# PhoneAsId Release 1 Summary

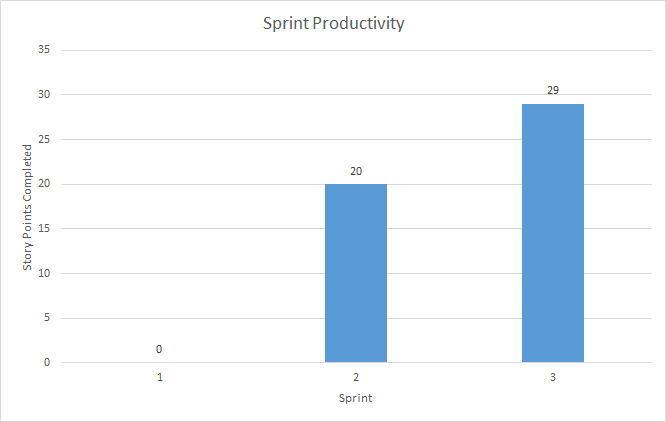
## Team members

|  |  |  |
| --- | --- | --- |
| Name and Student id | GitHub id | Number of story points that member was an **author** on. |
| Michał Wozniak  21941097 | mv740 | 5(#5) + 8(#9)+8(#7)=21 |
| Francis Côté-Tremblay 26615287 | francisct | 8(#9) |
| Ahmed Dorias  26649874 | ConfusedGiant | 2(#6) + 8(#7) = 10 |
| Harrison Ianatchkov 26607403 | zzharryzz | 8(#9) |
| Simon Monière Abes 26648568 | simonma1 | 8(#3) + 5(#4)+2(#6) + 13(#8) = 28 |
| Sebastian Rafique Proctor-Shah  29649727 | EXPSPACE | 5(#5) |

## 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. Some future features may include geo-detection with the use of iBeacons, purchasing printer credits, signing up and wirelessly authenticating membership to Concordia’s LeGym, as well as displaying shuttle bus schedules.

## Velocity



Our **velocity** is 49 points over 3 iterations = **16 user story points / iteration**

Total: 8 stories, 49 points over 6 weeks\*

[Iteration 1](https://github.com/mv740/E-Wok-MyConcordia/milestone/1) (0 stories, 0 points)

[Iteration 2](https://github.com/mv740/E-Wok-MyConcordia/milestone/2) (4 stories, 20 points)

[Iteration 3](https://github.com/mv740/E-Wok-MyConcordia/milestone/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.

## Plan up to next release

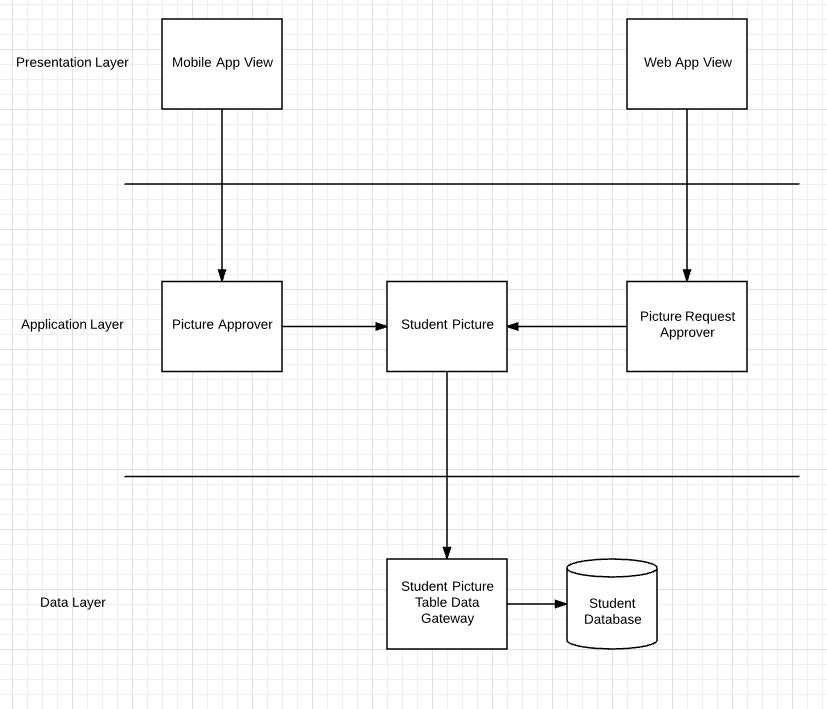
Total: 11 stories, 103 points, over 7 weeks

[Iteration 4](https://github.com/mv740/E-Wok-MyConcordia/milestone/4) (4 stories, 38 points)

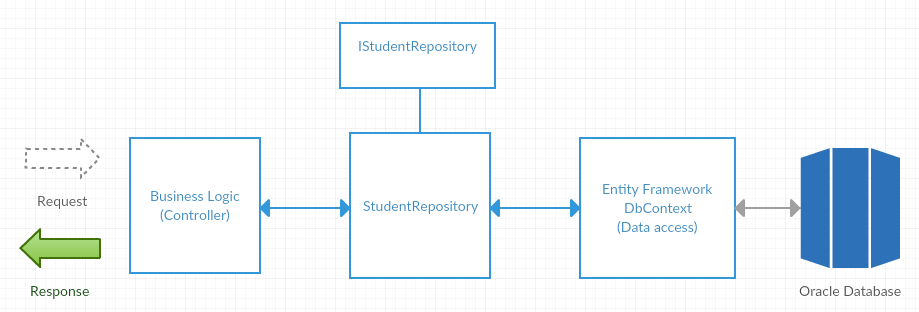
[Iteration 5](https://github.com/mv740/E-Wok-MyConcordia/milestone/5) ( 5 stories, 47 points)

[Iteration 6, Release 2](https://github.com/mv740/E-Wok-MyConcordia/milestone/6) (2 stories, 18 points)

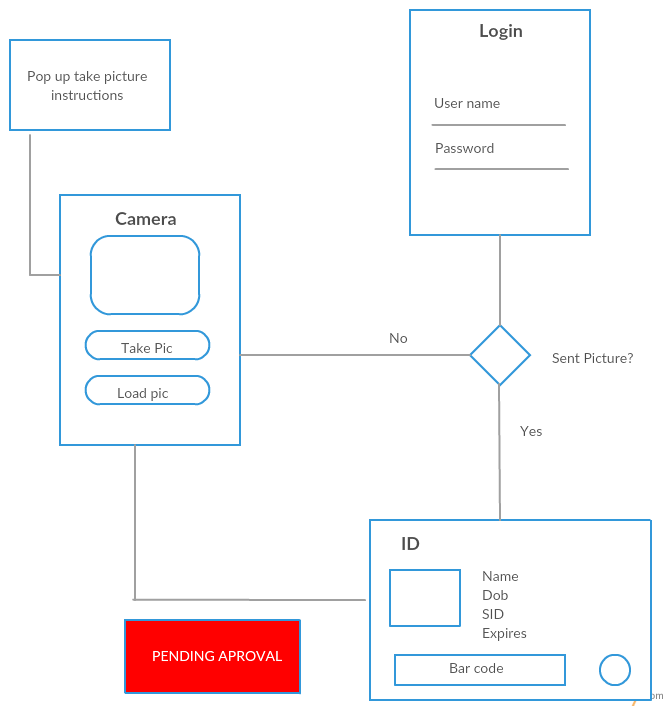
## Overall Arch and Class diagram



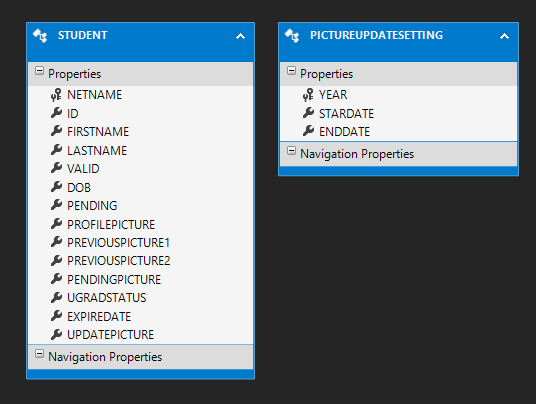
Repository Architecture (WebApp Backend)



Control Flow Mobile App



Database Class Diagram



## Infrastructure

[**Phonegap/ionic**](http://ionicframework.com/)

Using Phonegap was a requirement imposed by our stakeholder, IITS. Their team has experience developing mobile applications using Phonegap, so it will be easier to for them to do maintenance when they eventually take over the project. Phonegap allows for building cross-platform mobile applications that run on iOS, android and windows phones. Ionic is an HTML5 mobile app framework which will improve front end UI development.

[**ASP.NET**](https://www.asp.net/get-started)

ASP.NET was a requirement imposed by our stakeholder. The majority of the concordia’s online system is build using this framework, therefore they would able to continue maintaining the application.

We decided to use the ASP.NET Core version which used to be called ASP.NET 5. Microsoft made this version open-source and added architectural changes which resulted into a cross-platform framework for building modern cloud based and on-premise application. It consists of modular components with minimal overhead, which permit you to keep your flexibility while building your solution.

[**Entity Framework**](https://www.asp.net/entity-framework)

It is an object-relational mapper that enables .Net developers to work with relational data using domain-specific objects. We aren’t required anymore to use raw data-access code.

[**Active Directory**](https://azure.microsoft.com/en-us/documentation/articles/active-directory-aadconnect/)

Active Directory is the directory that will be used to be capable of implementing Oauth 2.0 authorization protocol. This will permit users to login using their concordia netname.

[**Oracle Database (Oracle 12C)**](http://www.oracle.com/us/corporate/features/database-12c/index.html)

For the backend of this application, we will use Oracle Database, more specifically Oracle 12C. Oracle RDBMS is an object-relational database management system. This system is used by IITS in order to store all the information related to the students at Concordia. Therefore, we decided to use it so that the integration with their system will be much easier to do once we hand over the project to IITS developers at the end of our capstone.

[**Amazon Relational database service**](https://aws.amazon.com/rds/)

We are using this service to host our oracle database online. It was the only available service that was offering free hosting for Oracle 12c.

[**Azure**](https://azure.microsoft.com/)

It is a cloud computing platform. We will used to deploy our web application. We can take advantage of Azure’s continuous deployment with github. Whenever we will commit to our Github repo, that code will be automatically synced up with our Azure Web app.

[**Application Insights**](https://azure.microsoft.com/en-us/services/application-insights/)

It is an extensible analytic service that helps you understand the performance and usage of our live web application. It’s designed to help us continuously improve the performance and usability of our application. Its help detect if some queries are failings.

[**Google Vision**](https://cloud.google.com/vision/)

Google Vision is an API made by Google that allows the user to send picture to their server and which will return an analysis of the image according to the criteria used. Such features include facial recognition and a module to detect inappropriate content. This could be useful in the application to prevent user from sending picture that are not following the preset criteria of acceptance.

Microsoft faces API was also considered for this purpose, but ultimately Google Vision was chosen because of the more robust list of features that are available through the Google API. Microsoft’s API was more powerful when it came to the facial recognition aspect, but in the end Google Vision had a more features that could be of use to us.

## Name Conventions

* [Javascript ES5](https://github.com/airbnb/javascript/tree/es5-deprecated/es5)
* [C# Coding style for asp.net core](https://github.com/dotnet/corefx/blob/master/Documentation/coding-guidelines/coding-style.md)

## Code

|  |  |
| --- | --- |
| File path with clickable GitHub link | Purpose (1 line description) |
| [StudentController](https://github.com/mv740/E-Wok-MyConcordia/blob/master/MyConcordiaID/src/MyConcordiaID/Controllers/StudentController.cs) | Student Related Apis |
| [Student Repository](https://github.com/mv740/E-Wok-MyConcordia/blob/master/MyConcordiaID/src/MyConcordiaID/Models/Student/StudentRepository.cs) | Store and Retrieve Student Information from the database |
| [Camera Controller](https://github.com/mv740/E-Wok-MyConcordia/blob/master/PhoneAsId/www/js/cameraController.js) (mobile application) | This controller is in charge of handling the user request to use the camera, such as taking picture, loading a picture, as well as sending the picture to the server. |
| [App.js](https://github.com/mv740/E-Wok-MyConcordia/blob/master/PhoneAsId/www/js/app.js) (Mobile Application) | This class starts the application. It’s also responsible for the sidebar, as well as navigating between the different pages. |
| [\_LoginPartial.cshtml](https://github.com/mv740/E-Wok-MyConcordia/blob/master/MyConcordiaID/src/MyConcordiaID/Views/Shared/_LoginPartial.cshtml) | This file includes backend checks to ensure that features are only displayed if the user is properly logged in. In addition, it includes the html structure for the sidenav bar. Finally, it functions as the link between our backend ASP.NET MVC strcture with our frontend AngularJS |

It should be noted that the file [searchResults.js](https://github.com/mv740/E-Wok-MyConcordia/blob/master/MyConcordiaID/src/MyConcordiaID/wwwroot/partials/review/searchResults/searchResults.js) contains some hard-coded data. Our initial assumption was that we would obtain student data by a queue, but we recently found out it was actually going to be by search. Until we are able to develop the search function (which will be done this sprint), we will use this hardcoded data.

## Testing and Continuous Integration

|  |  |
| --- | --- |
| Test File path with clickable GitHub link | What is it testing (1 line description) |
| FindById()  [Student Api Unit Test](https://github.com/mv740/E-Wok-MyConcordia/blob/master/MyConcordiaID/src/UnitTestCore/StudentUnitTest.cs) Line 64 | If we can retrieve a Student Information by searching using a Id |
| GetAll()  [Student Api Unit Test](https://github.com/mv740/E-Wok-MyConcordia/blob/master/MyConcordiaID/src/UnitTestCore/StudentUnitTest.cs) Line 102 | If we can retrieve all students registered in the database |
| AddPendingPicture()  [Student Api Unit Test](https://github.com/mv740/E-Wok-MyConcordia/blob/master/MyConcordiaID/src/UnitTestCore/StudentUnitTest.cs) Line 141 | If we are correctly storing the picture send by a student to that same student in the database. |
| FindPendingPicture()  [Student Api Unit Test](https://github.com/mv740/E-Wok-MyConcordia/blob/master/MyConcordiaID/src/UnitTestCore/StudentUnitTest.cs) Line 184 | If the stored pending picture is actually the same picture that the user did send previously |
| ValidatePicture()  [Student Api Unit Test](https://github.com/mv740/E-Wok-MyConcordia/blob/master/MyConcordiaID/src/UnitTestCore/StudentUnitTest.cs) Line 236 | if an administrator accept a pending picture, then that user’s pending picture become his profile picture and he become a valid student |

It should be of note that tests regarding the camera functionality of the mobile application are not properly finished for now. A problem occurs when mocking the camera by injecting the cordova camera mock, which prevents testing of this controller. This problem should be addressed and fixed in the coming weeks, but for now it is still an issue.