Building an Android Mobile application using Bluemix Mobile First Services:

Prerequisites:

1) Bluemix Account - https://console.ng.bluemix.net

2) Android Studio - http://developer.android.com/sdk/index.html

Instructions:

Use the following steps to configure a simple Android Application:

1) Download the sample code from Github
2) Configure the mobile backend for your application
3) Configure the front end for the application
4) Run the application

Download the sample code from Github:

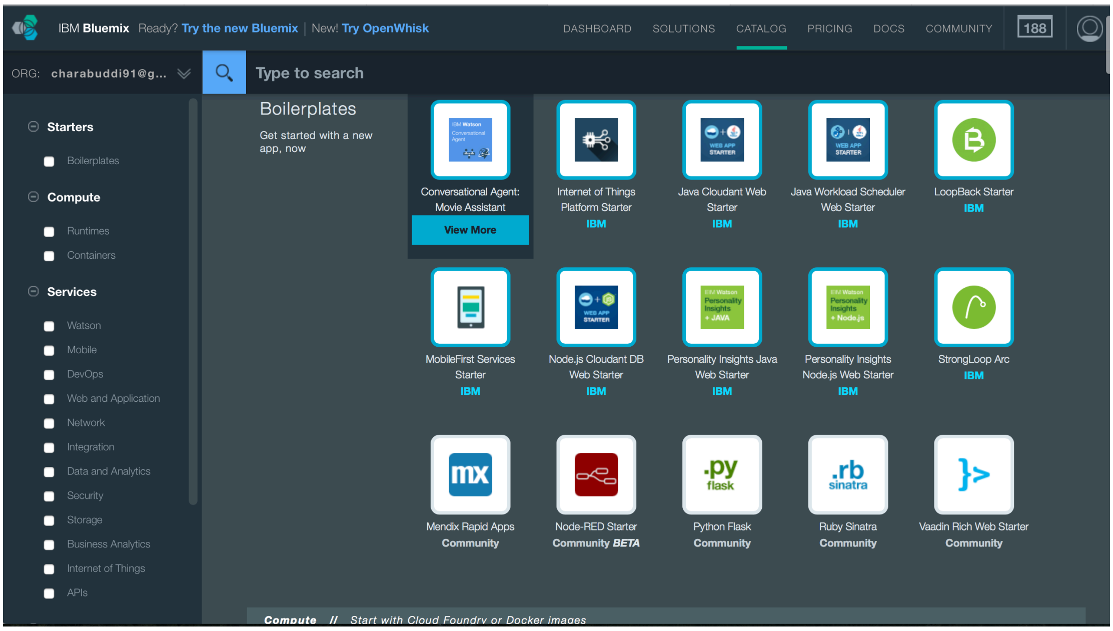
Get the sample code from Github with the following command:

git clone https://github.com/sumanth11/Bluemix-Application-using-MobileFirst-Services11.git

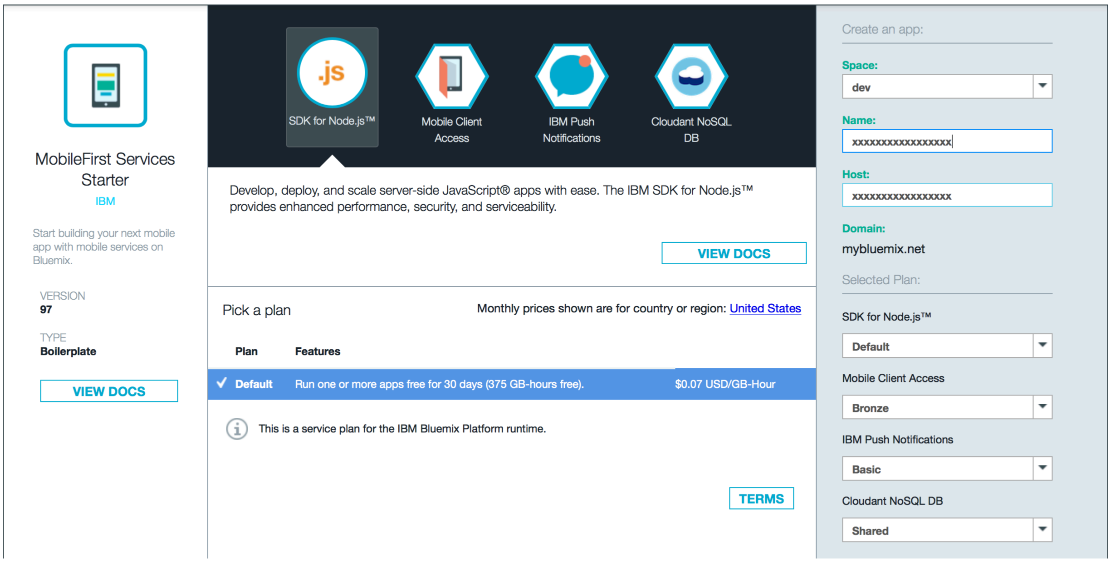
Configure the mobile backend for your application:

The following procedure shows you how to create a MobileFirst Services Starter application. Using the boilerplate to create your app automatically performs the few actions:

1) In the Boilerplates section of the Bluemix catalog, click MobileFirst Services Starter.


IBM Bluemix Ready? Try the new Bluemix | New! Try OpenWhisk
DASHBOARD
SOLUTIONS
CATALOG
PRICING
DOCS
COMMUNITY
188
ORG: charabuddi91@g...
Type to search
Starters
Boilerplates
Compute
Runtimes
Containers
Services
Watson
Mobile
DevOps
Web and Application
Network
Integration
Data and Analytics
Security
Storage
Business Analytics
Internet of Things
APIs
Boilerplates
Get started with a new app, now
Conversational Agent: Movie Assistant
View More
Internet of Things Platform Starter IBM
Java Cloudant Web Starter IBM
Java Workload Scheduler Web Starter IBM
LoopBack Starter IBM
MobileFirst Services Starter IBM
Node.js Cloudant DB Web Starter IBM
Personality Insights Java Web Starter IBM
Personality Insights Node.js Web Starter IBM
StrongLoop Arc IBM
Mendix Rapid Apps Community
Node-RED Starter Community BETA
Python Flask Community
Ruby Sinatra Community
Vaadin Rich Web Starter Community
Compute // Start with Cloud Foundry or Docker images

2) Enter a name and host for your mobile backend and click Create.



Start building your next mobile app with mobile services on Bluemix.

VERSION
97

TYPE
Boilerplate

Develop, deploy, and scale server-side JavaScript® apps with ease. The IBM SDK for Node.js™ provides enhanced performance, security, and serviceability.

Pick a plan

Monthly prices shown are for country or region: United States

Plan	Features	
✓ Default	Run one or more apps free for 30 days (375 GB-hours free).	$0.07 USD/GB-Hour

This is a service plan for the IBM Bluemix Platform runtime.

Create an app:

Space: dev

Name: xxxxxxxxxxxxxxxxx

Host: xxxxxxxxxxxxxxxxx

Domain: mybluemix.net

Selected Plan:

SDK for Node.js™: Default

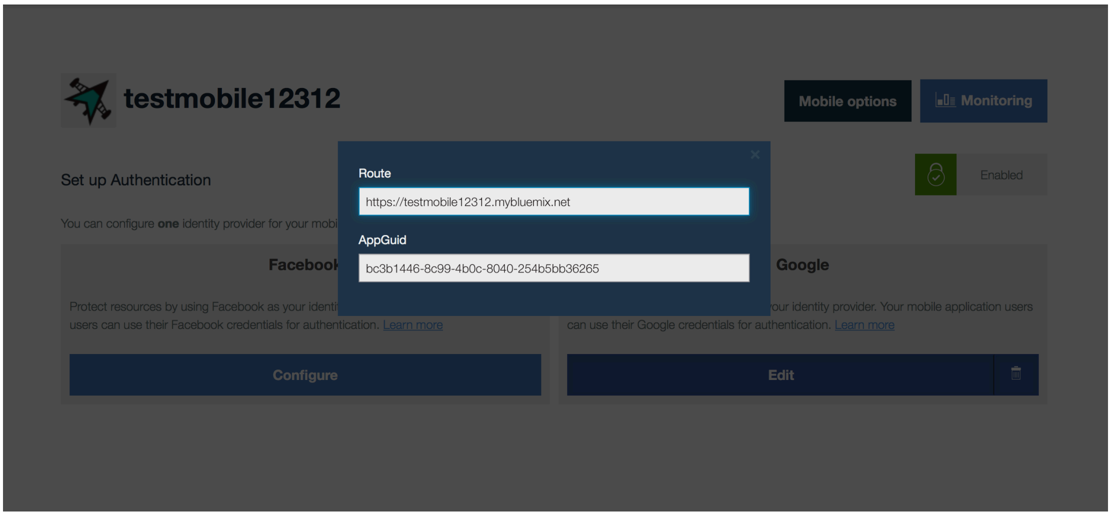
Mobile Client Access: Bronze

IBM Push Notifications: Basic

Cloudant NoSQL DB: Shared

3) Click Finish.

4) Now we need to get information about your app. you will see a new application in your dashboard. Click on the application and go to Mobile Client access service. Open it and click the Mobile Options link in top right part of a screen to find your appRoute and appGUID. Copy the appRoute and appGUID.



testmobile12312

Mobile options | Monitoring

Set up Authentication

Enabled

You can configure **one** identity provider for your mobi

Route: https://testmobile12312.mybluemix.net

AppGuid: bc3b1446-8c99-4b0c-8040-254b5bb36265

Facebook

Protect resources by using Facebook as your identi users can use their Facebook credentials for authentication. Learn more

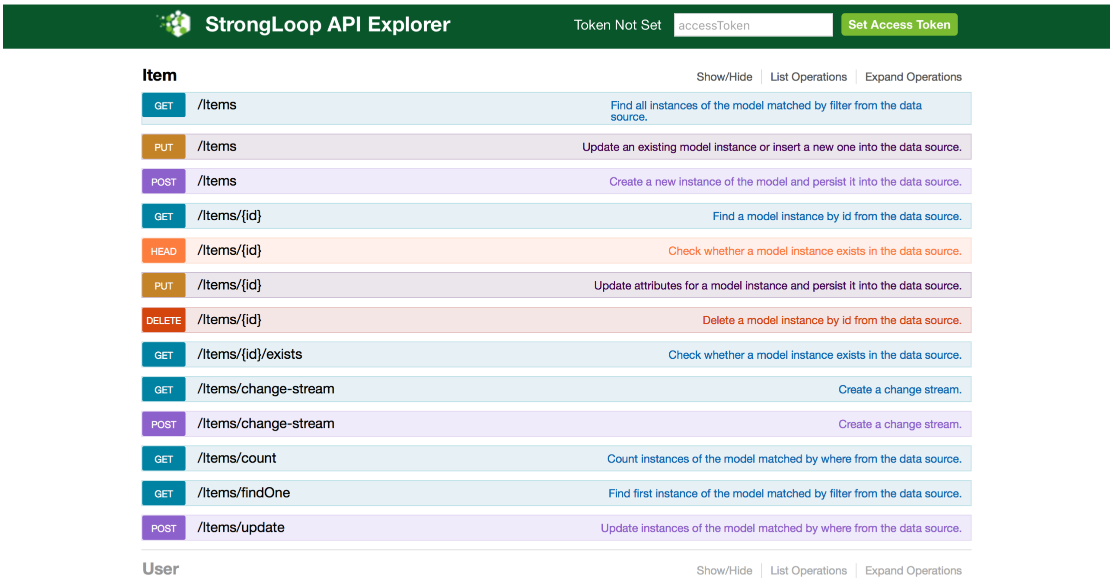
Configure

Google

our identity provider. Your mobile application users can use their Google credentials for authentication. Learn more

Edit

5) This application uses the LoopBack framework to expose the /api/Items API, which is used by both the Web UI and the sample application from this Github repository.
Adds the following services to the app: Cloudant® NoSQL DB, IBM Push Notifications, and Mobile Client Access.
Access the StrongLoop backend app
Open the appRoute URL that you copied from the Bluemix dashboard in your browser. You will see the web interface for the backend application.



StrongLoop API Explorer — Token Not Set — accessToken — Set Access Token

Item — Show/Hide | List Operations | Expand Operations

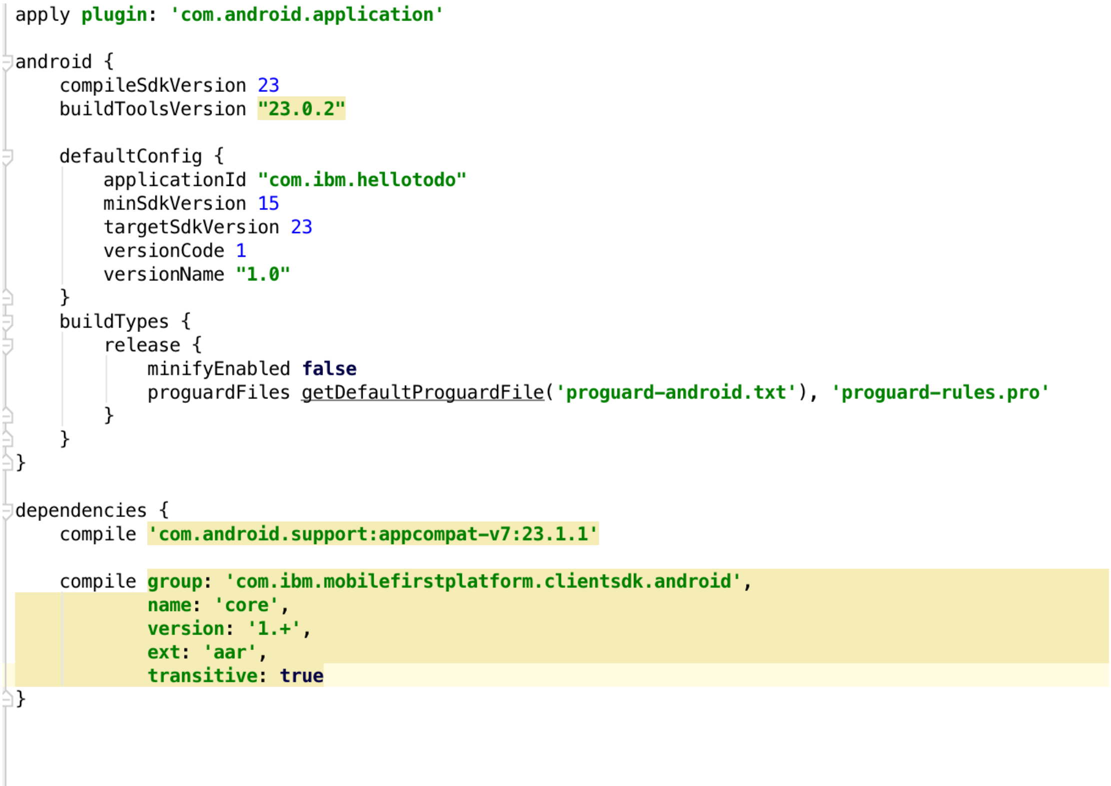
GET	/Items	Find all instances of the model matched by filter from the data source.
PUT	/Items	Update an existing model instance or insert a new one into the data source.
POST	/Items	Create a new instance of the model and persist it into the data source.
GET	/Items/{id}	Find a model instance by id from the data source.
HEAD	/Items/{id}	Check whether a model instance exists in the data source.
PUT	/Items/{id}	Update attributes for a model instance and persist it into the data source.
DELETE	/Items/{id}	Delete a model instance by id from the data source.
GET	/Items/{id}/exists	Check whether a model instance exists in the data source.
GET	/Items/change-stream	Create a change stream.
POST	/Items/change-stream	Create a change stream.
GET	/Items/count	Count instances of the model matched by where from the data source.
GET	/Items/findOne	Find first instance of the model matched by filter from the data source.
POST	/Items/update	Update instances of the model matched by where from the data source.

User — Show/Hide | List Operations | Expand Operations

Configure the front end for the application:

1) Using Android Studio, open the Bluemix-Application-using-MobileFirst-Services directory where the project was cloned.

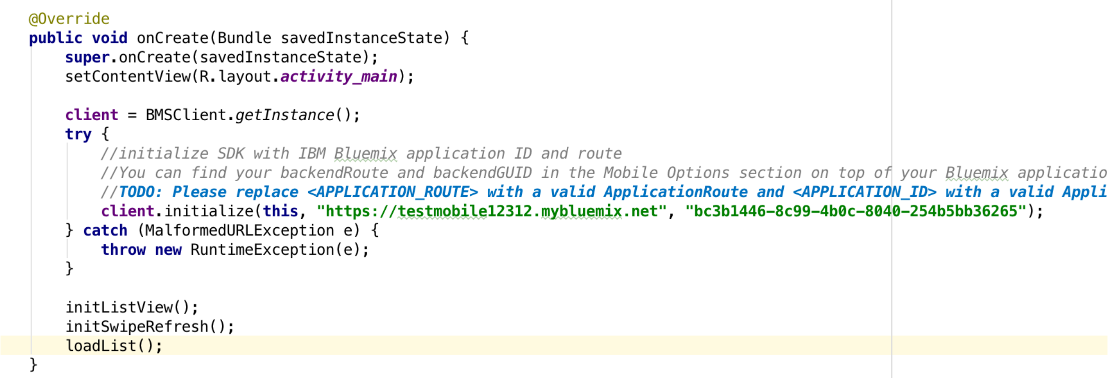
2) Run a Gradle sync (usually starts automatically) to import the required core SDK. You can view the build.gradle file in the helloTodo\app\ directory.



```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 23
    buildToolsVersion "23.0.2"

    defaultConfig {
        applicationId "com.ibm.hellotodo"
        minSdkVersion 15
        targetSdkVersion 23
        versionCode 1
        versionName "1.0"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    compile 'com.android.support:appcompat-v7:23.1.1'

    compile group: 'com.ibm.mobilefirstplatform.clientsdk.android',
            name: 'core',
            version: '1.+',
            ext: 'aar',
            transitive: true
}
```

This section in build.gradle file tells Android Studio to automatically download the Bluemix Mobile Services Core SDK and add it to the project.

3) After the gradle sync completes, open the MainActivity.java file and locate the try block within the onCreate() function. In the BMSClient.getInstance() .initialize() function, replace the <APPLICATION_ROUTE> and <APPLICATION_ID> values with the application route and ID you were given when creating your application on Bluemix.



```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    client = BMSClient.getInstance();
    try {
        //initialize SDK with IBM Bluemix application ID and route
        //You can find your backendRoute and backendGUID in the Mobile Options section on top of your Bluemix applicatio
        //TODO: Please replace <APPLICATION_ROUTE> with a valid ApplicationRoute and <APPLICATION_ID> with a valid Appli
        client.initialize(this, "https://testmobile12312.mybluemix.net", "bc3b1446-8c99-4b0c-8040-254b5bb36265");
    } catch (MalformedURLException e) {
        throw new RuntimeException(e);
    }

    initListView();
    initSwipeRefresh();
    loadList();
}
```

Run the application:

In Android Studio, click Run > Run.

This is a single view application with a simple list of to do items. If you previously added data through your web application, you will see the data is automatically pulled into the application.

References:

https://github.com/ibm-bluemix-mobile-services/bms-samples-android-hellotodo/blob/master/README.md#download-the-hellotodo-sample