

# Mobile Apps Development Training (Ionic + Firebase)

## **Trainers:**

Mark Rizam

Muhammad Shahrin Nidzam bin Effendy

# Ionic Content

- Ionic Framework Introduction
- Ionic Framework Command Line Interface (CLI)
- Ionic Framework Structure
- Ionic Life Cycle
- User Interface in Ionic
- Service in Ionic

# Ionic Framework Introduction



# Command Line Interface (CLI)

```
$ ionic start
$ ionic start --list
$ ionic start myApp
$ ionic start myApp blank
$ ionic start myApp tabs --cordova
$ ionic start myApp tabs --capacitor
$ ionic start myApp super --type=ionic-angular
$ ionic start myApp blank --type=ionic1
$ ionic start cordovaApp tabs --cordova
$ ionic start "My App" blank
$ ionic start "Conference App" https://github.com/ionic-team/ionic-conference
```

# Command Line Interface (CLI)

```
$ ionic generate  
$ ionic generate page  
$ ionic generate page contact  
$ ionic generate component contact/form  
$ ionic generate component login-form --change-detection=OnPush  
$ ionic generate directive ripple --skip-import  
$ ionic generate service api/user
```

- > e2e
- > node\_modules
- > resources
- > **src**

◆ .gitignore

{ } angular.json

🔗 config.xml

⦿ ionic.config.json

{ } package-lock.json

{ } **package.json**

📁 src.zip

TS tsconfig.json

{ } tslint.json

# Ionic Framework Structure

```

import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { RouteReuseStrategy } from '@angular/router';

import { IonicModule, IonicRouteStrategy } from '@ionic/angular';
import { SplashScreen } from '@ionic-native/splash-screen/ngx';
import { StatusBar } from '@ionic-native/status-bar/ngx';

import { AppComponent } from './app.component';
import { AppRoutingModuleModule } from './app-routing.module';
import firebaseConfig from './firebase'
import { AngularFireModule } from '@angular/fire';
import { AngularFireFirestoreModule } from '@angular/fire/firestore';
import { IonicStorageModule } from '@ionic/storage';
import { WebView } from '@ionic-native/ionic-webview/ngx';

```

```

@NgModule({
  declarations: [AppComponent],
  entryComponents: [],
  imports: [
    BrowserModule,
    IonicModule.forRoot(),
    AngularFireModule.initializeApp(firebaseConfig),
    AngularFireFirestoreModule,
    IonicStorageModule.forRoot(),
    AppRoutingModuleModule
  ],
  providers: [
    StatusBar,
    SplashScreen,
    WebView,
    { provide: RouteReuseStrategy, useClass: IonicRouteStrategy }
  ],
  bootstrap: [AppComponent]
})
export class AppModule {}

```

# Ionic app.module

```
import { NgModule } from '@angular/core';
import { PreloadAllModules, RouterModule, Routes } from '@angular/router';

const routes: Routes = [
  { path: '', redirectTo: 'home', pathMatch: 'full' },
  { path: 'home', loadChildren: './home/home.module#HomePageModule' },
  { path: 'details', loadChildren: './details/details.module#DetailsPageModule' },
];

@NgModule({
  imports: [
    RouterModule.forRoot(routes, { preloadingStrategy: PreloadAllModules })
  ],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

# Ionic app-routing



```

import { Component } from '@angular/core';
import { Platform } from '@ionic/angular';
import { SplashScreen } from '@ionic-native/splash-screen/ngx';
import { StatusBar } from '@ionic-native/status-bar/ngx';

@Component({
  selector: 'app-root',
  templateUrl: 'app.component.html'
})
export class AppComponent {
  constructor(
    private platform: Platform,
    private splashScreen: SplashScreen,
    private statusBar: StatusBar
  ) {
    this.initializeApp();
  }

  initializeApp() {
    this.platform.ready().then(() => {
      this.statusBar.styleDefault();
      this.splashScreen.hide();
    });
  }
}

```

# ionic app.component

# ionic app.component

```
<ion-app>  
|  <ion-router-outlet></ion-router-outlet>  
</ion-app>
```

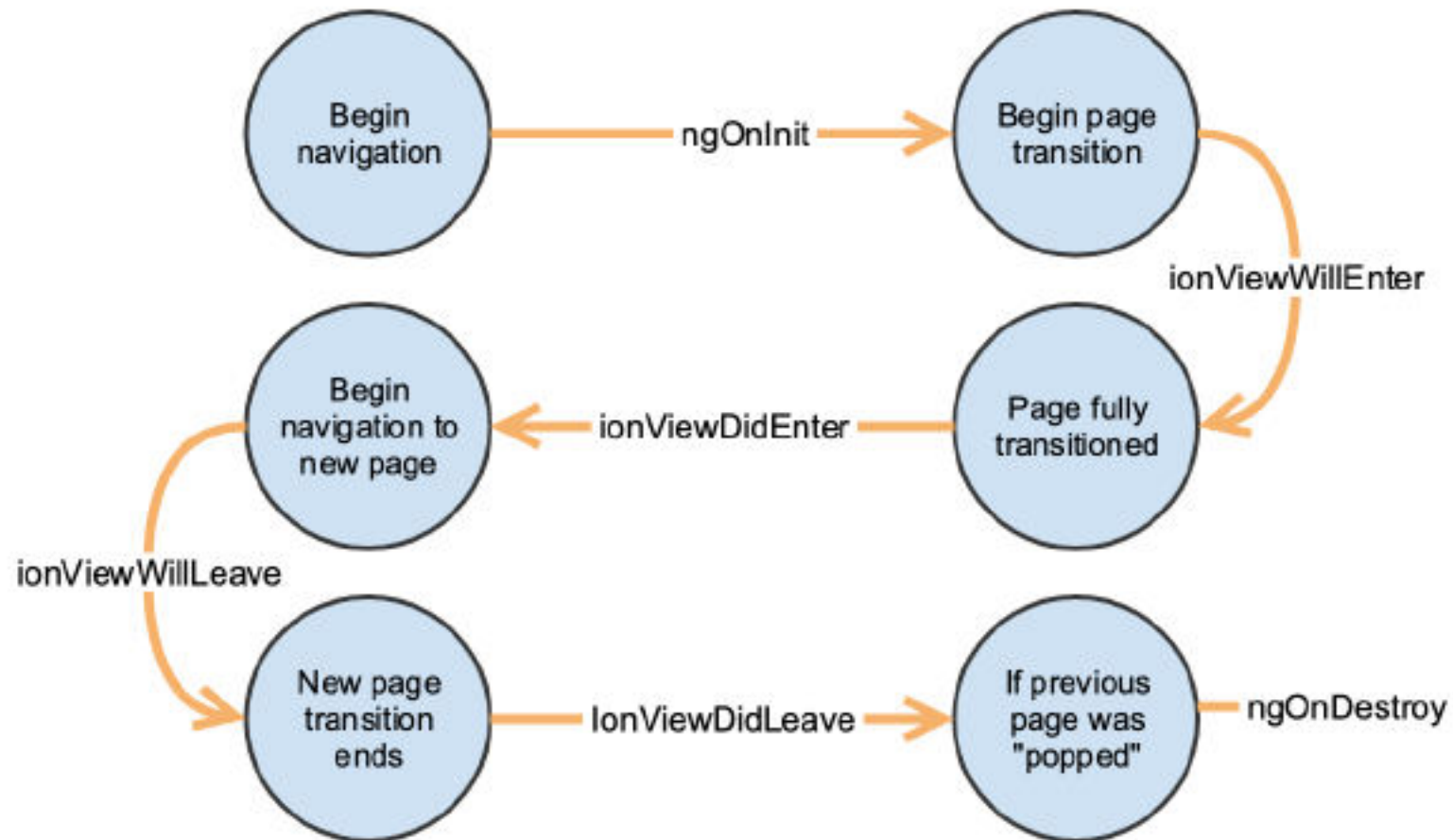
# ionic app.component

```
// App Styles
// -----
// Put style rules here that you want to apply to the entire application. These
// styles are for the entire app and not just one component. Additionally, this
// file can hold Sass mixins, functions, and placeholder classes to be imported
// and used throughout the application.

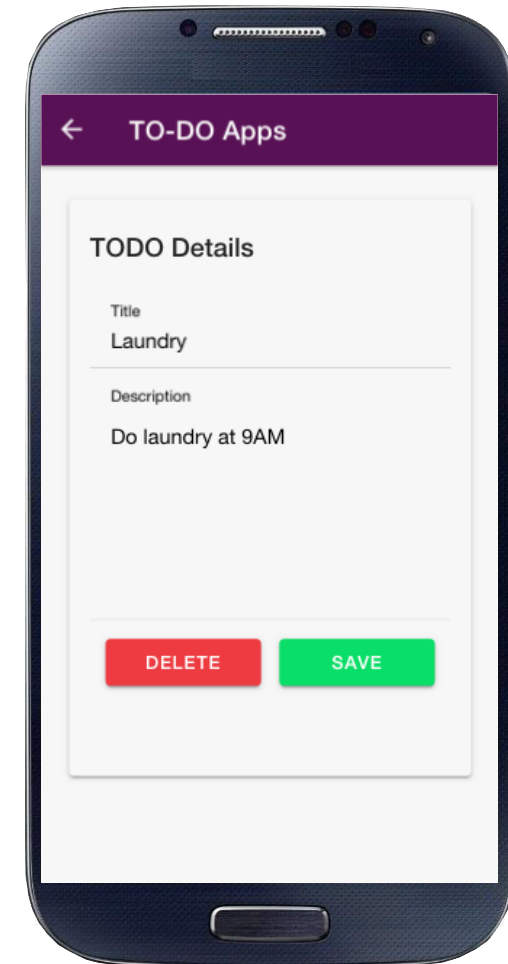
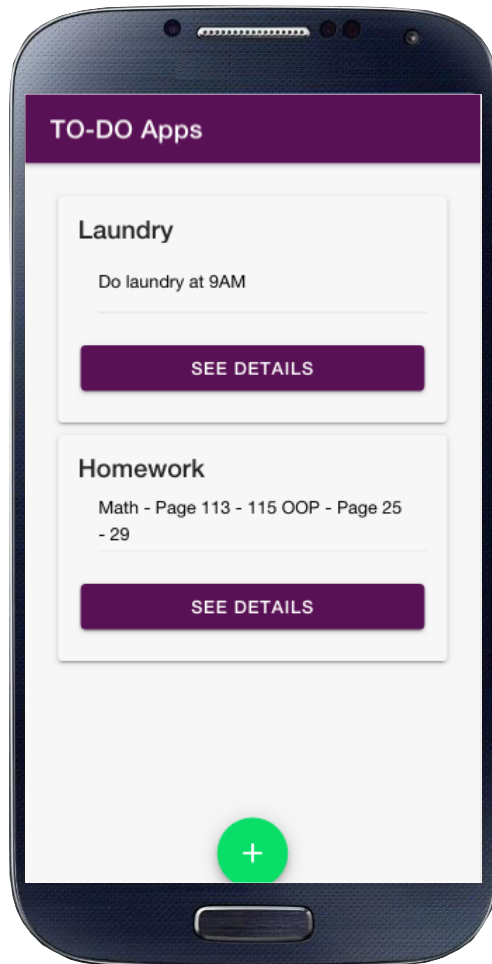
.body{
  --ion-background-color: #f7f7f7;
}

.header{
  --background: #591556;
  color: white;
}
```

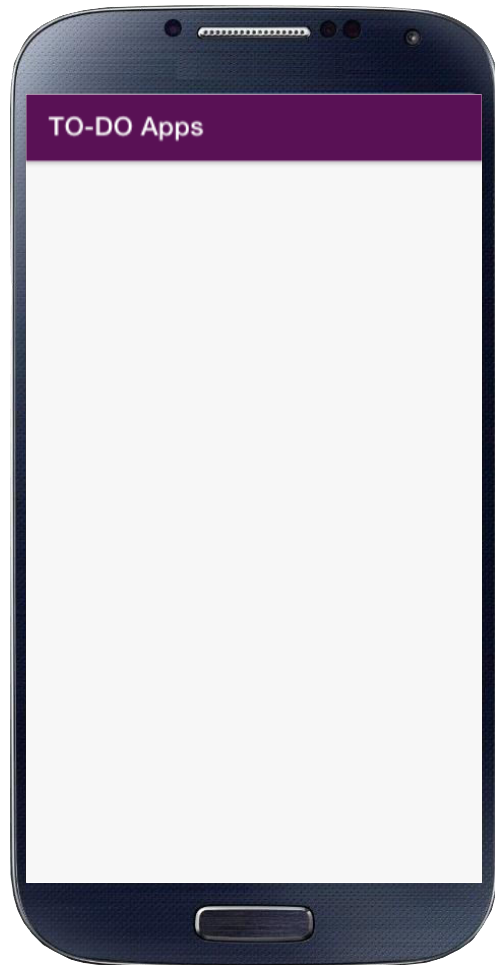
# Ionic Life Cycle



# User Interface in Ionic

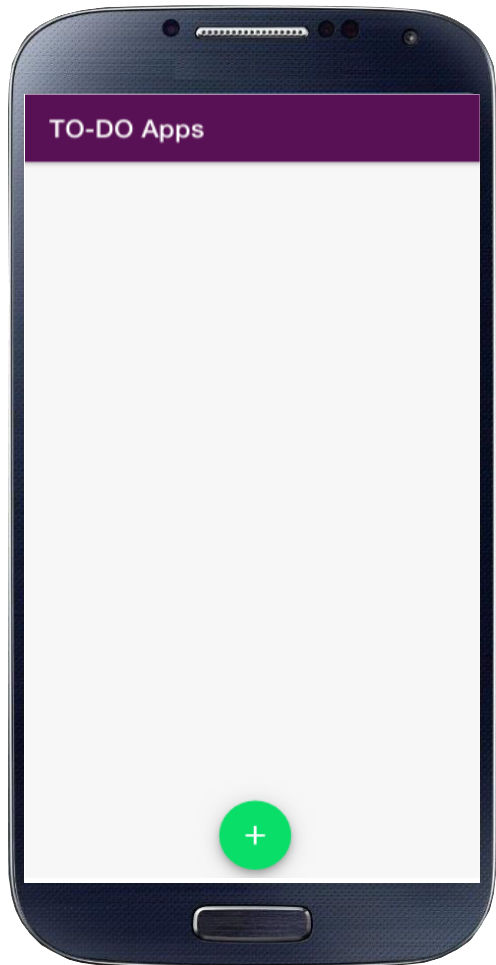


# Home



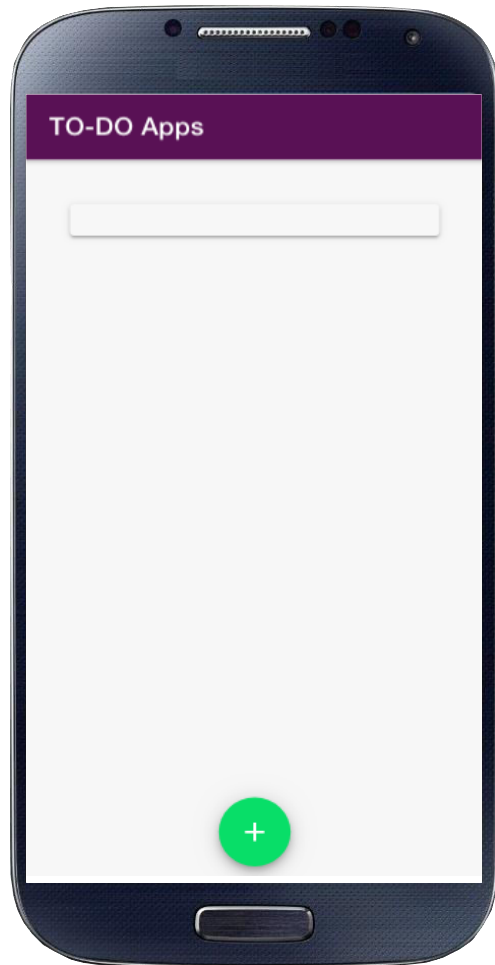
```
<ion-header>
  <ion-toolbar class="header">
    <ion-title>TO-DO Apps</ion-title>
  </ion-toolbar>
</ion-header>
<ion-content padding class="body">
</ion-content>
```

# Home



```
<ion-fab vertical="bottom" horizontal="center" slot="fixed">  
  <ion-fab-button (click)="goToDetails()" color="success">  
    <ion-icon name="add"></ion-icon>  
  </ion-fab-button>  
</ion-fab>
```

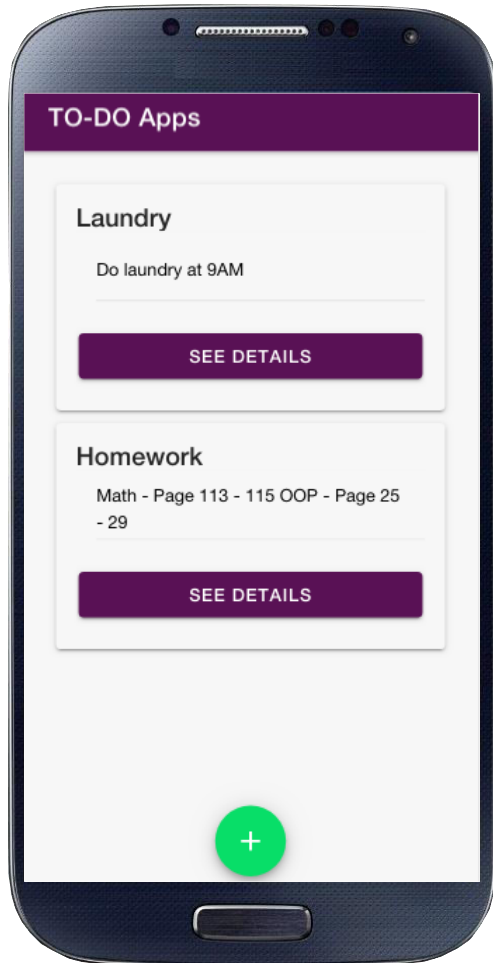
# Home



```
<ion-grid>
  <ion-row>
    <ion-col>
      <ion-card>
        <ion-card-content>
        </ion-card-content>
      </ion-card>
    </ion-col>
  </ion-row>
</ion-grid>
```

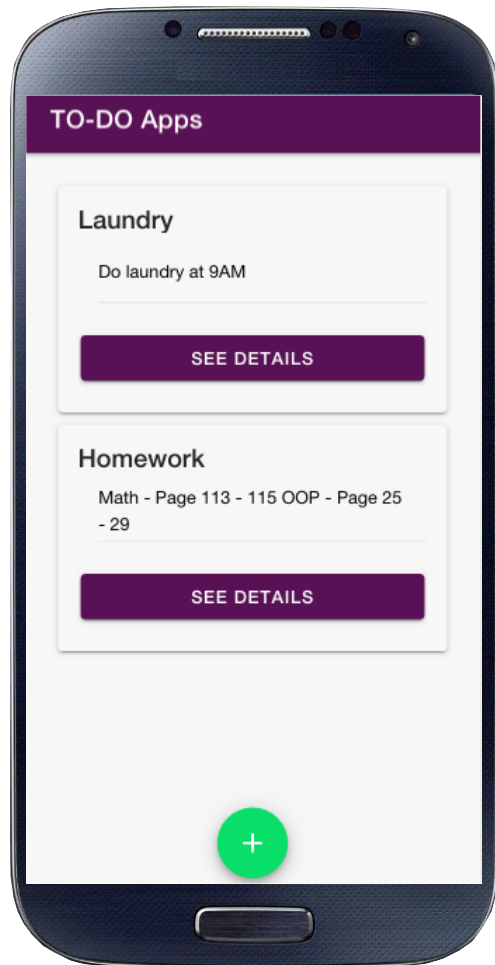


# Home



```
<ion-header>
  <ion-toolbar class="header">
    <ion-title>TO-DO Apps</ion-title>
  </ion-toolbar>
</ion-header>
<ion-content padding class="body">
  <ion-fab vertical="bottom" horizontal="center" slot="fixed">
    <ion-fab-button (click)="goToDetails()" color="success">
      <ion-icon name="add"></ion-icon>
    </ion-fab-button>
  </ion-fab>
  <ion-card>
    <ion-card-content>
      <ion-card-title>
        Laundry
      </ion-card-title>
      <ion-card-subtitle>
        <ion-list>
          <ion-item>
            <p>Do laundry at 9AM</p>
          </ion-item>
          <br/>
          <ion-button color="tertiary" expand="block">See Details</ion-button>
        </ion-list>
      </ion-card-subtitle>
    </ion-card-content>
  </ion-card>
</ion-content>
```

# Home



```
import { Component } from '@angular/core';
import { Router } from '@angular/router';
```

```
@Component({
  selector: 'app-home',
  templateUrl: 'home.page.html',
  styleUrls: ['home.page.scss'],
})
export class HomePage {

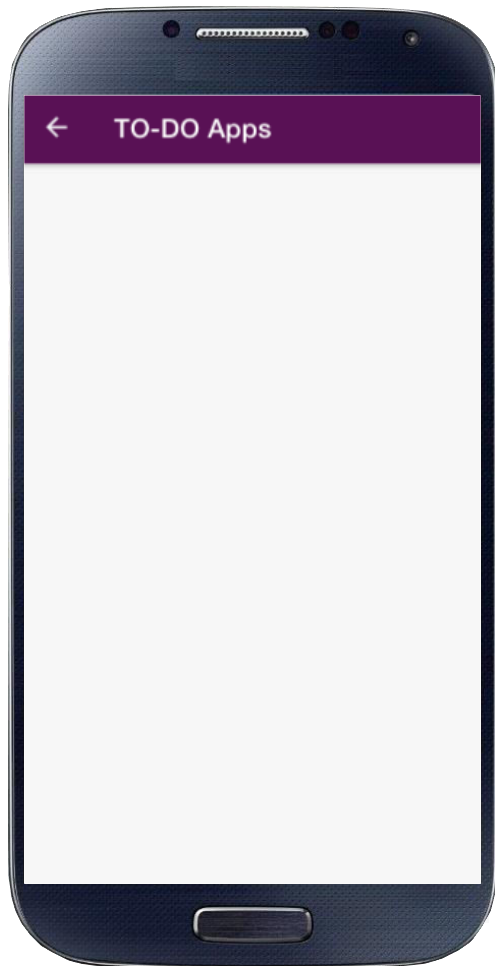
  receipts:any;
  constructor(
    private router: Router,
  ) {

  }

  goToDetails() {
    this.router.navigate(['/details']);
  }

}
```

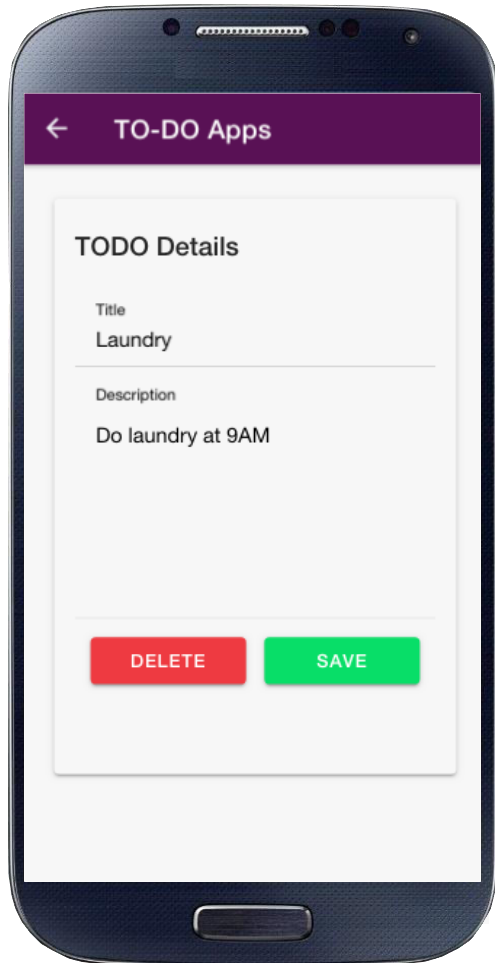
# Details



```
<ion-header>
  <ion-toolbar class="header">
    <ion-buttons slot="start">
      <ion-back-button color="light"></ion-back-button>
    </ion-buttons>
    <ion-title>TO-DO Apps</ion-title>
  </ion-toolbar>
</ion-header>

<ion-content padding class="body">
</ion-content>
```

# Details



```
<ion-header>
  <ion-toolbar class="header">
    <ion-buttons slot="start">
      <ion-back-button color="light"></ion-back-button>
    </ion-buttons>
    <ion-title>TO-DO Apps</ion-title>
  </ion-toolbar>
</ion-header>

<ion-content padding class="body">
  <ion-card>
    <ion-card-header>
      <ion-card-subtitle></ion-card-subtitle>
      <ion-card-title>TODO Details</ion-card-title>
    </ion-card-header>
    <ion-card-content>
      <ion-item>
        <ion-label position="stacked">Title</ion-label>
        <ion-input type="text" placeholder="Title"></ion-input>
      </ion-item>
      <ion-item>
        <ion-label position="stacked">Description</ion-label>
        <ion-textarea placeholder="Description" rows="6"></ion-textarea>
      </ion-item>
      <ion-grid>
        <ion-row>
          <ion-col>
            <ion-button color="danger" expand="block">Delete</ion-button>
          </ion-col>
          <ion-col>
            <ion-button color="success" expand="block">Save</ion-button>
          </ion-col>
        </ion-row>
      </ion-grid>
    </ion-card-content>
  </ion-card>
</ion-content>
```

# Service in Ionic

```
import { Injectable } from '@angular/core';
import { AlertController } from '@ionic/angular';
import { LoadingController } from '@ionic/angular';

@Injectable({
  providedIn: 'root'
})
export class CommonService {
  constructor(
    private alertController: AlertController,
    private loadingController: LoadingController
  ) { }
}
```

# Service in Ionic

```
async showAlert(title: string, content: string) {  
  const alert = await this.alertController.create({  
    header: title,  
    message: content,  
    buttons: ['OK']  
  })  
  
  await alert.present();  
}  
  
async loading(){  
  const loading = await this.loadingController.create({  
    message: 'Loading...'  
  });  
  
  return loading;  
}
```

# Firebase Content

- Firebase Introduction
- Firebase Create Project
- Ionic Firebase Integration
- Firebase Implementation
- Firebase Deploy

# Firebase Introduction

- Cloud Platform Solution
- Platform-as-a-Service (PaaS)
- Serverless Architecture
- Notable Features:
  - Authentication (Email, Social, SMS)
  - Database (Realtime Database & Cloud Store)
  - Storage
  - Cloud Hosting
  - Cloud Function
  - Crashlytics
- Owned by Google



# Firebase Introduction

The screenshot displays the Firebase console interface. On the left is a dark sidebar with the 'Firebase' logo and a navigation menu. The main content area is titled 'Database' and shows the 'Cloud Firestore' instance 'kvsepang-12096'. The 'Data' tab is selected, showing a collection named 'details' with a document ID 'UPU2xa7jPK30tPnAL8y0'. The document's content is visible, showing fields 'desc' and 'title'.

**Left Sidebar:**

- Project Overview
- Develop
  - Authentication
  - Database
  - Storage
  - Hosting
  - Functions
  - ML Kit
- Quality
  - Crashlytics
  - Performance
  - Test Lab
- Analytics
  - Spark (Free \$0/month) [Upgrade]

**Main Content Area:**

Database Cloud Firestore

kvsepang-12096 details UPU2xa7jPK30tPnAL8y0

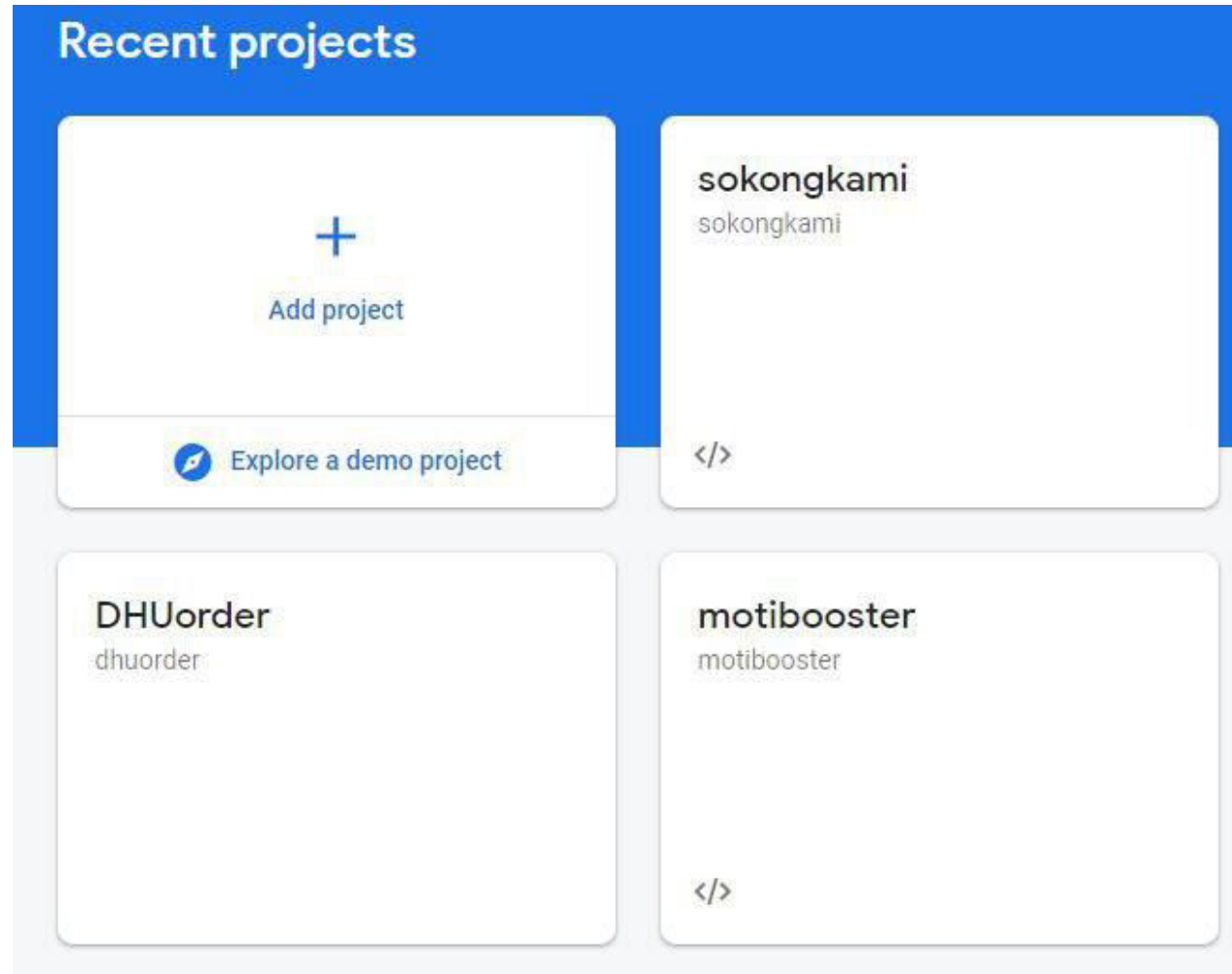
+ Start collection + Add document + Start collection

details > UPU2xa7jPK30tPnAL8y0 > + Add field

desc: "Do laundry at 9AM"

title: "Laundry"

# Firestore Create Project




# Firebase Create Project

× Create a project (Step 1 of 3)

Let's start with a name for your project

Project name

todo

 todo-c1566

Continue







# Firestore Create Project

✕ Create a project (Step 2 of 3)

## Google Analytics for your Firebase project

Google Analytics is a free and unlimited analytics solution that enables targeting, reporting, and more in Firebase Crashlytics, Cloud Messaging, In-App Messaging, Remote Config, A/B Testing, Predictions, and Cloud Functions.

Google Analytics enables:

-  A/B testing ⓘ
-  User segmentation & targeting across Firebase products ⓘ
-  Predicting user behavior ⓘ
-  Crash-free users ⓘ
-  Event-based Cloud Functions triggers ⓘ
-  Free unlimited reporting ⓘ

☒ Enable Google Analytics for this project  
Recommended

[Previous](#)


[Continue](#)

# Firebase Create Project

× Create a project (Step 3 of 3)

## Configure Google Analytics

Choose or create a Google Analytics account ⓘ

 Default Account for Firebase ▼

Upon project creation, a new Google Analytics property will be created in your chosen Google Analytics account and linked to your Firebase project. This link will enable data flow between the products. Data exported from your Google Analytics property into Firebase is subject to the Firebase terms of service, while Firebase data imported into Google Analytics is subject to the Google Analytics terms of service. [Learn more](#)

[Previous](#)

[Create project](#)

# Firebase Create Project

The screenshot displays the Firebase console interface for a project named "todo". On the left, a dark sidebar contains the "Firebase" logo and a navigation menu with sections: "Project Overview" (with a gear icon), "Develop" (listing Authentication, Database, Storage, Hosting, Functions, and ML Kit), "Quality" (listing Crashlytics, Performance, and Test Lab), and "Analytics" (listing Spark with "Free \$0/month" and an "Upgrade" button). The main content area has a blue background with the text "Get started by adding Firebase to your app". Below this text are four circular icons for "iOS", "Android", "Code" (represented by a code symbol), and "Cloud" (represented by a cloud icon). Under these icons is the text "Add an app to get started". At the bottom of the main area is a light gray banner that reads "Store and sync app data in milliseconds" with a close button (X) on the right. The top right of the interface includes a "Go to docs" link, a notification bell, and a user profile picture.

firebase

Project Overview

Develop

- Authentication
- Database
- Storage
- Hosting
- Functions
- ML Kit

Quality

- Crashlytics
- Performance
- Test Lab

Analytics

Spark  
Free \$0/month Upgrade

todo

Spark plan

Go to docs

Get started by adding Firebase to your app

iOS Android Code Cloud

Add an app to get started

Store and sync app data in milliseconds

# Ionic Firebase Integration



```
$ ionic cordova plugin add cordova-plugin-firebase  
$ npm install @ionic-native/firebase
```

```
npm install firebase @angular/fire --save
```

# Ionic Firebase Integration

The screenshot displays the Firebase console interface. On the left is a dark sidebar with the 'Firebase' logo and a navigation menu. The menu includes 'Project Overview', 'Develop' (with sub-items: Authentication, Database, Storage, Hosting, Functions, ML Kit), 'Quality' (with sub-items: Crashlytics, Performance, Test Lab), and 'Analytics' (with sub-items: Spark, Free \$0/month, and an 'Upgrade' button). The main content area is titled 'Settings' and shows a dropdown menu with 'Project settings' and 'Users and permissions'. The 'Project settings' section includes fields for 'Public-facing name' (set to 'project-849053871948') and 'Support email' (set to 'Not configured'). Below this is a section titled 'Your apps' which contains the message 'There are no apps in your project' and 'Select a platform to get started'. To the right of this message are four circular icons: 'iOS', 'Android', 'Web' (represented by a code symbol), and 'Flutter'. At the bottom right of the console, there is a 'Delete project' button with a trash can icon.

firebase

Project Overview

Develop

- Authentication
- Database
- Storage
- Hosting
- Functions
- ML Kit

Quality

- Crashlytics
- Performance
- Test Lab

Analytics

Spark

Free \$0/month

Upgrade

Settings

Project settings

Users and permissions

Public-facing name ? project-849053871948

Support email ? Not configured

Your apps

There are no apps in your project

Select a platform to get started

iOS Android Web Flutter

Delete project



# Ionic Firebase Integration

## × Add Firebase to your web app

### 1 Register app

App nickname [?](#)

todo

☒ Also set up **Firebase Hosting** for this app. [Learn more](#) [↗](#)

Hosting can also be set up later. It's free to get started anytime.

 todo-c1566 (No deploys yet) [↕](#)

Register app

### 2 Add Firebase SDK

### 3 Install Firebase CLI

### 4 Deploy to Firebase Hosting

# Ionic Firebase Integration

## × Add Firebase to your web app

- ✓ Register app
- ✓ Add Firebase SDK
- 3 Install Firebase CLI

To host your site with Firebase Hosting, you need the Firebase CLI (a command line tool).

Run the following [npm](#)  command to install the CLI or update to the latest CLI version.

```
$ npm install -g firebase-tools
```



Doesn't work? Take a look at the [Firebase CLI reference](#)  or change your [npm permissions](#) .

Next

- 4 Deploy to Firebase Hosting

# Ionic Firebase Integration

## Firebase SDK snippet

☐ Automatic  ☐ CDN  ☒ Config 

Copy and paste these scripts into the bottom of your <body> tag, but before you use any Firebase services:

```
const firebaseConfig = {  
  apiKey: "AIzaSyC8W55xWz5WEqzZV24kL65co1W4XK33aEE",  
  authDomain: "todo-c1566.firebaseio.com",  
  databaseURL: "https://todo-c1566.firebaseio.com",  
  projectId: "todo-c1566",  
  storageBucket: "",  
  messagingSenderId: "849053871948",  
  appId: "1:849053871948:web:a3721fe496e971ab8629e9"  
};
```



# Ionic Firebase Integration

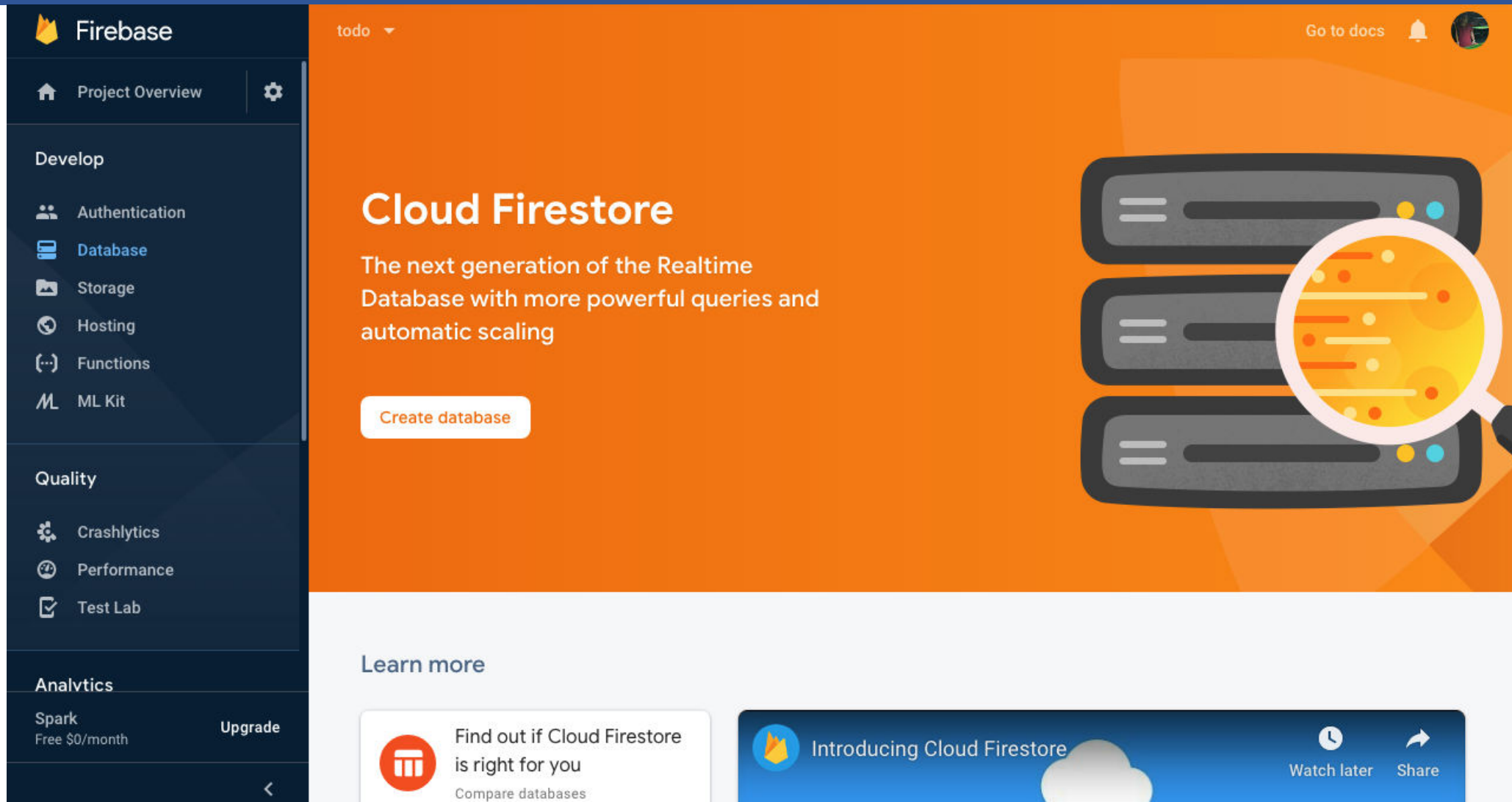
```
const config = {  
  apiKey: "AIzaSyAYtferixUJ_ZsWjaJKPsv5THyD3Eghlz4",  
  authDomain: "kvsepang-12096.firebaseio.com",  
  databaseURL: "https://kvsepang-12096.firebaseio.com",  
  projectId: "kvsepang-12096",  
  storageBucket: "kvsepang-12096.appspot.com",  
  messagingSenderId: "530965130177",  
  appId: "1:530965130177:web:98eef19f01084742f71d40"  
};  
  
export default config
```

# Ionic Firebase Integration

```
import firebaseConfig from './firebase'  
import { AngularFireModule } from '@angular/fire';  
import { AngularFirestoreModule } from '@angular/fire/firestore';
```

```
imports: [  
  BrowserModule,  
  IonicModule.forRoot(),  
  AngularFireModule.initializeApp(firebaseConfig),  
  AngularFirestoreModule,  
  IonicStorageModule.forRoot(),  
  AppRoutingModule  
],
```

# Ionic Firebase Integration



The screenshot displays the Firebase console interface. On the left is a dark sidebar with the 'Firebase' logo and a 'Project Overview' section containing icons for Authentication, Database, Storage, Hosting, Functions, and ML Kit. Below these are sections for 'Quality' (Crashlytics, Performance, Test Lab) and 'Analytics' (Spark, with a 'Free \$0/month' label and an 'Upgrade' button). The main content area has an orange header with 'todo' and 'Go to docs' links. The central focus is the 'Cloud Firestore' section, which includes the text 'The next generation of the Realtime Database with more powerful queries and automatic scaling' and a 'Create database' button. To the right of this text is an illustration of three server racks, with a magnifying glass highlighting a glowing yellow sphere containing data points. At the bottom, a 'Learn more' section features two cards: one for a database comparison tool and another for a video titled 'Introducing Cloud Firestore' with 'Watch later' and 'Share' options.

**Firebase**

Project Overview

**Develop**

- Authentication
- Database
- Storage
- Hosting
- Functions
- ML Kit

**Quality**

- Crashlytics
- Performance
- Test Lab

**Analytics**

Spark  
Free \$0/month

Upgrade

todo

Go to docs

## Cloud Firestore

The next generation of the Realtime Database with more powerful queries and automatic scaling

Create database

**Learn more**

Find out if Cloud Firestore is right for you  
Compare databases

Introducing Cloud Firestore

Watch later Share

# Ionic Firebase Integration

## Create database

1 Secure rules for Cloud Firestore


2 Set Cloud Firestore location

After you define your data structure, you will need to write rules to secure your data.  
[Learn more](#)

☐ **Start in locked mode**  
Make your database private by denying all reads and writes

☒ **Start in test mode**  
Get set up quickly by allowing all reads and writes to your database

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write;
    }
  }
}
```

 **Anyone with your database reference will be able to read or write to your database**

Enabling Cloud Firestore will prevent you from using Cloud Datastore with this project, notably from the associated App Engine app

CancelNext

# Ionic Firebase Integration

## Create database

✓ Secure rules for Cloud Firestore

2 Set Cloud Firestore location

Your location setting is where your Cloud Firestore data will be stored.

⚠

After you set this location, you cannot change it later. Also, this location setting will be the location for your default Cloud Storage bucket.

Learn more

Cloud Firestore location

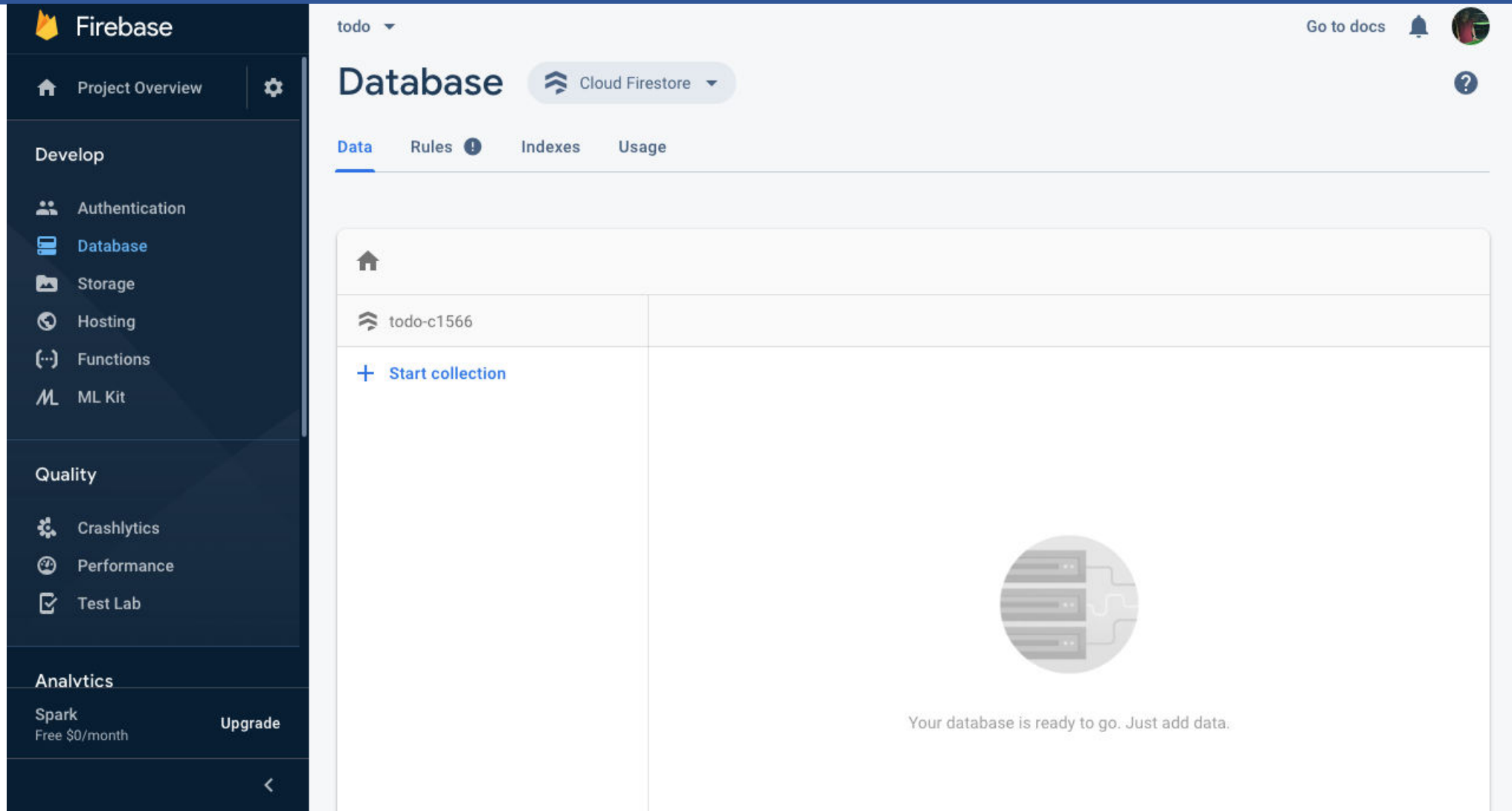
nam5 (us-central) ▼

Enabling Cloud Firestore will prevent you from using Cloud Datastore with this project, notably from the associated App Engine app

CancelDone



# Ionic Firebase Integration



The screenshot displays the Firebase console interface. On the left is a dark sidebar with the 'Firebase' logo and a navigation menu. The menu includes 'Project Overview', a 'Develop' section with links to Authentication, Database (highlighted), Storage, Hosting, Functions, and ML Kit, a 'Quality' section with Crashlytics, Performance, and Test Lab, and an 'Analytics' section with Spark (Free \$0/month) and an 'Upgrade' button. The main content area is titled 'Database' and shows the 'todo' project selected. A dropdown menu indicates 'Cloud Firestore' is the active database. Below this are tabs for 'Data', 'Rules', 'Indexes', and 'Usage', with 'Data' being the active tab. The main workspace shows a home icon, the project ID 'todo-c1566', and a '+ Start collection' button. A large circular graphic with server icons is centered, accompanied by the text 'Your database is ready to go. Just add data.'

firebase

Project Overview

Develop

- Authentication
- Database
- Storage
- Hosting
- Functions
- ML Kit

Quality

- Crashlytics
- Performance
- Test Lab

Analytics

Spark  
Free \$0/month

Upgrade

todo

Database

Cloud Firestore

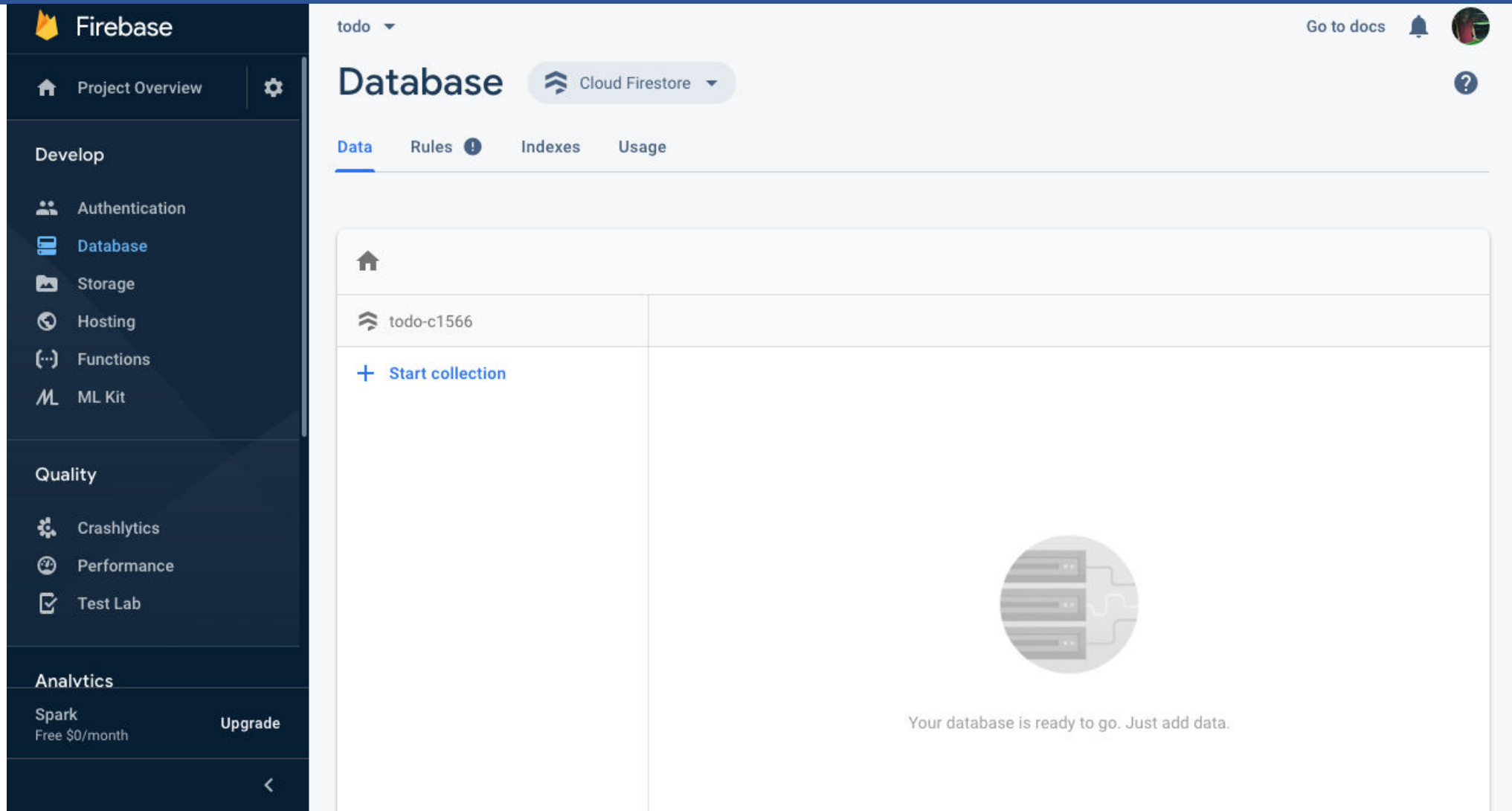
Data Rules Indexes Usage

todo-c1566

+ Start collection

Your database is ready to go. Just add data.

# Ionic Firebase Integration



The screenshot displays the Firebase console interface. On the left is a dark sidebar with the 'Firebase' logo and a navigation menu. The menu includes 'Project Overview', a 'Develop' section with links to Authentication, Database (highlighted), Storage, Hosting, Functions, and ML Kit, a 'Quality' section with links to Crashlytics, Performance, and Test Lab, and an 'Analytics' section with a link to Spark (labeled 'Free \$0/month' and 'Upgrade'). The main content area is titled 'Database' and shows the 'todo' project selected. It features tabs for 'Data', 'Rules', 'Indexes', and 'Usage', with 'Data' being the active tab. Below the tabs, a home icon is visible, followed by the project ID 'todo-c1566' and a '+ Start collection' button. A large circular graphic with server icons is centered on the page, accompanied by the text 'Your database is ready to go. Just add data.'

firebase

Project Overview

Develop

- Authentication
- Database
- Storage
- Hosting
- Functions
- ML Kit

Quality

- Crashlytics
- Performance
- Test Lab

Analytics

Spark  
Free \$0/month Upgrade

todo

Database Cloud Firestore

Data Rules Indexes Usage

+ Start collection

Your database is ready to go. Just add data.

# Firestore Implementation (Ionic Service)

```
import { Injectable } from '@angular/core';
import { AngularFireStore } from '@angular/fire/firestore';

@Injectable({
  providedIn: 'root'
})
export class FirestoreService {

  constructor(
    private afstore: AngularFireStore,
  ) { }
}
```

# Firebase Implementation (Create Query)

```
async create(details){  
  try{  
    const receipt = await this.afstore.collection('details').add(details);  
    return {  
      value: receipt.id,  
      success: true  
    }  
  } catch (err){  
    return {  
      value: err.message,  
      success: false  
    }  
  }  
}
```

# Firestore Implementation (Read Query)

```
async read(){
  try{
    const receipt = await this.afstore.collection("details").snapshotChanges();
    return {
      value: receipt,
      success: true
    }
  } catch (err){
    return {
      value: err.message,
      success: false
    }
  }
}
```

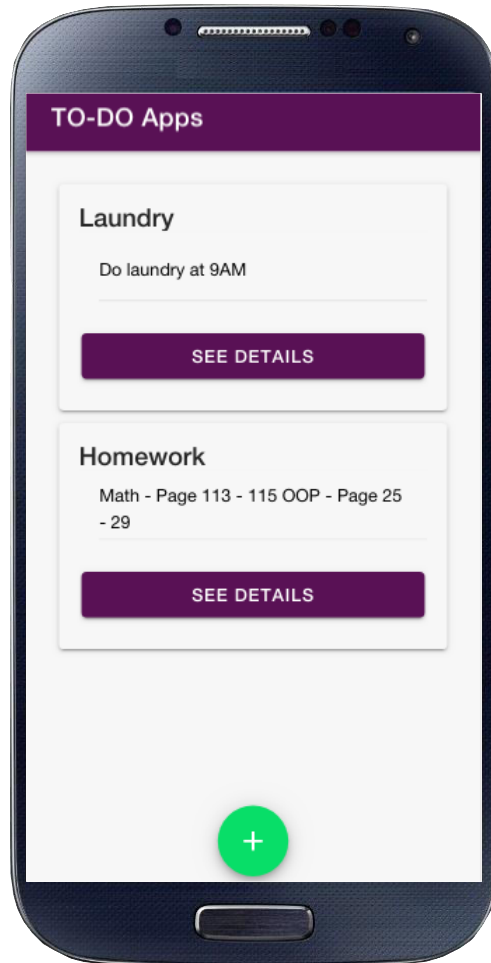
# Firestore Implementation (Update Query)

```
async update(details, id){
  try{
    delete details.id;
    const rec = await this.afstore.doc('details/' + id).set(details);
    return {
      value: rec,
      success: true
    }
  } catch (err){
    return {
      value: err.message,
      success: false
    }
  }
}
```

# Firestore Implementation (Delete Query)

```
async delete(id){
  try{
    const rec = await this.afstore.doc('details/' + id).delete();
    return {
      value: rec,
      success: true
    }
  } catch (err) {
    return {
      value: err.message,
      success: false
    }
  }
}
```

# Home



```
<ion-grid>
  <ion-row>
    <ion-col *ngFor="let item of receipts">
      <ion-card>
        <ion-card-content>
          <ion-card-title>
            {{item.title}}
          </ion-card-title>
          <ion-card-subtitle>
            <ion-list>
              <ion-item>
                <p>{{item.desc}}</p>
              </ion-item>
            <br/>
            <ion-button color="tertiary" (click)="toDetails(item)" expand="block">
              See Details
            </ion-button>
          </ion-list>
        </ion-card-subtitle>
      </ion-card-content>
    </ion-card>
  </ion-col>
</ion-row>
</ion-grid>
```



# Home

```
import { CommonService } from '../service/common.service';  
import { FirebaseService } from '../service/firebase.service';
```

```
receipts:any;  
constructor(  
  private router: Router,  
  private common : CommonService,  
  private fb: FirebaseService  
) {  
  this.subscribeToReceipt();  
}
```

# Home

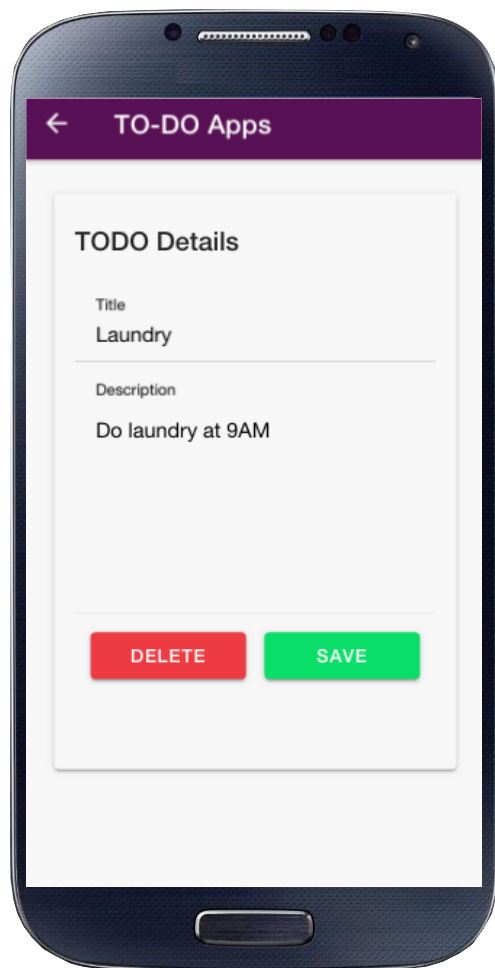
```
async subscribeToReceipt(){
  let loading = await this.common.loading();
  loading.present();
  this.receipts = [];
  let receipt = await this.fb.read();

  if(receipt.success){
    receipt.value.subscribe(res => {
      console.log(res);
      this.receipts = [];
      res.map(r => {
        let temp = Object.assign({id:r.payload.doc.id}, r.payload.doc.data());
        console.log(temp);
        this.receipts.push(temp);
      });
      loading.dismiss();
      // this.receipts = res;
      console.log(this.receipts)
    });
  } else {
    loading.dismiss();
    this.common.presentAlert("Error", receipt.value);
  }
}
```

```
toDetails(obj){
  this.common.setDetailId(obj.id);
  this.common.setReceipt(obj);
  this.router.navigate(['/details']);
}
```

```
goToDetails() {
  this.common.setDetailId("0");
  this.router.navigate(['/details']);
}
```

# Details



```
<ion-content padding class="body">
  <ion-card>
    <ion-card-header>
      <ion-card-subtitle></ion-card-subtitle>
      <ion-card-title>TODO Details</ion-card-title>
    </ion-card-header>
    <ion-card-content>
      <ion-item>
        <ion-label position="stacked">Title</ion-label>
        <ion-input type="text" placeholder="Title" [(ngModel)]="details.title"></ion-input>
      </ion-item>
      <ion-item>
        <ion-label position="stacked">Description</ion-label>
        <ion-textarea placeholder="Description" [(ngModel)]="details.desc" rows="6"></ion-textarea>
      </ion-item>
      <ion-grid>
        <ion-row>
          <ion-col>
            <ion-button color="danger" expand="block" (click)="delete()" *ngIf="deletIf()">
              Delete
            </ion-button>
          </ion-col>
          <ion-col>
            <ion-button color="success" expand="block" (click)="save()">Save</ion-button>
          </ion-col>
        </ion-row>
      </ion-grid>
    </ion-card-content>
  </ion-card>
```

# Details

```
import { CommonService } from '../service/common.service';
import { FirebaseService } from '../service/firebase.service';

result:any;
details:{title:string, desc:string};
tag:any;
detailId:any;
constructor(
  private common : CommonService,
  private fb:FirebaseService,
  private navCtrl: NavController
) { }
```

# Details

```
ngOnInit() {  
  this.initValue();  
  if(!(this.detailId == '0' || this.detailId == 0)){  
    this.details = this.common.getReceipt();  
  }  
}  
  
initValue(){  
  this.detailId = this.common.getDetailId();  
  this.details = {  
    title: "",  
    desc: ""  
  };  
}  
  
deleteIf(){  
  return this.detailId == '0' ? false : true;  
}
```

```
async save(){  
  let loading = await this.common.loading();  
  loading.present();  
  switch(this.detailId){  
    case 0:  
    case "0":  
      this.create(loading);  
      break;  
    default:  
      this.update(loading);  
      break;  
  }  
}
```

# Details

```
async create(loading){
  try{
    const rec = await this.fb.create(this.details);
    console.log(rec);
    if(rec.success){
      loading.dismiss();
      this.common.presentAlert("Success", "TODO details successfully saved");
      this.navCtrl.back();
    } else {
      loading.dismiss();
      this.common.presentAlert("Error", rec.value);
    }
  } catch (err){
    loading.dismiss();
    this.common.presentAlert("Error", err.message);
  }
}
```

# Details

```
async update/loading){
  const update = await this.fb.update(this.details, this.detailId);
  if(update.success){
    loading.dismiss();
    this.common.presentAlert("Success", "TODO details successfully updated");
    this.navCtrl.back();
  } else {
    loading.dismiss();
    this.common.presentAlert("Error", update.value);
  }
  loading.dismiss();
}
```

# Details

```
async delete(){
  this.common.deleteConfirm(async ()=>{
    let loading = await this.common.loading();
    loading.present();
    const update = await this.fb.delete(this.detailId);
    if(update.success){
      loading.dismiss();
      this.common.presentAlert("Success", "TODO details successfully deleted");
      this.navCtrl.back();
    } else {
      loading.dismiss();
      this.common.presentAlert("Error", update.value);
    }
    loading.dismiss();
  }, this.details.title);
}
```