**INCREMENT REPORT-3**

PROJECT NAME: GO EASY

PAVAN KUMAR BOLLARAM (CLASS ID: 07)

PREETHAM KUMAR DANABOINA (CLASS ID: 11)

LAKSHMI PRIYANKA (CLASS ID: 49)

VARAPRASAD REDDY JAGGU (CLASS ID: 22)

**Introduction**

Go Easy, is a mobile based application which is helpful for novice students who are attending a school or university. Every Student when attending a school at new location explicitly needs someone to guide them regarding school details and information about transportation in and around his school. Using this app a student can find out University Events, Transportation details, Live Events happening around his place. This app uses a friendly user interface which enables simple easy access with existing services. The aim of the project is to guide a student effectively without the need of a third party. The project is implemented on android platform.

**Significance**

Even though there are many apps specific to a particular school what we have noticed is all these apps do not cover all the topics such as transportation details, events happening around the university. So we decided to develop an app which covers all these topics and would be helpful to the new students attending a school. We need to create various web applications capable of managing a user’s interface collection to provide a relevant information for the student attending a school.

**API’s and web services used**

The below services are consumed in our project:

* http://api.eventful.com/rest/events/search?[...](http://api.eventful.com/docs/auth)&keywords=books&location=San+Diego&date=Future

This Eventful API is collection of events going around in particular city. This API acknowledges about live concerts, exhibitions and sport activity held in the city. Eventful.com open unique platform that enables web applications to maximum take advantage of Eventful’s data and features using Eventful API.

Eventful’s data is categorized using geo-graphical area and some other event parameters.

* <https://data.cityofchicago.org/resource/alternative-fuel-locations.json?>

This API is also used to retrieve data from particular location using geo-graphical location basis.

* <https://maps.googleapis.com/maps/api/js?v=3.exp&sensor=true>
  + - * This API is provides help to explore a location and monitor maps for a particular place. Navigation purposes are also met through this API in our application. . This is usually helpful for people programming with java scripts and object orientated programming. The user should be familiar with the how to explore the google maps and how to navigate in point of view.

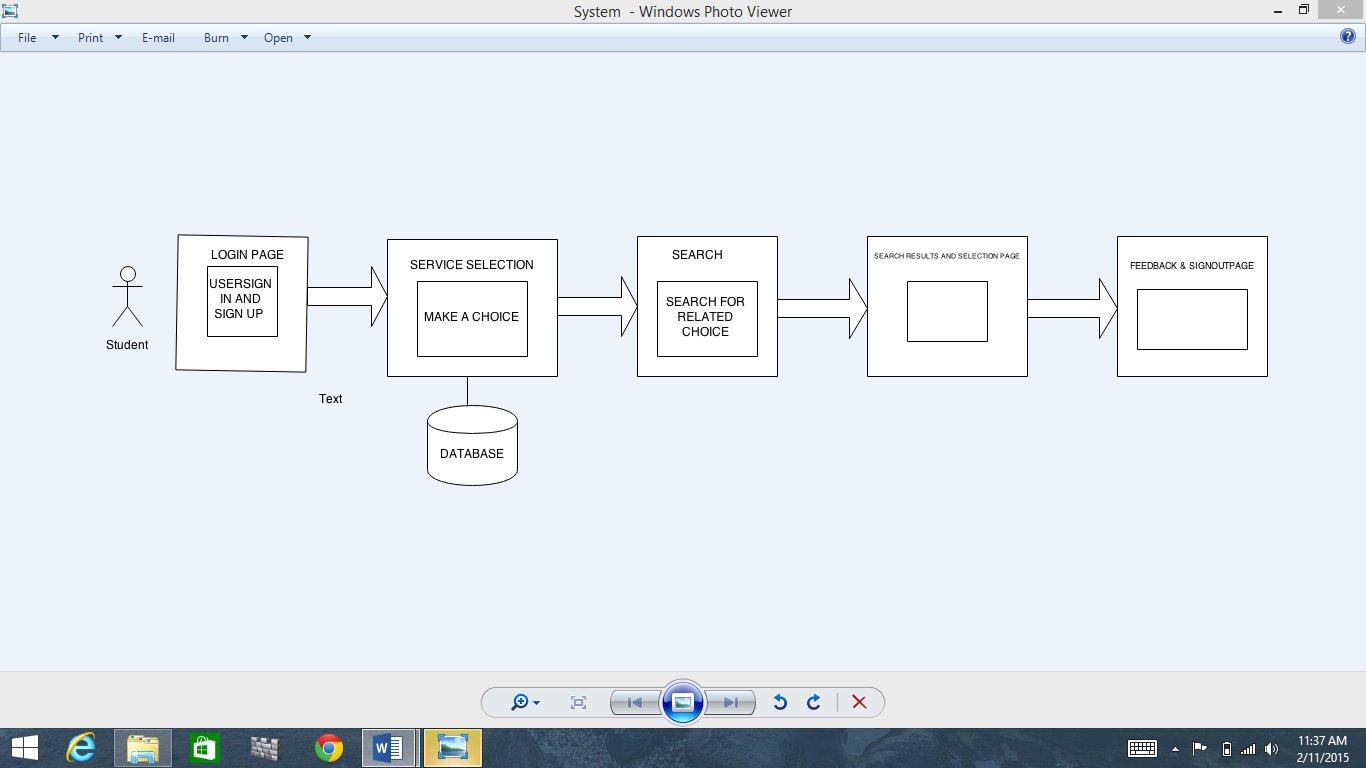
Custom API ‘s used in project

* <http://kc-sce-cs551.kc.umkc.edu/aspnet_client/Group3/Transportation/Service1.svc/transportation/details>
* http://localhost:56120/Service1.svc/GetEventsDetails
* http://localhost:56120/Service1.svc/GetSportsDetails

**DETAIL DESIGN OF SERVICES:**

Go Easy Web services:

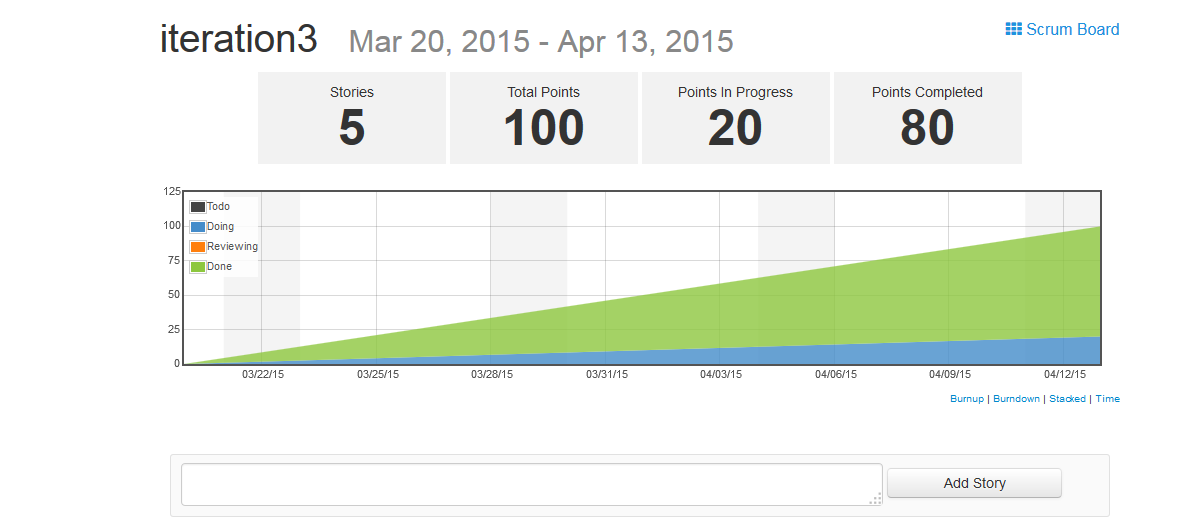
Developing customized web services all at one place which provides feasible access to the events by user. The architecture explains each layer functionality starting from Login page to data layer.

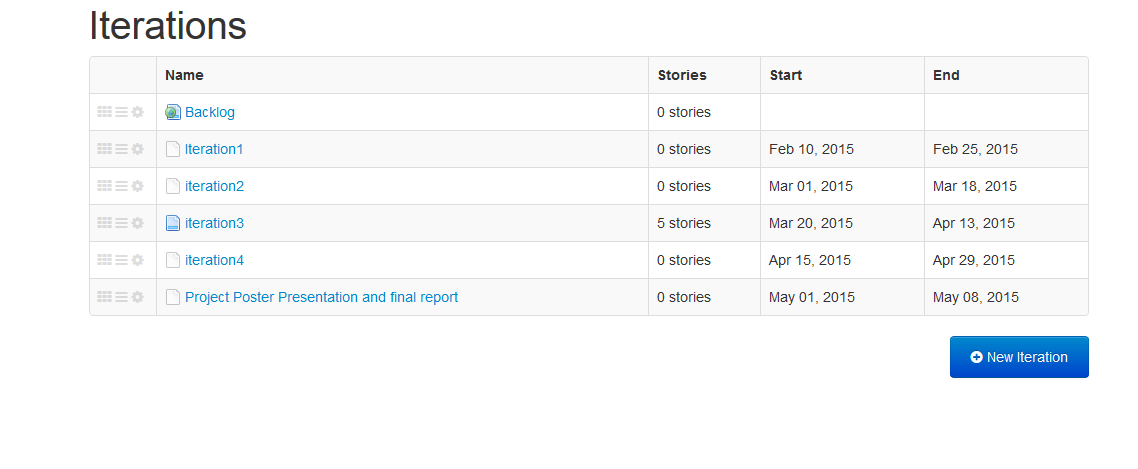


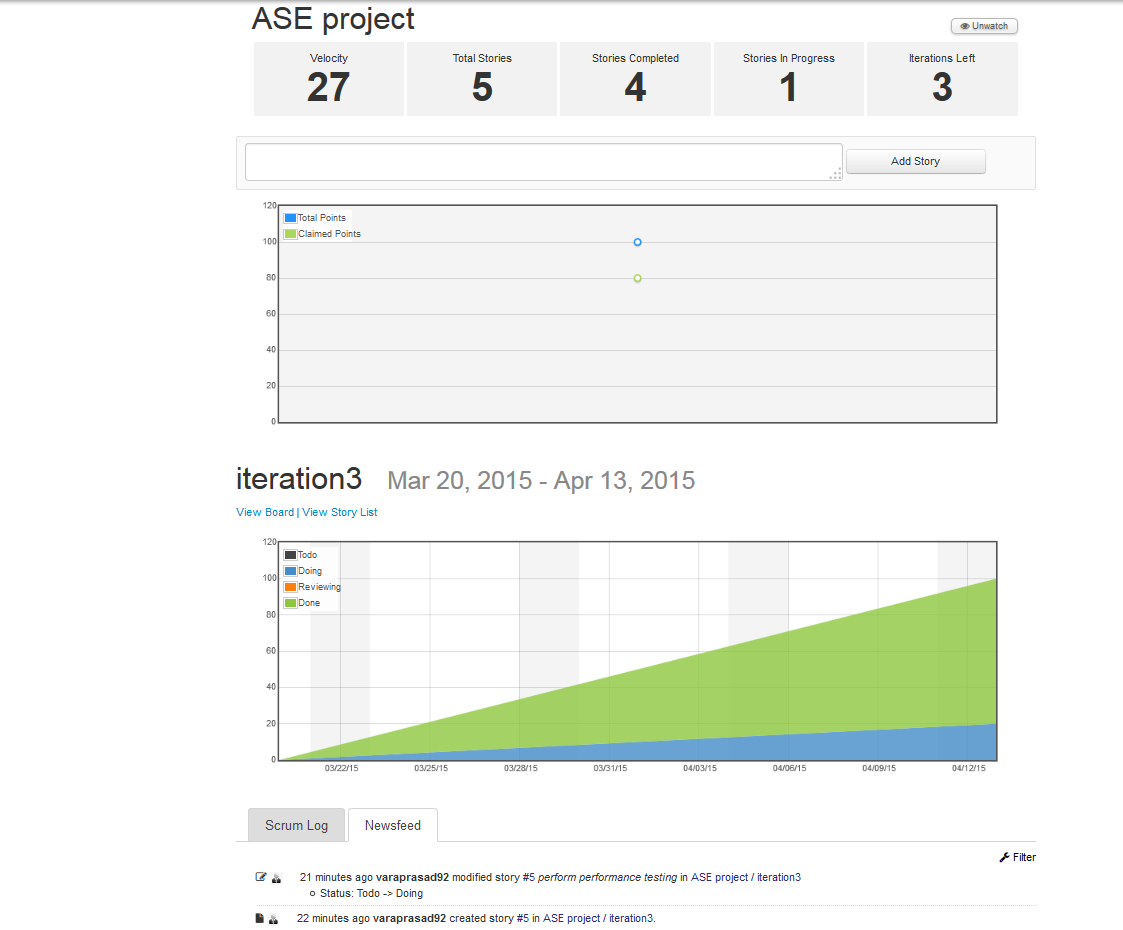
By using the openly available API to connect to end point and get the JSON response all the events; transportation events, university events and Live events can be viewed.

**Scrum Do and Stories**:

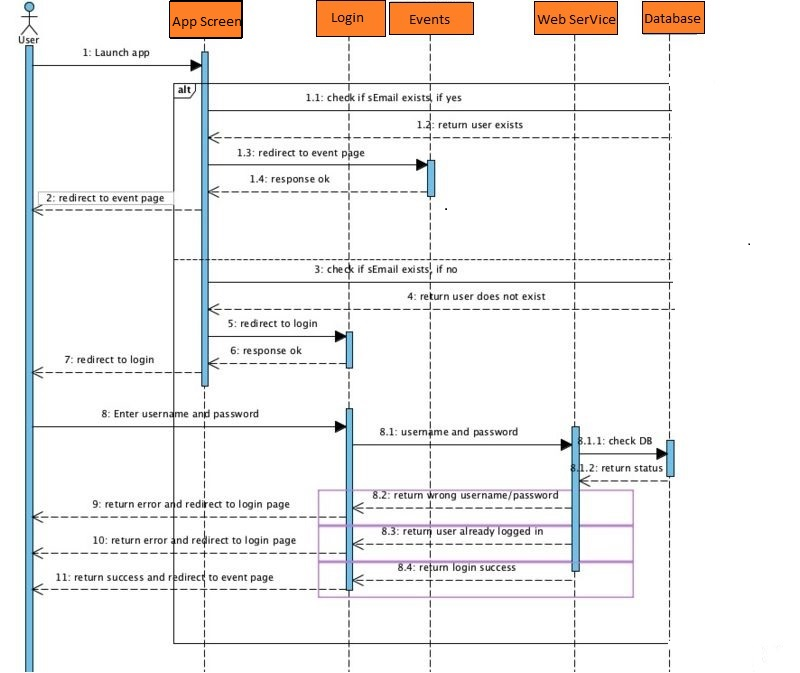
For effective product development, an iterative and incremental methodology is needed. Agile product development methodology is present flexible product development strategy involving many various users as team workers working on unique and checking their progress simultaneously. Self-organizing makes agile process an encouraging toward a good product development irrespective of location and physical presence of the team member. Scrum do uses this agile methodology by adding stories and priority is set in the backlog. Backlog changes the course of the product development and it is not complete initially. In each iteration consists of stories to be done. After the stories are resolved then the iteration is closed and it get directed to next iteration. If any unfinished stories are present then they are directed to next iteration and further resolved. For an iteration the stories which are having priority are added from the backlog. So during this process the total progress is measured and overall project is accessed. The contribution of each team member is also visible which makes a clarity review of a team. The below screen shots depicts the iterations and dashboard of our Scrum Do. The dynamic flow represents the flexibility of the project.







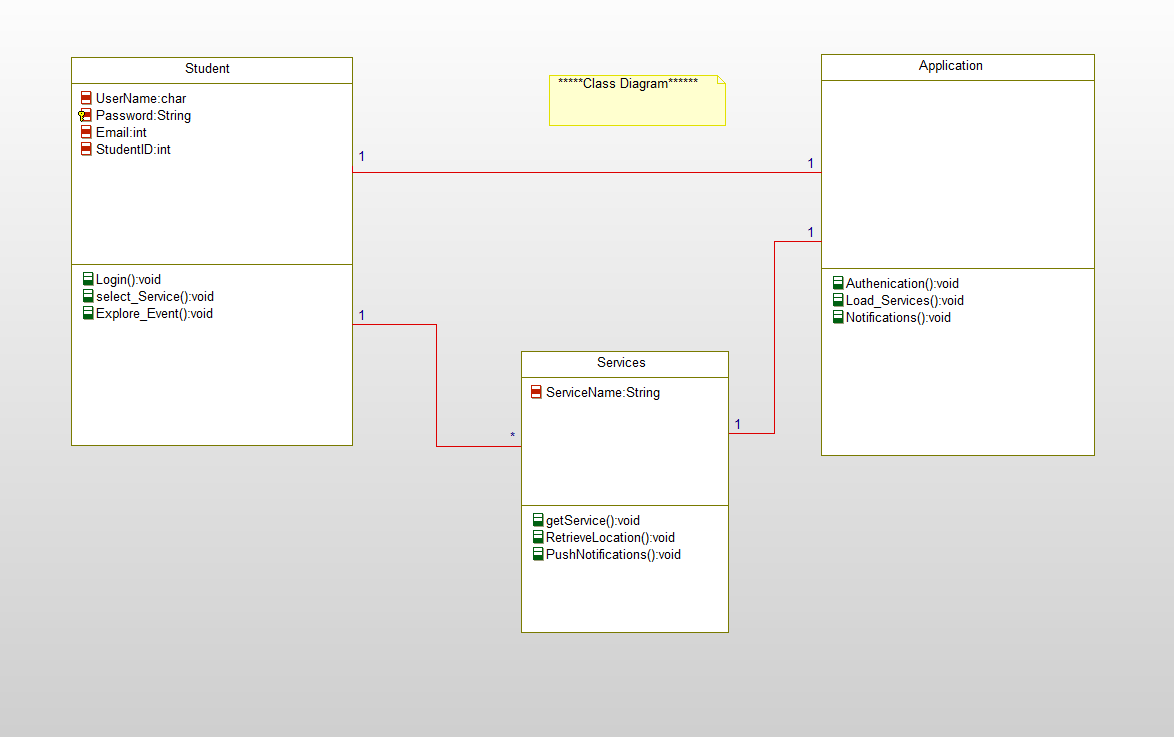
**Sequence Diagrams**:



Message interaction chart diagrams which are sometimes called event diagrams are sequence diagrams explaining how the ordering of processes is done and process they do. In the diagram we consider objects and classes that interact in time sequence. The vertical lines are life lines that objects live on time sequence. Horizontal lines depicts the message interactions between the objects that carried in the process. The above sequence diagram is abstract level representation of services and flow of them. In above diagrams, the message interaction between objects exhibiting the role of student, services and System is shown. First the student registers the app and validation of details mentioned by the user is made by checking out the database present already regarding the user enrolled. At the database a query is run to check whether the student is registered beforehand or need to create a new account. If old user, displaces the services to the user window. If new user, an account created and then displaces the services to the user window. This is abstract level of communication between the objects Student, system and Services.

**Class Diagram**:

Structured representation of classes, attributes and methods (or Operations) which involved in a process are implemented using class diagram.



Here are the classes which we implemented in our project.

Student Class:

This class is primarily associated with students to register with the app by providing his/her login credentials. If the user is already registered then the application prompts the user to login in to the app or else a new login id is created. After logging the user can select the services that are in the app.

Service Class:

This class consists of all the services that we implemented in the application. The primary attribute of this class is a service name and the operations are to push and retrieve the data.

Application Class:

This class authenticates the users with the help of API’s. Along with that this class is responsible for loading the services requested by the user.

**IMPLEMENTATION:**

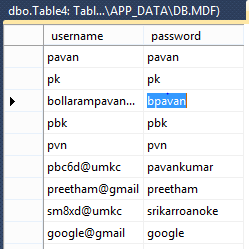
Implementation of REST Services:

Application authorization and registration is done using two API’s

Login API and Registration API

Login API:   
This API does accepts input from the end user and validates them for authentic login to avoid fraud attempts to enter into application

As part of storing the user details we created database table for registration purpose which can be used while user attempts login.



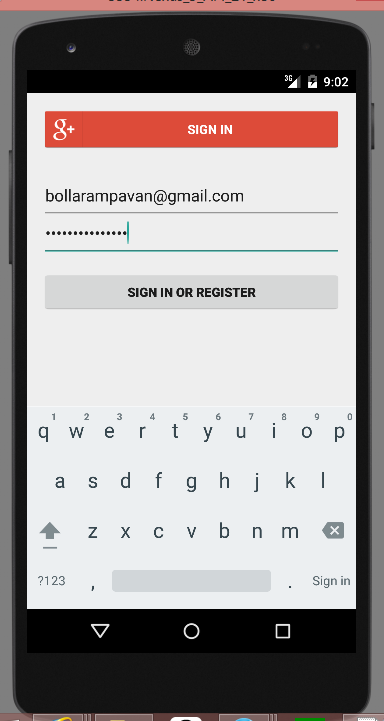
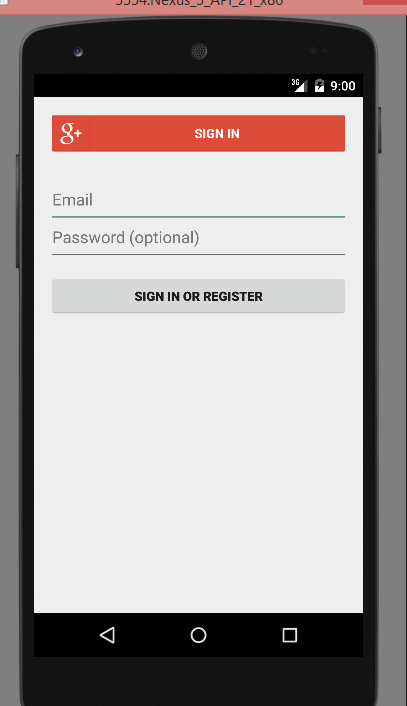
Registration API:

Registration API deals with registration of end user into application as of now we are accepting email id as primary attribute and password as second.

End user with email id and password which were given at the moment of registration can login into application as mentioned above.

Implementation of User Interface (Mobile Interface):

Screen I: Login Screen

****

Login Screen

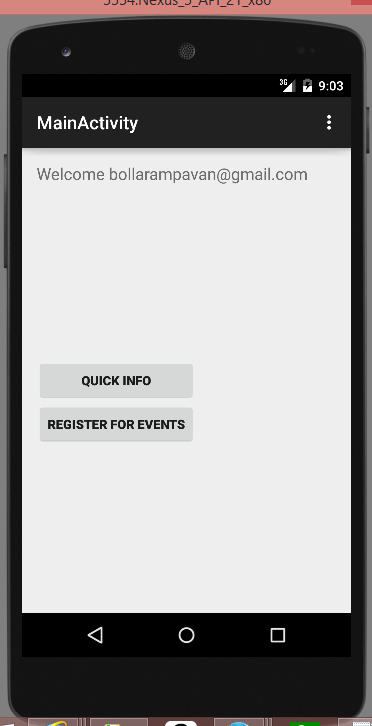
A user can login to system using existing google+ sign in services so that one can easily login with his/her Gmail id’s which reduces the use of separate database for our application .This login screen contains two separate parts which are

Sign in and Register: This button used incase user is already registered with our application. He can directly go into access of further screens. Or else if he is not registered already it generates into new screen showing you are not registered / wrong password; login again.

Email and Password:

These fields which takes input from user and validates the user details which are already into the database when registered. Once validated user can login to application it pops into his dashboard.

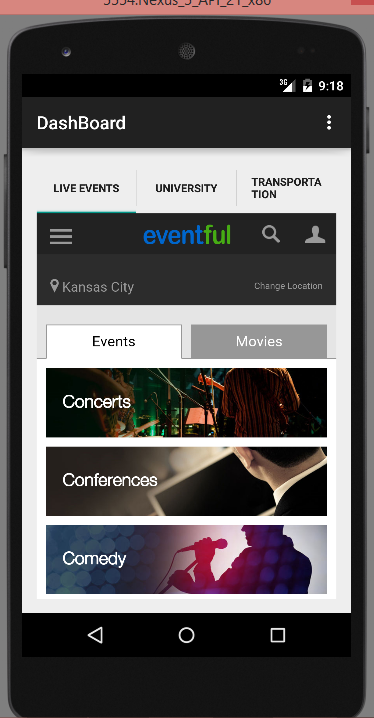
Welcome Page (Screen II)



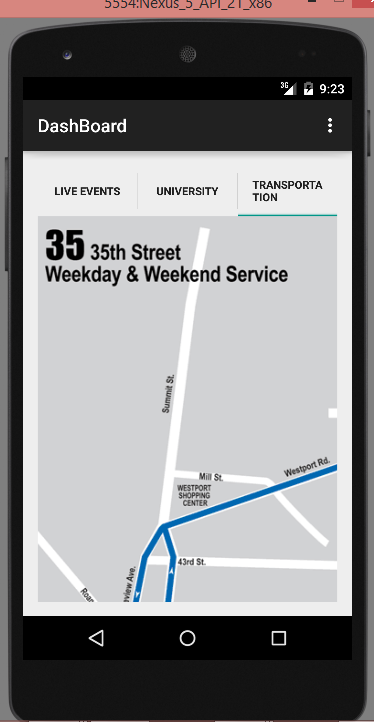
Welcome Page:

The above figure depicts the welcome page displayed to the user when he / she login into the application. We included two Events buttons **Quick-Info** and **Register for Events**. On click of Quick Info, the button navigates into dashboard screen which is categorized into Live-Events, University and Transportation.

Dashboard (Screen III)



Live Event Tab.



On Click of Transportation Tab.

DASHBOARD

Dashboard:

This is dashboard screen which will be displayed when user press the Quick info button in Screen II. It is completely mashup application where user can easily check the required tab.

We simplified GUI with simple tab host concept so that end user can easily navigate between Live-events, University and Transportation tabs. The content is displayed according to the selection of tabs.

University Dash Board(Screen IV)

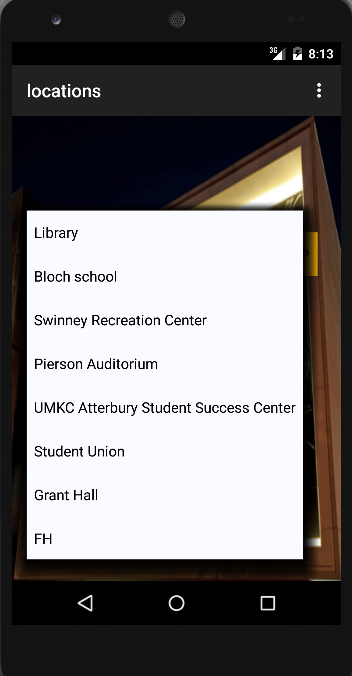
University details are presented to the user with tab in the Dashboard and when clicked navigated to different screen.

This screen consists of all data regarding university and it can be further improvised using deeper user interface

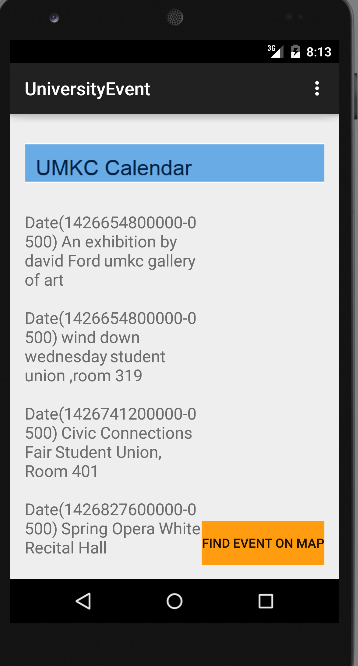
As of now we included few of the basic required data sections across screen they are as shown in the following images

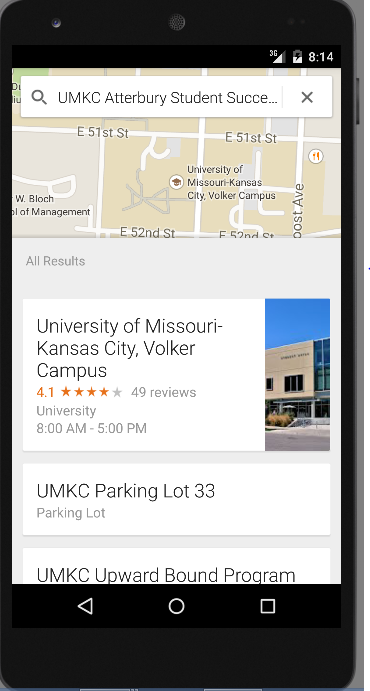
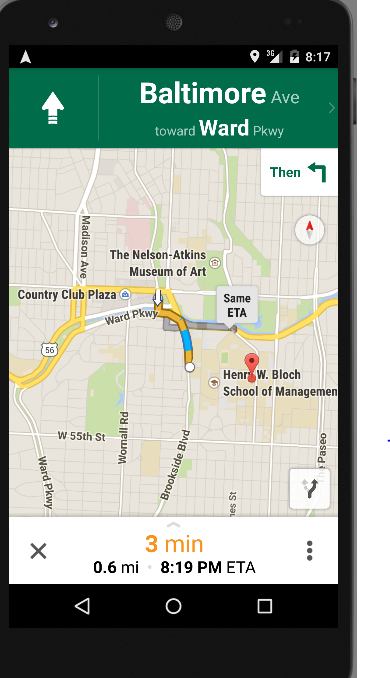
The main dashboard screen consists of data sections like About UMKC, Map, Transportation, Academic details and sports sections.



Locations page displays the event locations around university

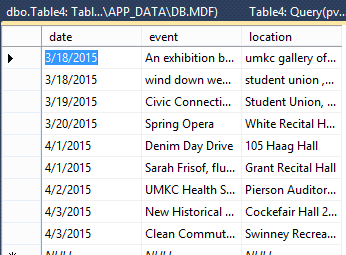
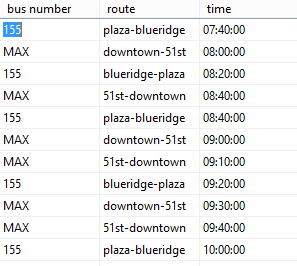
UMKC event calendar

Displays event names and their timings across all locations in UMKC

  
  
 Google maps are integrated into application to display user the routes and available navigation options.

**Database Design implementation**

Database tables are created to store all the data required by web API

Created database to store live events happening around UMKC and web service is implemented to fetch these events

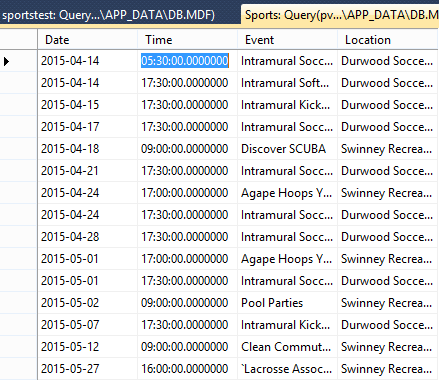
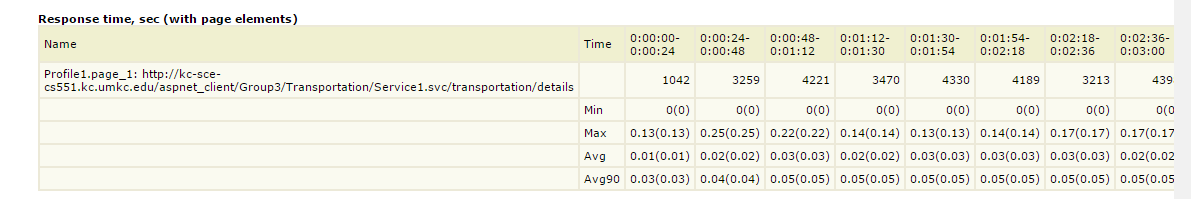
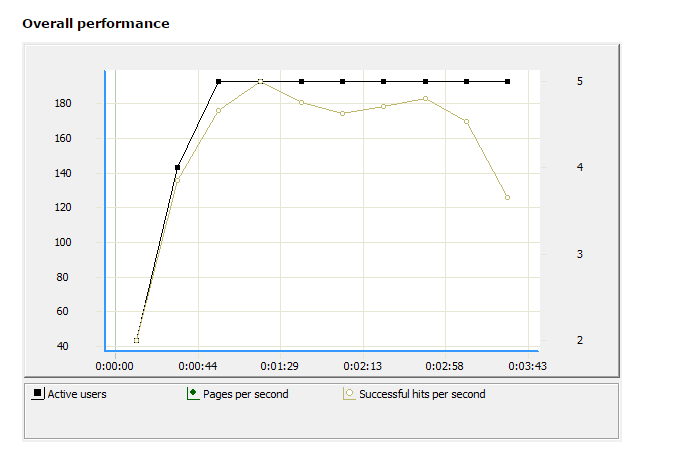
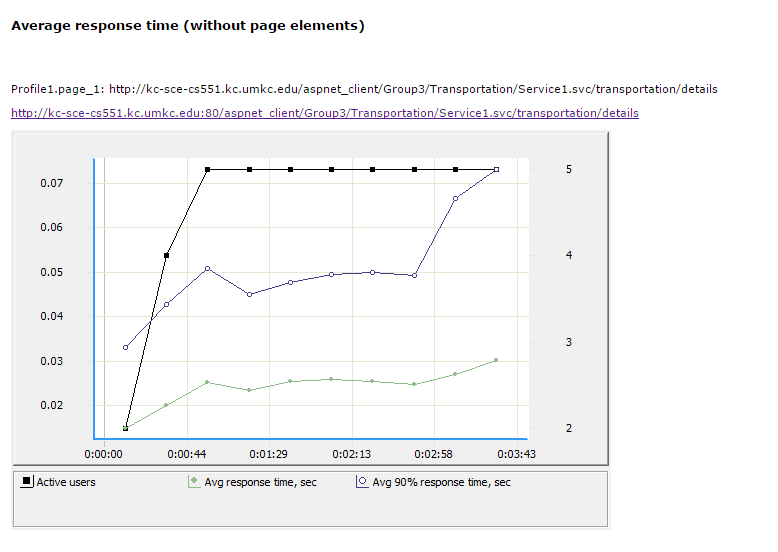


Table to store sports events data

TESTING:

WAPT is a load and stress testing tool that provides effective way to test a web page, web portal etc., We performed WAPT on web service on transportation. Here are the screenshots of testing using WAPT on link : <http://kc-sce-cs551.kc.umkc.edu/aspnet_client/Group3/Transportation/Service1.svc/transportation/details>

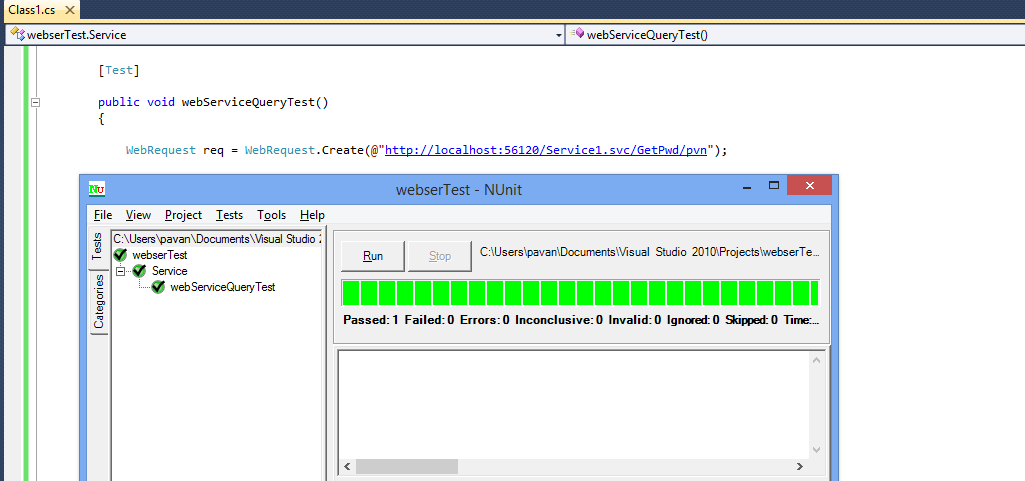




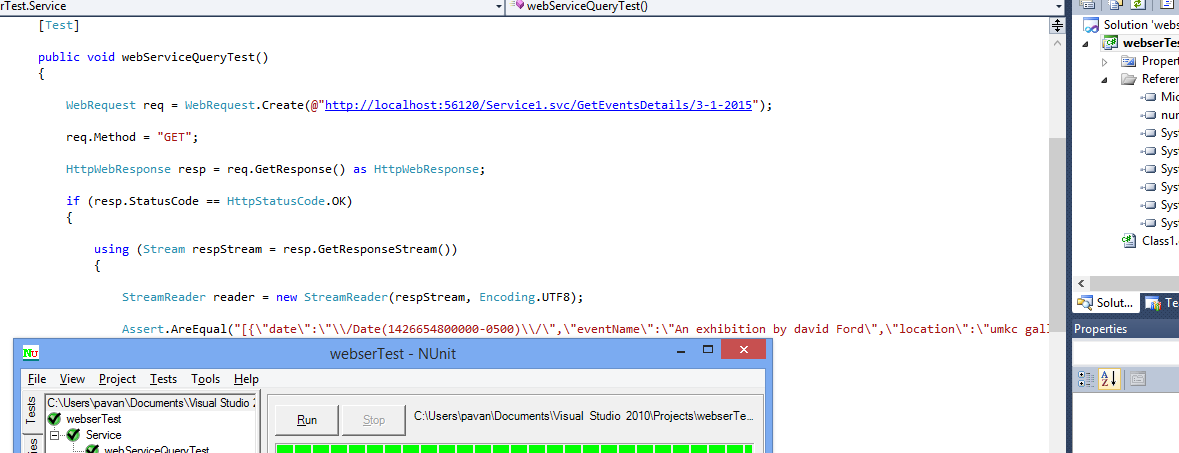
**NUnit TESTING**

Tested web API’s using NUnit testing tool which works on .NET platform and works same as Junit for Java. Successfully tested all the web service api’s

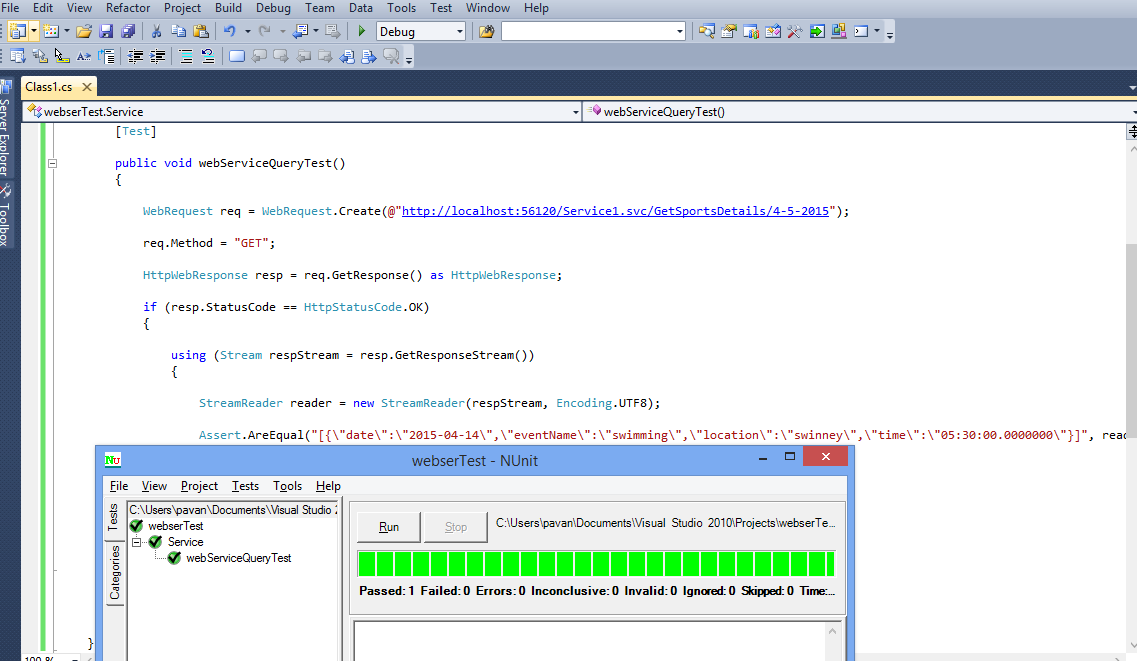
1. [**http://localhost:56120/Service1.svc/GetPwd**](http://localhost:56120/Service1.svc/GetPwd)/



**2. http://localhost:56120/Service1.svc/GetEventsDetails/3-1-2015**



**3. http://localhost:56120/Service1.svc/GetSportsDetails/4-5-2015**



**DEPLOYMENT**:

Scrum do link:

<https://www.scrumdo.com/projects/project/ase-project8/summary>

GIT Hub Link:

Source code- https://github.com/pavankumar-b/ASEspringSem/Increment3

Documentation-<https://github.com/pavankumar-b/ASEspringSem/IncrementDoc>3

Work Completed:

**Description**

* Design of Android Login Screen.
* Successfully implemented Login Functionality using web API and database. Validation of login users with already registered email and password or even when registered with Gmail.
* Created custom web API’s to provide all the info regarding university
* Integrated Google maps engine into application so that all navigation related info can be available very easily
* Creating, validating and the database tables of user registered and while signing in.
* Tables for university live events and transportation tables were created.
* Tested application using NUnit testing tool
* Performance test of application was done

**Responsibility**

Task 1: Implementation of web service / University Event API

Pavankumar Bollaram/ Varaprasad jaggu

Task 2: Implementation of web service / Transportation API

Lakshmi Priyanka / Preetham Kumar

Task 3: Implementation of web service / SportsAPI

Varaprasad jaggu / Preetham kumar /PavanKumar

Task 4: Database table creation

Pavan kumar Bollaram,varaprasad, preetham kumar, Priyanka

Task 4: User Interface (All screens)

Pavan Kumar, Preetham kumar, Varaprasad jaggu, Priyanka

Project management:

**Time Taken**

300 Man Hours

**Contribution**

Pavan Kumar Bollaram (25%)

Preetham Kumar Danaboina (25%)

Lakshmi Priyanka (25%)

Vara Prasad (25%)

VIII. Risk management

1. working with Web API specially with visual studio was bit complex
2. Emulator runs poorly
3. Very less performance testing tools for android applications.
4. Data Tables needs to refreshed regularly to make up to date information in application