PhoneAsId Final Release Summary

Team members

Name and Student id	GitHub id	Number of story points that member was an **author** on.
Michał Wozniak 21941097	mv740	**20(#78)+ 3(#132) + 2(#133)+ 8(#137) = 33**
Francis Côté-Tremblay 26615287	francisct	**13(#57) + 3(#168) + 1 (#136) + 13(#127) = 30**
Ahmed Dorias 26649874	ConfusedGiant	**8(#134) + 8(#135) = 16**
Harrison Ianatchkov 26607403	zzharryzz	**2(#141) + 3(#154) + 8(#158) = 13**
Simon Monière Abes 26648568	simonma1	**8(#134) + 8(#135) = 16**
Sebastian Rafique Proctor-Shah 29649727	EXPSPACE	**8(#134) + 8(#135) + 5(#138) + 8(#159) = 29**

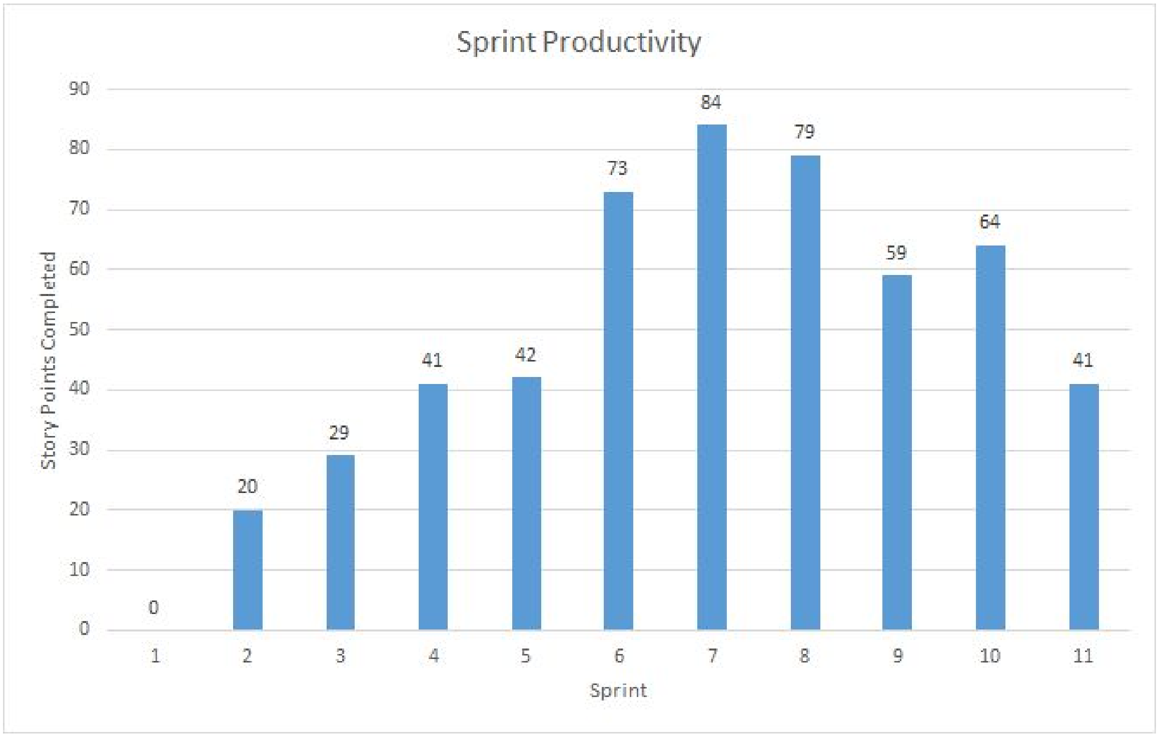
Stakeholders

Name	Github id
Jean Francois Bourgault	jfbourgault
Timothy Ni	timni

Project summary

This project is meant to virtualize the student ID system at Concordia. A mobile application will be used to allow students to readily have their student IDs with them. It will also allow them to take their own pictures to be used as their picture ID. The picture goes through an initial layer of validation by the mobile application. It is then validated in the backend by a person on a web application. The barcode on the student's ID page of the application will substitute the physical barcode, in that it can be validated by being scanned for different purposes on campus. There will also be an events feature, which will allow students and staff at concordia to organize events, keep track of desired events, as well as take attendance during events.

Velocity



Our **velocity** is 532 points over 11 iterations = **48.36 user story points / iteration**

Total: 26 stories, 205 points over 13 weeks*
Iteration 1 (0 stories, 0 points)
Iteration 2 (4 stories, 20 points)

Iteration 3, Release 1 (3 stories, 29 points)
*We started our first sprint 3 days before it was due because we were solidifying our project with IITS.

Iteration 4 (5 stories, 41 points)
Iteration 5 (6 stories, 42 points)
Iteration 6, Release 2 (8 stories, 73 points)

Total: 32 stories, 222 points, over 7 weeks
Iteration 7 (10 stories, 84 points)
Iteration 8 (11 stories, 79 points)
Iteration 9, Release 3 (~~9~~ 11 stories, ~~71~~ 59 points)

Total: 14 stories, 105, over 5 weeks
Iteration 10 (~~9~~ 8 stories, ~~70~~ 64 points)
Iteration 11, End of Project (6 stories, 41 points)

Final Release Summary

The last 2 sprints leading to the final release of our project, mainly focused on documentation, localization, and bug fixing.

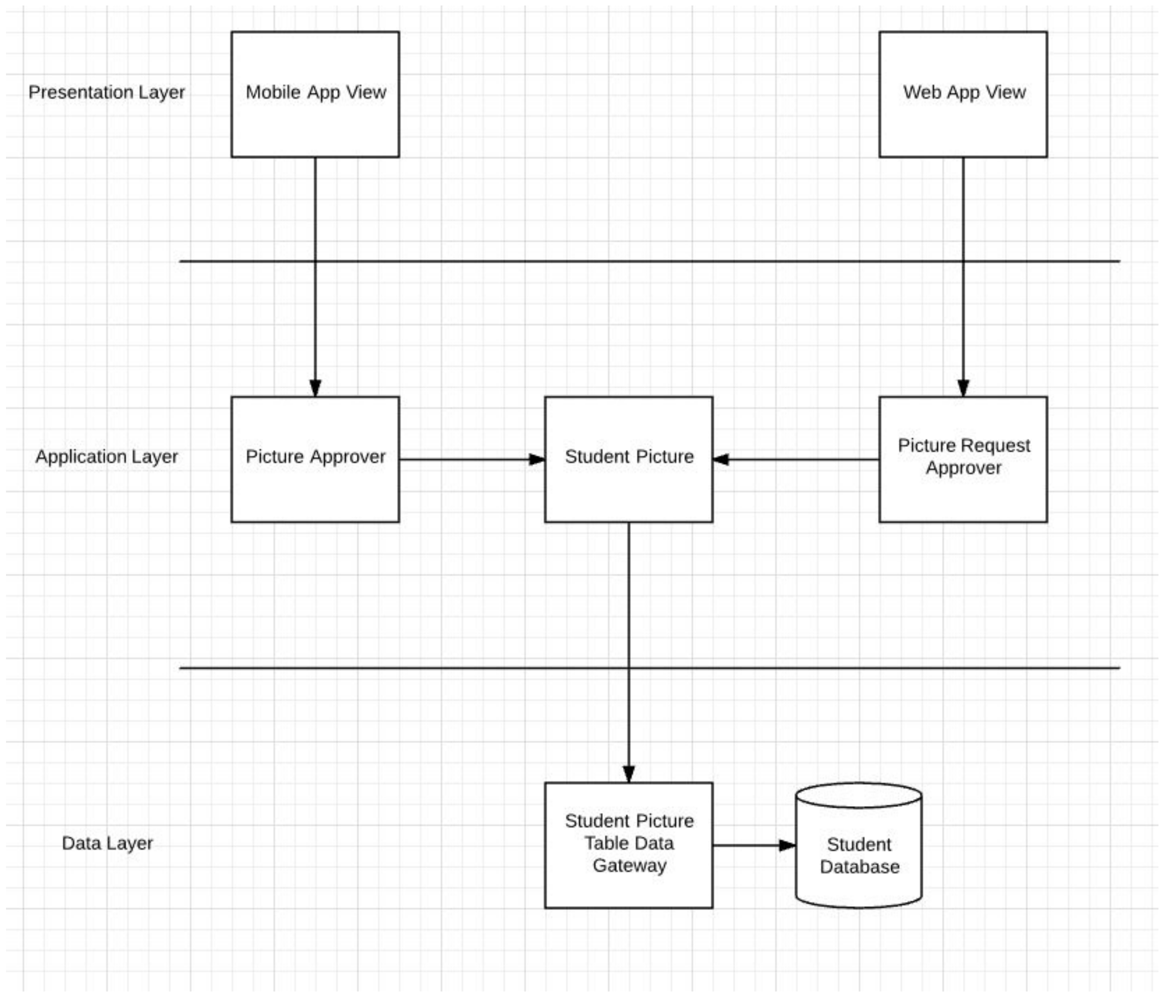
In term of documentation, our stakeholders were very intent on us documenting our efforts, so that they could easily take over the project after the final release. Multiple documents were then made to satisfy this demand. The first document was an installation manual detailing the steps necessary in order to run and execute the code. This manual contained very detailed steps on what to install and how to do so. This was done for all the backend, web application, and both versions of the mobile application.

Another release document that was given to our stakeholder was a detailed user manual describing the features available on the mobile and web applications. The steps on how to use these features were accompanied by pictures which give an overview of our application. This document is used to showcase all the work done during the project as well as all the features that were implemented since it's inception.

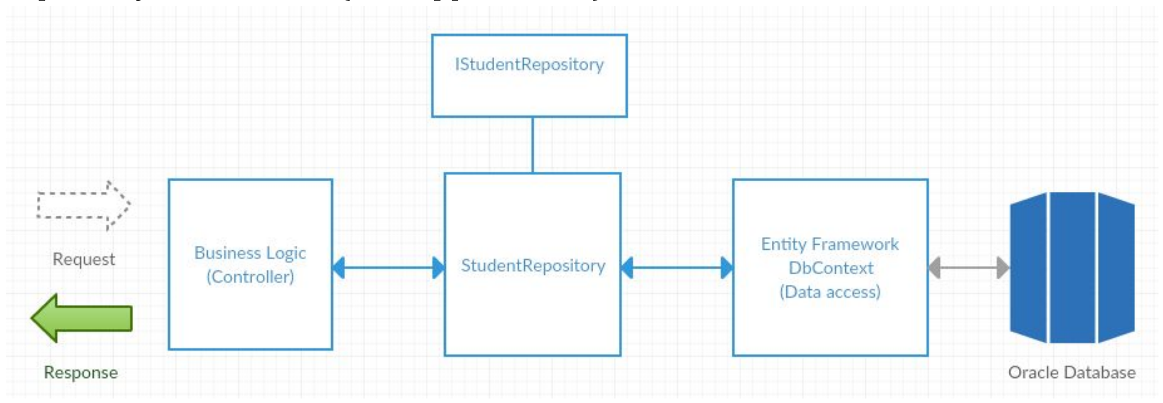
The last document requested by our stakeholders was a technical document containing all the diagrams pertaining to the project. The document contains control flow diagrams for the application, as well as architectural diagrams, such as a layer-diagram and a database schema. This document will prove useful for helping the team taking over the project better understand the structure of the project.

Aside from documentation, the main feature that was implemented for this release was the localization feature, which we had been told at the beginning of the final release at to be part of our project. This feature entailed that our application was useable in both french and english. This feature was implemented in our final sprint and is correctly working on both versions of the mobile application as well as the web application.

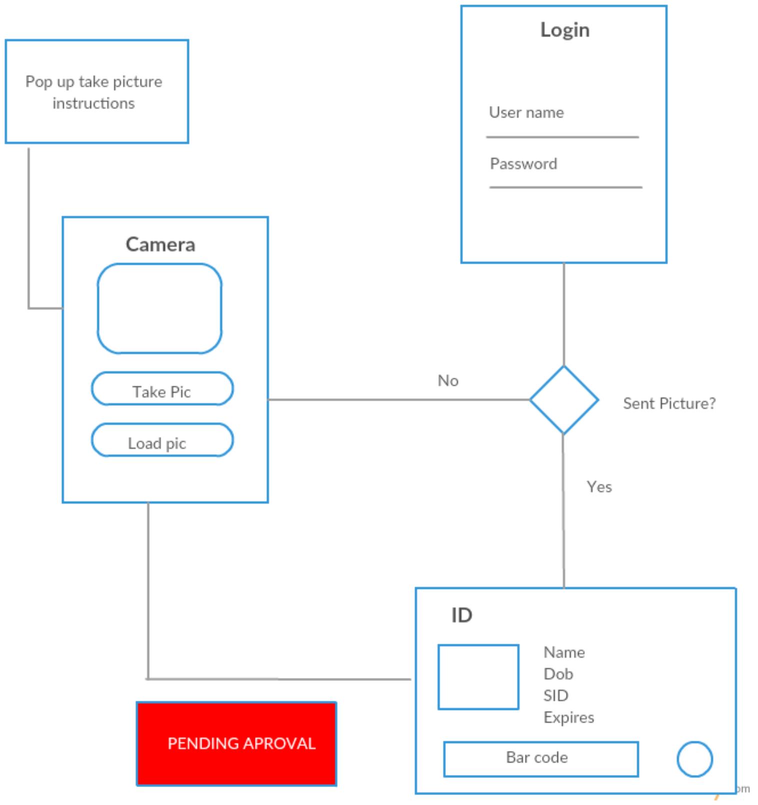
Overall Arch and Class diagram



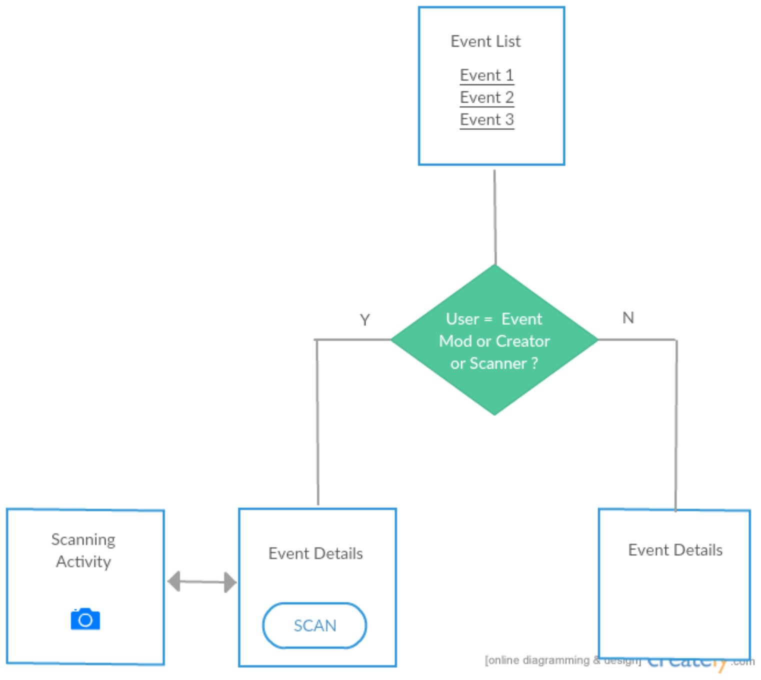
Repository Architecture (WebApp Backend)


IStudentRepository
Request
Business Logic (Controller)
StudentRepository
Entity Framework DbContext (Data access)
Response
Oracle Database

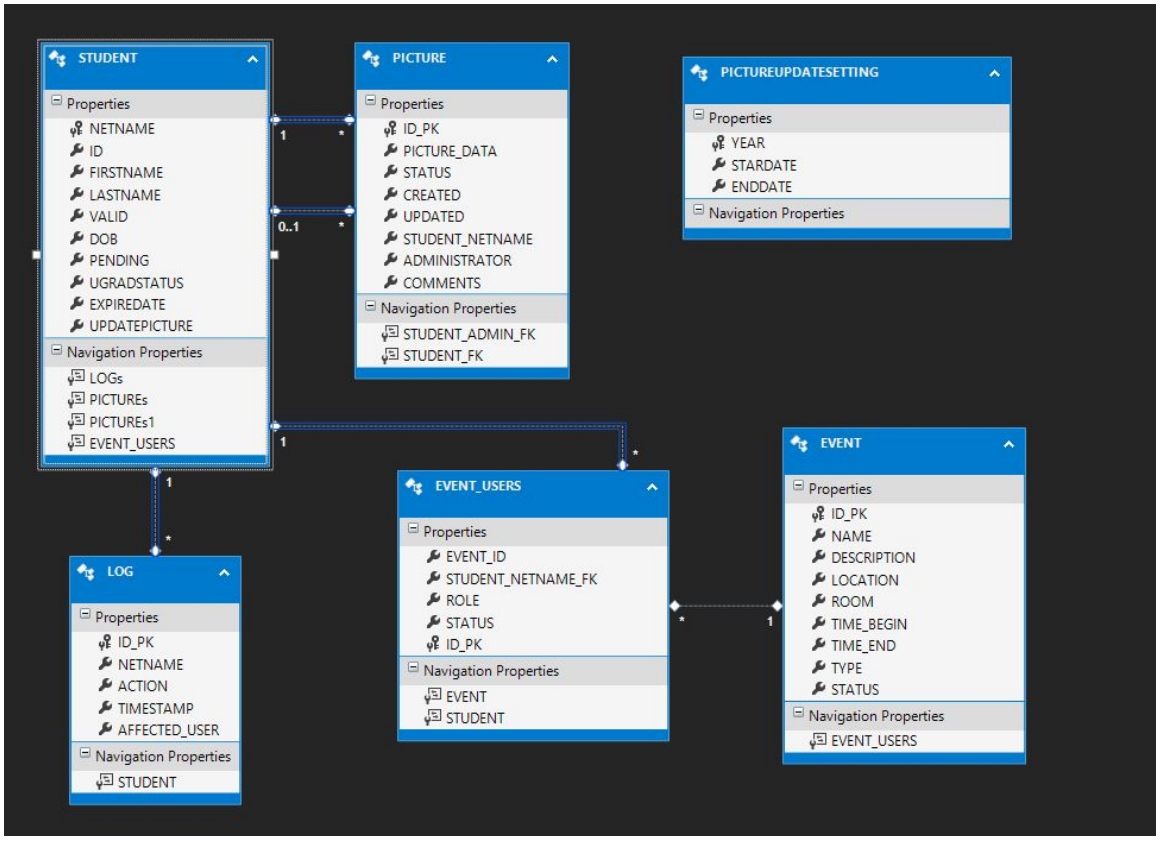
Control Flow Camera - Mobile App



Control Flow Events - Mobile App


Event List
Event 1
Event 2
Event 3
User = Event Mod or Creator or Scanner ?
Y
N
Scanning Activity
Event Details
SCAN
Event Details
[online diagramming & design] creately.com

Database Diagram



The database contain 6 tables :

- Student
- Picture
- PictureUpdatePeriodSetting
- Logs
- Event_users
- Event

The student tables are containing each pertinent information about a student. Each picture send by a student will be stored in the Picture table. The PictureUpdatePeriodSetting is used to store the valid period of time where everyone valid student will be able to update their profile picture. Log table is used to logs users actions. We can therefore know which user did which action and which user was affected by it.

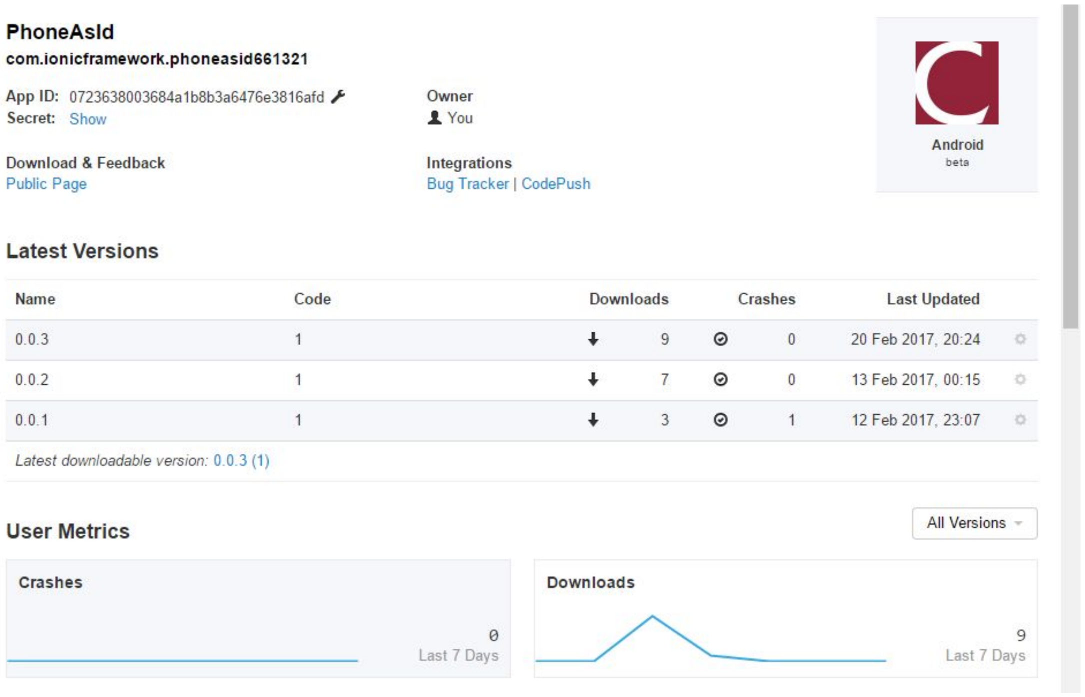
Event table is holding information about an event. Event_user is containing a list of every users that are participating to an event.

Infrastructure

- **Phonegap/ionic**
- **ASP.NET**
- **Entity Framework**
- **Active Directory**
- **Oracle Database (Oracle 12C)**
- **Amazon Relational database service**
- **Azure**
- **Application Insights**
- **Google Vision**
- **netlify**
- **Jenkins**
- **swagger**
- Jenkins url
- netlify url

Infrastructure Changes

- **https://hockeyapp.net/**
- **android.myconcordiaid.me**



PhoneAsId
com.ionicframework.phoneasid661321

App ID: 0723638003684a1b8b3a6476e3816afd
Secret: Show

Owner
You

Download & Feedback
Public Page

Integrations
Bug Tracker | CodePush

Android
beta

Latest Versions

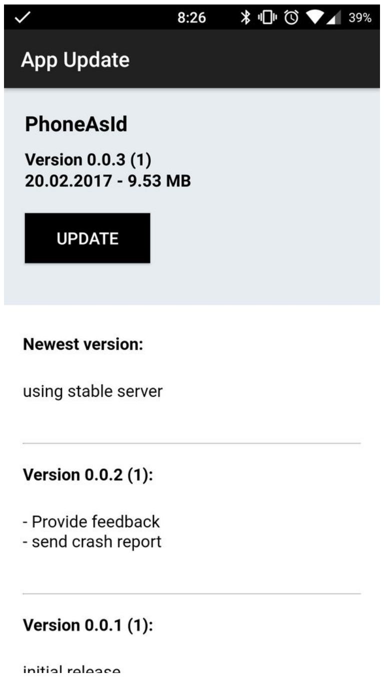
Name	Code	Downloads	Crashes	Last Updated
0.0.3	1	9	0	20 Feb 2017, 20:24
0.0.2	1	7	0	13 Feb 2017, 00:15
0.0.1	1	3	1	12 Feb 2017, 23:07

Latest downloadable version: 0.0.3 (1)

User Metrics

We are using hockeyapp for beta build distribution, crash reporting and getting user feedback. We automatically track specific event happening in our app. We will get a log of each successful login, attending to view the event page or id card.

User are forced to always be on the latest version.



PhoneAsId

Version 0.0.3 (1)
20.02.2017 - 9.53 MB

Newest version:

using stable server

Version 0.0.2 (1):

- Provide feedback
- send crash report

Version 0.0.1 (1):

initial release

Building Ios version

It’s important to mention is that there were difficulties with regards to running the application on an iOS device. First off, building an iOS version of the application requires a MacBook and all the software related to it. Nobody on our team has a MacBook, and therefore we needed to acquire one in order to continue working on the issue. Because of this, there was a lot of back and forth between our stakeholders in order for them to finally lend one to us. Another issue was that they forgot to give us admin rights on the MacBook, and this delayed the process for building the application on the MacBook. Furthermore, a big challenge was getting accustomed to using a MacBook, and all the software needed to build and run our application on an iOS device. There were a lot of workarounds that needed to be done before a build would successfully compile. This includes needing an apple developer account when building the app, needing to gain familiarity with the software “XCode” in order to build and run the application on an iOS device, needing to use a 4k image in order to generate the iOS splash-screens, etc… That being said, the application has successfully compiled on an iOS device, but some of the features aren’t functioning as planned, and some of the UI isn’t displaying as intended. Those issues will be further looked into in the upcoming release.

Internationalization

Mobile localization was achieved using the “angular-translate” library as well as the “cordova-plugin-globalization”. The cordova plugin is used on app launch to detect the system language of the phone. Once this is done the angular translate library is used for providing the appropriate translation. Using the library involves modifying the hardcoded strings in the html to string references which are then filtered using the translate service. Two JSON files are used which contain references keys and translated values for english and french. For the web application, the same angular-translate library was used, but we also used angular-translate-storage-cookie in order to store the selected language in the browser as a cookie.

Name Conventions

- John Papa’s Angular style guide
- C# Coding style for asp.net core

Code

File path with clickable GitHub link	Purpose (1 line description)
translate.service.js	Translation service to translate a given expression

Event Repository	Store and Retrieve Event Information from the database
Event Controller (Mobile Application)	The event controller will get and display the list of all events that are available for the user. It will call a service that will return the list of events. That list will be displayed in the event.html page using ion-list.
EventController(Web application)	The event controller will get and display the list of all events that are available for the user. It will call a service that will return the list of events. That list will be displayed in the event.html page. In addition, it will handle the logic for displaying a list of a specific event's attendees.
studentModal.Controller.js	This controller is the biggest controller in the webapp and the most important. It handles most of the write interactions with the database as well as most of the reads. In addition to its designated job it is also an intermediary between some other components such as the imageModalController and the searchController. It follows the John papa's Angular style guide. Most of the logic is abstracted from the controller with the use of factories and services (Eg. StudentModalController uses StudentService)..
student.service.js (web app)	This service handles some the logic behind all the controllers. Every bit of logic that is related to student information passes through this service. It sorts of connect the backend and the frond end by handling all get and post requests. The services works as an abstraction layer between the backend and the frontend. All the services follows John papa's Angular style guide.
toastedHttp.service.js	This service wraps the angular http requests and displays configured feedback based on provided settings.

Testing and Continuous Integration

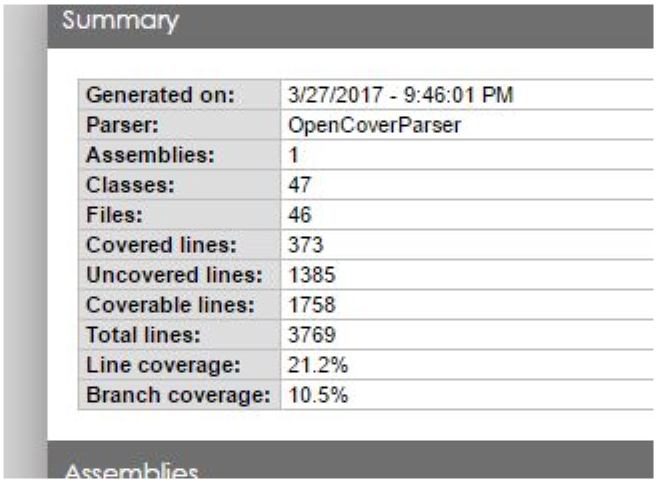
Test File path with clickable GitHub link	What is it testing (1 line description)
GetEventById() Event Controller Unit Test Line 127	If we can retrieve a event information by using a Id.
GetEventByIdNotFound() Event Controller Unit Test Line 168	Test error handling if we can't find a event.
CancelEventNotFound() Event Api Unit Test Line 256	We send event id that doesn't exist, we are supposed to receive a 403 error since we can't cancel it.
UpdateEventNotFound() Event Api Unit Test Line 322	When we update an event we are required to send a correct event id which exist in the db or else we won't be able to find and will return a 403.
Student search (E2E) Web app E2E testing	In the Web app, it tests the search functionality by looking for a specific student and opening its corresponding student modal.
New event (E2E) Web app E2E testing	In the Web app, it tests the event creation functionality
Delete event(E2E) Web app E2E testing	In the Web app, it tests the delete event functionality

All unit tests made for the api server are runned each time by jenkins during the build process.

Code Coverage

Coverage is low because we can't unit test most of our queries because they are modifying the database. Our queries are mainly post, put and delete.

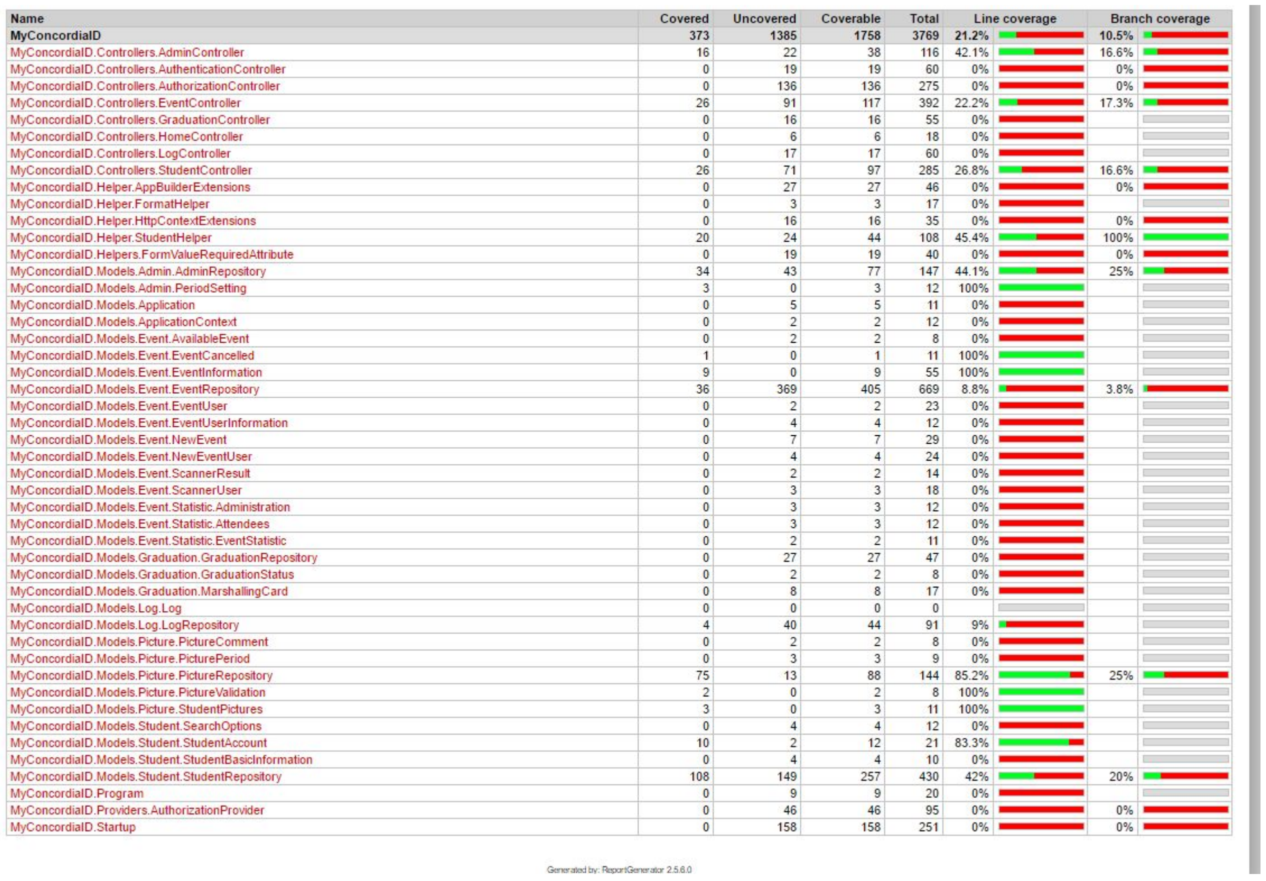
Direct Link to coverage report



Summary

Generated on:	3/27/2017 - 9:46:01 PM
Parser:	OpenCoverParser
Assemblies:	1
Classes:	47
Files:	46
Covered lines:	373
Uncovered lines:	1385
Coverable lines:	1758
Total lines:	3769
Line coverage:	21.2%
Branch coverage:	10.5%

Assemblies



Name	Covered	Uncovered	Coverable	Total	Line coverage	Branch coverage
MyConcordialD	**373**	**1385**	**1758**	**3769**	**21.2%**	**10.5%**
MyConcordialD.Controllers.AdminController	16	22	38	116	42.1%	16.6%
MyConcordialD.Controllers.AuthenticationController	0	19	19	60	0%	0%
MyConcordialD.Controllers.AuthorizationController	0	136	136	275	0%	0%
MyConcordialD.Controllers.EventController	26	91	117	392	22.2%	17.3%
MyConcordialD.Controllers.GraduationController	0	16	16	55	0%	
MyConcordialD.Controllers.HomeController	0	6	6	18	0%	
MyConcordialD.Controllers.LogController	0	17	17	60	0%	
MyConcordialD.Controllers.StudentController	26	71	97	285	26.8%	16.6%
MyConcordialD.Helper.AppBuilderExtensions	0	27	27	46	0%	0%
MyConcordialD.Helper.FormatHelper	0	3	3	17	0%	
MyConcordialD.Helper.HttpContextExtensions	0	16	16	35	0%	0%
MyConcordialD.Helper.StudentHelper	20	24	44	108	45.4%	100%
MyConcordialD.Helpers.FormValueRequiredAttribute	0	19	19	40	0%	0%
MyConcordialD.Models.Admin.AdminRepository	34	43	77	147	44.1%	25%
MyConcordialD.Models.Admin.PeriodSetting	3	0	3	12	100%	
MyConcordialD.Models.Application	0	5	5	11	0%	
MyConcordialD.Models.ApplicationContext	0	2	2	12	0%	
MyConcordialD.Models.Event.AvailableEvent	0	2	2	8	0%	
MyConcordialD.Models.Event.EventCancelled	1	0	1	11	100%	
MyConcordialD.Models.Event.EventInformation	9	0	9	55	100%	
MyConcordialD.Models.Event.EventRepository	36	369	405	669	8.8%	3.8%
MyConcordialD.Models.Event.EventUser	0	2	2	23	0%	
MyConcordialD.Models.Event.EventUserInformation	0	4	4	12	0%	
MyConcordialD.Models.Event.NewEvent	0	7	7	29	0%	
MyConcordialD.Models.Event.NewEventUser	0	4	4	24	0%	
MyConcordialD.Models.Event.ScannerResult	0	2	2	14	0%	
MyConcordialD.Models.Event.ScannerUser	0	3	3	18	0%	
MyConcordialD.Models.Event.Statistic.Administration	0	3	3	12	0%	
MyConcordialD.Models.Event.Statistic.Attendees	0	3	3	12	0%	
MyConcordialD.Models.Event.Statistic.EventStatistic	0	2	2	11	0%	
MyConcordialD.Models.Graduation.GraduationRepository	0	27	27	47	0%	
MyConcordialD.Models.Graduation.GraduationStatus	0	2	2	8	0%	
MyConcordialD.Models.Graduation.MarshallingCard	0	8	8	17	0%	
MyConcordialD.Models.Log.Log	0	0	0	0		
MyConcordialD.Models.Log.LogRepository	4	40	44	91	9%	
MyConcordialD.Models.Picture.PictureComment	0	2	2	8	0%	
MyConcordialD.Models.Picture.PicturePeriod	0	3	3	9	0%	
MyConcordialD.Models.Picture.PictureRepository	75	13	88	144	85.2%	25%
MyConcordialD.Models.Picture.PictureValidation	2	0	2	8	100%	
MyConcordialD.Models.Picture.StudentPictures	3	0	3	11	100%	
MyConcordialD.Models.Student.SearchOptions	0	4	4	12	0%	
MyConcordialD.Models.Student.StudentAccount	10	2	12	21	83.3%	
MyConcordialD.Models.Student.StudentBasicInformation	0	4	4	10	0%	
MyConcordialD.Models.Student.StudentRepository	108	149	257	430	42%	20%
MyConcordialD.Program	0	9	9	20	0%	
MyConcordialD.Providers.AuthorizationProvider	0	46	46	95	0%	0%
MyConcordialD.Startup	0	158	158	251	0%	0%

Generated by: ReportGenerator 2.5.6.0

Lot of Models class can't be covered because they are only used during a insert or update query. In my unit tests, we are mocking a entity framework database context. We could test that an insert was made, but we can't verify if it has inserted the correct item. We can't do a get request to find the inserted item and compare it to the previous input, therefore we have only tested the methods in which it was possible.

Code coverage are done each time a new build is triggered by jenkins.

Finished SHORT Story summaries

Order your summaries by risk and priority. They should have the following form.

Points: 20, Priority: 1, Risk: Medium
Story #: 78 Production Environment
Iteration # 10
We were required to create production environment on Concordia's network. We got major complication with firewalls which pushed back the issue. The production API server is running 24/7. All the API are protected by there all request must have an authorized token to receive a successful response.

Points: 8, Priority: 1, Risk: Medium
Story #: 137 Improve performance
Iteration # 11
Increase scalability of the API server. Transform multiple api to use asynchronous operations.

Points: 8, Priority: 1, Risk: Low
Story #: 134 Application Guide
Iteration # 10
This user story was to write a guide on how to use the mobile and web application, complete with picture so that any user could pick up the application and go through the features that are available. This was requested by the stakeholder as part of the knowledge transfer part.

Points: 8, Priority: 1, Risk: Low
Story #: 135 Software Specification Design document
Iteration # 10
This user story was to write a software specification document to be hand out to the developer at IITS that would take over the project. This would give them a good idea about the structure of the project and facilitate the transition. These document contained diagrams of the structure of the project, as well as installation document to set up the work environment. This was requested by the stakeholder as part of the knowledge transfer part.

Points: 8, Priority: 1, Risk: Low
Story #: 159 Mobile Application Localization - Language
Iteration # 11
This user story consists of allowing the mobile application to switch between the language set on the user's phone using localization. This feature allows the

application's text to be displayed in either english or french. This functionality is implemented for both Android and iOS.

Points: 8, Priority: 1, Risk: Low
Story #: 158 Web Localization - Language
Iteration # 11
This user story involved making use of the angular-translate library in order to make available the entire web application in both english, and french. Work included adding translate directives to all html tags where translation occurred, handling translations in the controllers and services, writing and structuring JSON files containing the translation mappings, and handling storage of selected language in a cookie.

The feature was given to us as a requirement very late in the project, and the application was not yet structured to handle it. A lot of documentation and good practices had to be learned in order to understand how to properly perform localization. Translations in javascript controllers/services were much easier than in html pages, so a lot of time was spent on the controllers/services. Date translations were very tricky for dates that contained the months in name such as "March 3, 2017" in english versus "3 mars 2017" in french. For these reasons, we believe the story should be re-estimated to 20 user story points with Medium risk.

Points: 5, Priority: 3, Risk: Medium
Story #: 138 Logout confirmation
Iteration # 10
This user story was to add a confirmation check before signing out the user from the mobile application to prevent frustration from being logged out in the case of clicking the button by mistake. This story consisted of adding an alert displayed to the user asking for confirmation before logging out.

Points: 3, Priority: 3, Risk: Low
Story #: 132 Event Edit User permission
Iteration # 10
User must have required authorization before making anything modification to another user account.

Points: 2, Priority: 3, Risk: Low
Story #: 133 Event Cancel permission
Iteration # 10
User must have the required authorization to cancel an event. Only the creator of the event can cancel it.

Points: 13, Priority: 3, Risk: Low
Story #: 127 Event Statistic

Iteration # 10
Event administrator can view the outcome of an event. Event attendance statistic will be provided. Diagram will display percent of attendance. Concordia administration will be able view these information and determine if associations are creating event that have high attendance.

Points: 3, Priority: 3, Risk: Low
Story #: 154 Custom feedback messages
Iteration # 10
This user story involved configuring the mdToasts to show feedback relevant to whatever the action that the user performed was. It involved making javascript constant files that contained messages corresponding to success and error scenarios, including the specific types of errors from the http callback (e.g. 401, 403, 500, etc). The constant files were later refactored into json files to accommodate localization. The estimate of 3 story points was a bit conservative. We would re-estimate this task to 5 because the story involved adding a large number of lines of code, as well as some complexity in how the problem should be solved.

Points: 2, Priority: 3, Risk: Low
Story #: 141 Name of event in attendee list
Iteration # 10
This story involved a simple line of code change in order to add the name of the event in the attendees page to indicate to the user the attendees of which event he/she was looking at. We believe we were generous with this estimate because we already knew the solution was easy, but added some complexity in case it wasn't. We would change the estimate from 2 to ½.

Points: 3, Priority: 3, Risk: Low
Story #: 168 Description of Open and Closed
Iteration # 11
Provide extra information while the cursor is hovering these specific tags.

Points: 13, Priority: 5, Risk: Low
Story #: 57 Behavior testing
Iteration # 11
Automate behavior testing with protractor. We pushed back this issue multiple times since our user interface was changing every week.