options

Search in Conversation

Edit Nicknames

Change Color

Change Emoji

Notifications

Group Options

Require Admin Approval

People

Add People

Alex Butler Admin

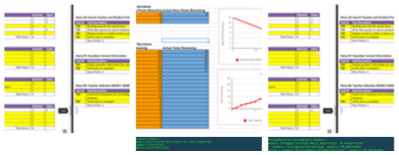
Jacob Bullock Admin

Kenneth Cheung Admin

Nick Pettigrew Admin

Sebastian Joseph Peeler Admin

Shared Photos



APR 4TH, 12:22PM

Okay so after picking around for a while and following a bunch of different tutorials and examples on the internet I decided that the project was going to get super messy and way out of control....there is actually a fair bit of work involved just to have a Skelton project/application created.

I've started from scratch on my laptop (seeing as there weren't any changes pushed to the git repository anyway)...I'll push this shortly. It is called musicschool2. I've found a really in-depth tutorial that creates an online library catalogue website with multiple levels of users and respective access levels that reports to a database (so very similar to the functionality we need to achieve). I've spend the last 3 hours reading and working through it and am up to part 3/12. I'm up to the point where the data models are created (relationships between items in the DB). @Jacob Bullock or @Alex Butler are you able to have a read through this step and try and get a UML class diagram created thats relevant for the music school? from there I can continue to progress with this tutorial

https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Models

APR 5TH, 8:37PM

Jacob

Good stuff nick, ive got a lot of work this weekend but i can definitely do a uml diagram but i may not have time until monday 😕 im gonna take a look at the project and hopefully will be able to help finish off the database before the end of next week

I've progressed a bit with the db, but only enough to continue with the general functionality of the site (databases aren't my strong suit...). I'll make sure I keep pushing changes to github (remember my current version is within MusicSchool2)

MAR 8TH, 8:12AM

Hey guys, just touching base after yesterday. We spoke with our tutor during the workshop, and the deliverables for week 5 are our user stories, and release & sprint planning doco - these NEED to be completed and submitted using the templates on blackboard (sorry Jacob haha). I've included the work we did yesterday in the week 3 folder on google drive, as well as a brief update in the todo doc but I thought it would be best to chat about how we wanted to distribute our workload as a group 🙂

Hey guys, I've had to shuffle things around a little bit to make sure we deliver what is required for the assignment. Please take a look over the current documents (week 5 folder). I've rewritten so that it is in the provided template, as I think we were complicating things having them spread across 2 docs.

MAR 10TH, 3:50PM

Hi guys, FYI I'm in the process of moving our stories (to date) from the CURRENT versions in 'week 3 overview' doco into the required template pp. I'll upload once complete

Speak up now if you have some story versions that have been updated and aren't in that location ✌️

APR 2ND, 1:52PM

Hey Bois, just touching base...how are we travelling with the first set of tasks?

Jacob

ive updated the stories as listed in my section of the homework and made the changes in the release plan which i updated to v 1.4 on google drive

👍 2

MAR 20TH, 2:22PM

Doco is looking pretty good. All we need to finish off now to hand in by tomorrow is any story's to include in Release 3 (not sure if needed....) and our Sprint 1 Plan (with estimated velocity).

WED 1:41PM

Hi guys, progress update from me. Template is now complete. Still looks basic, but it can now be filled with whatever content we want for each page/app 🙂

SUN 8:35PM

Kenneth

g'day guys

just an update

I am working on a revised component and flow diagram as well as a BPMN diagram which will show the entire business process model of our site

for the tute we missed last week, we also needed to have a class diagram finished and uploaded on our github too

😮 1

I'm not as sure about that stuff as you guys so some assistance in that will be great

finally, please remember to update me if yall have knocked out any other tasks

I found out that the burndown chart needs to be submitted too