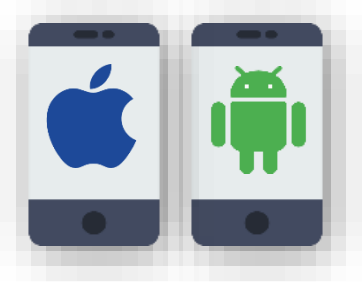
Rapid Fire Mobile 1.0

ARCHITECTURE



Client Access Layer

Business Logic Layer

Data Access Layer

PRESENTATION CORE

Widgets | Style & Theme | Language | Data Operation

Template | Onboarding | Job Scheduling | Messaging/Alert/Bot

Identity | System/OS Service | API | GIS | Global Exception

ILayoutRF
IWidgetStyleRF
IOnboardRF
IThemeDataRF

STORE
Default Implementation

IAlert
IMessage

IModel
Validator Callback

Domain Models & Model State Validation

Database & External Storage Access Provider

DB/MODEL PROVIDER

DB Access Provider

Dynamic Model Mapping

Audit Trail/ Logger

IModelRF

EXTERNAL CONSUMER

Method Channel

External RF Template

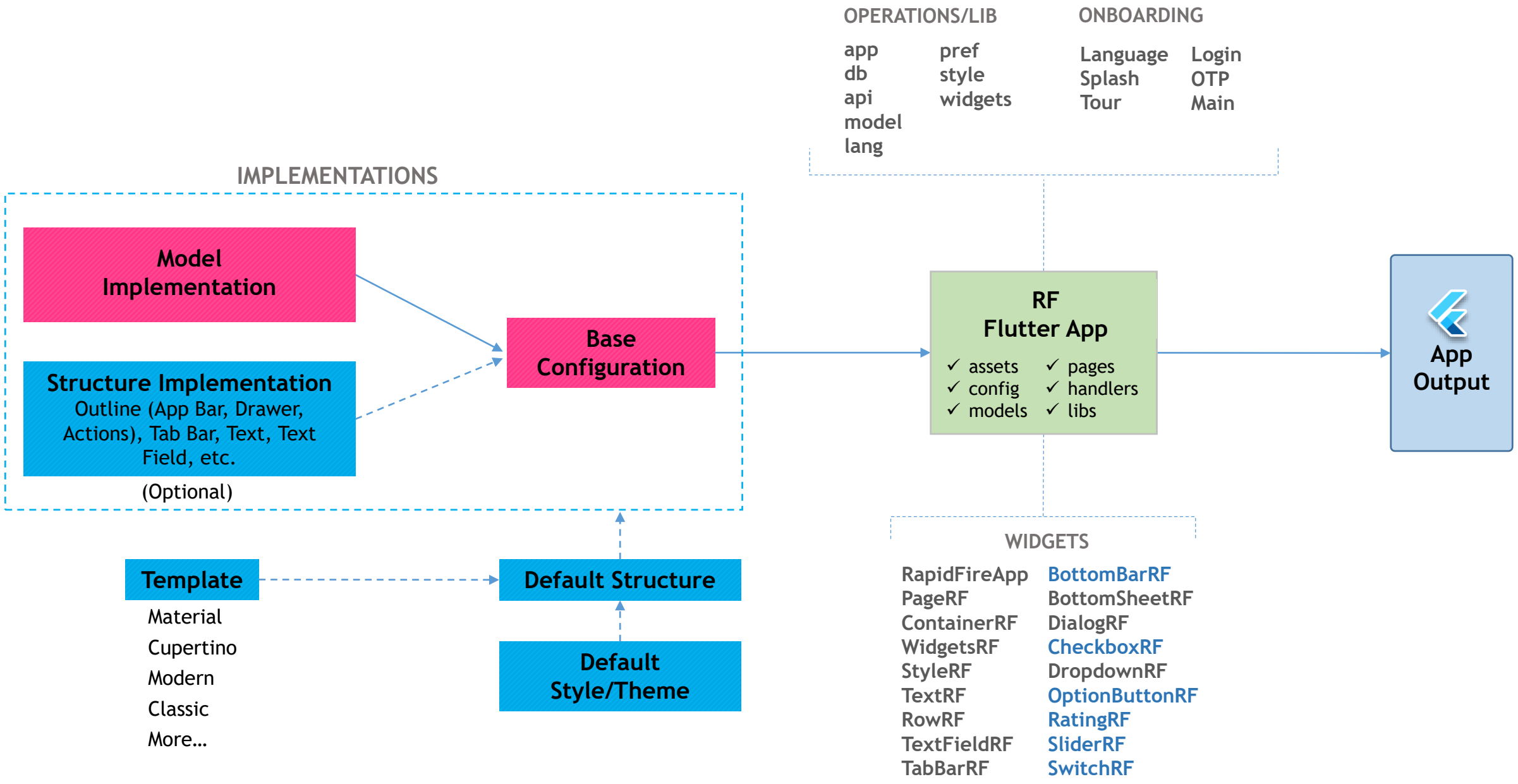
External API Resolver

ILogger
IApiProvider

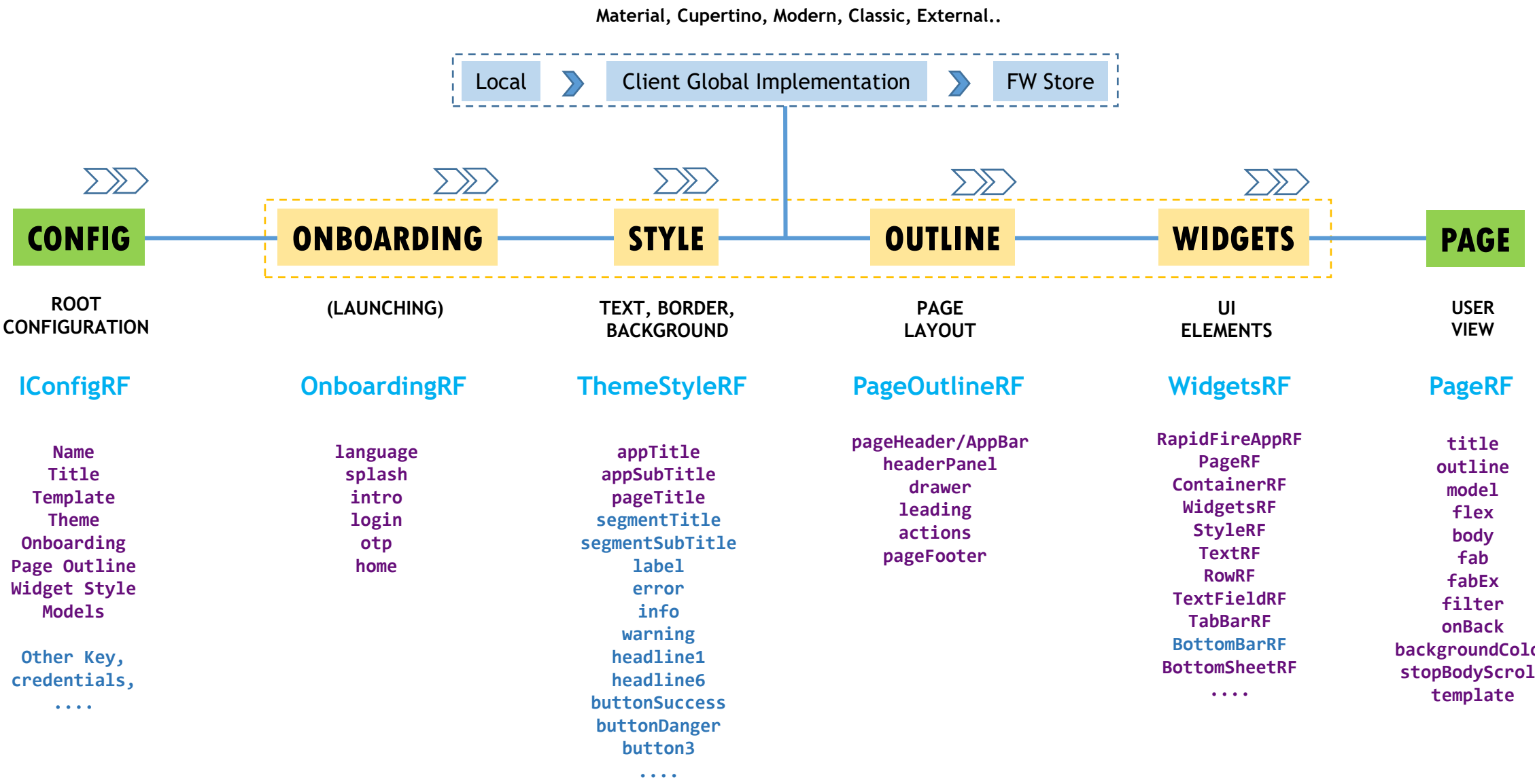
IConfig

external
{json:api}

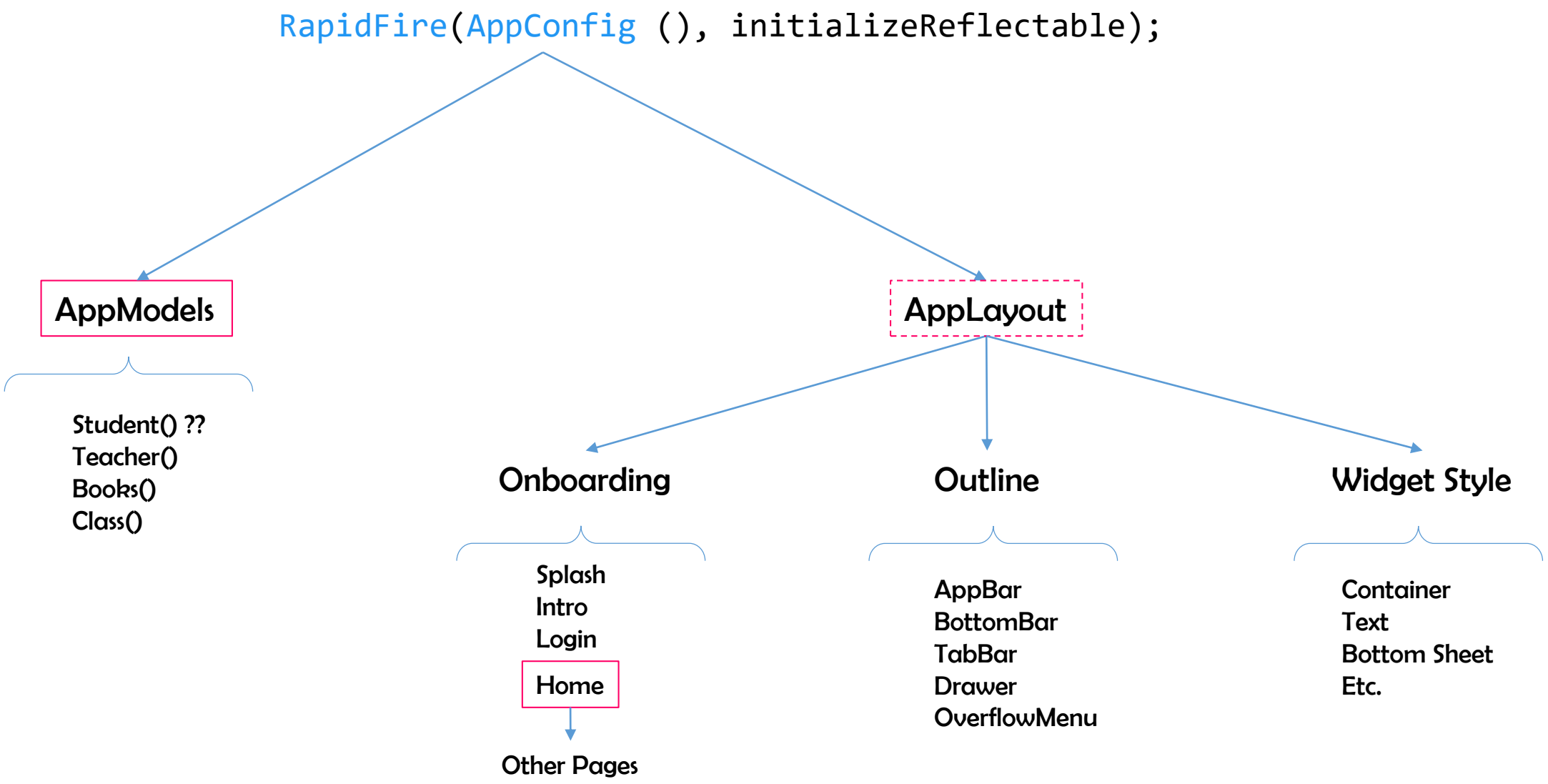
Context & Project Structure



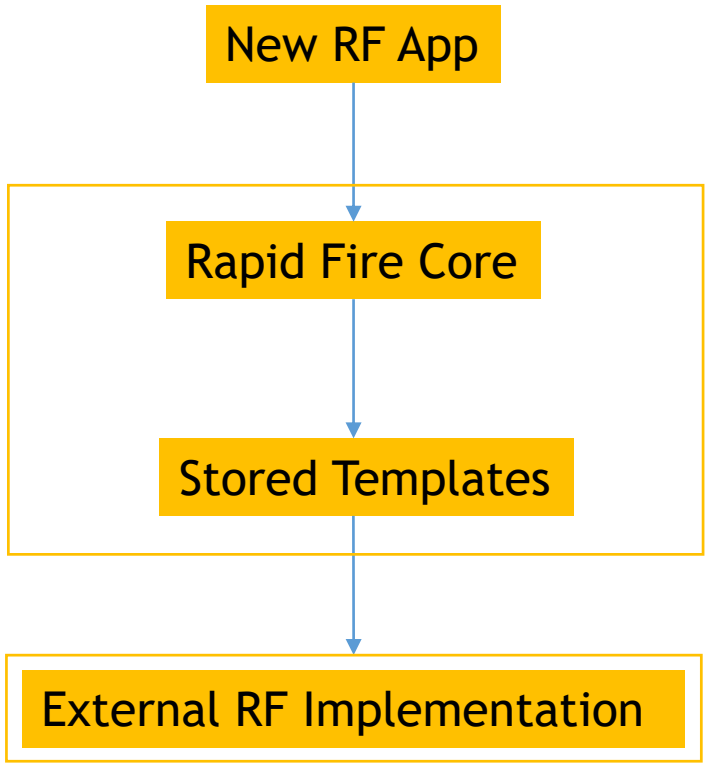
Implementation Journey



Implementation (Client Global)



External Template Implementation



pubspec.yaml

```
dependencies:
  rapidfire: 1.0
  rapidfire-session-meeting: 1.0
  rapidfire-ecom: 1.0
  rapidfire-lms: 1.0
```

```
TemplateRF {
  TemplateNameRF name;
  String version;
  StyleRF style;
  dynamic directives;
}
```

Basic Configuration

Minimum

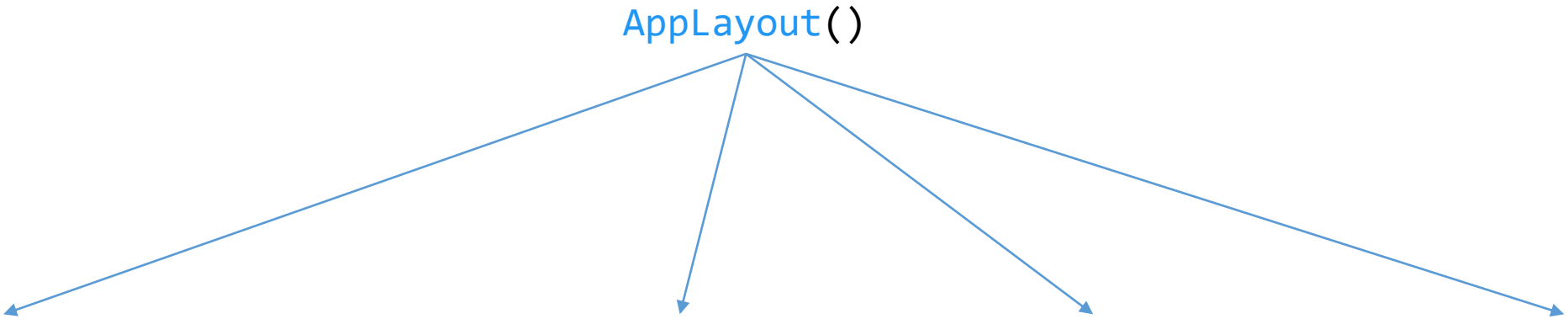
```
void configure(Configuration _configuration) {
    _configuration.app.title = "RapidFire";
    _configuration.app.subTitle = “SubTitle";
    _configuration.model.appModels = AppModels();
}
```

```
class AppModels implements IModelRF {
  @override
  void register(Models models) {
    models.add(
      ModelRF(
        model: Student(),
        fetchDependencies: [Subject],
        displayIcon: null,
        displayName: null,
        fetchIncludeChild: true,
        syncStatusField: 'SyncStatus',
        fetchDependencyCallback: null
      ),
    );
  }
}
```

Maximum

```
void configure(Configuration _configuration) {
    _configuration.model.appModels = AppModels();
    _configuration.model.validators = [ModelValidator.isDataSyncd];

    _configuration.app.title = "RapidFire";
    _configuration.app.subTitle = “Mobile development framework";
    _configuration.app.template = TemplateRF(TemplateNameRF.modern, version: 'Oval');
    _configuration.app.themeData = AppLayout();
    _configuration.app.theme = ThemeRF(
      appTitle: StyleRF(color: Colors.blue, size: 35, weight: FontWeight.bold),
      appSubTitle:
          StyleRF(color: Colors.green, size: 22, weight: FontWeight.bold),
    );
    _configuration.app.pageOutline = AppLayout();
    _configuration.app.widgetStyle = StyleStoreDefaults(everything: AppLayout());
    _configuration.app.onboard = AppLayout();

    _configuration.db.name = "RFDB";
    _configuration.db.virsion = 2;

    _configuration.page.transition = TransitionTypeRF.leftToRight;
    _configuration.page.scrollEffect = ScrollEffectRF.elastic;

    _configuration.data.pageSize = 15;
    _configuration.data.downloadLimit = ‘3 Months’;

    _configuration.messaging.firebase/account = 2123;
    _configuration.messaging.sms/httpClient = 2123;
    _configuration.map.key/account = 2123;

    _configuration.app.IsSAAS = true;

    _configuration.ui.alterStockItems = StockRF(outline: Outline(drawer:InsertRF(item, 1)));
}
```

Extended Configuration

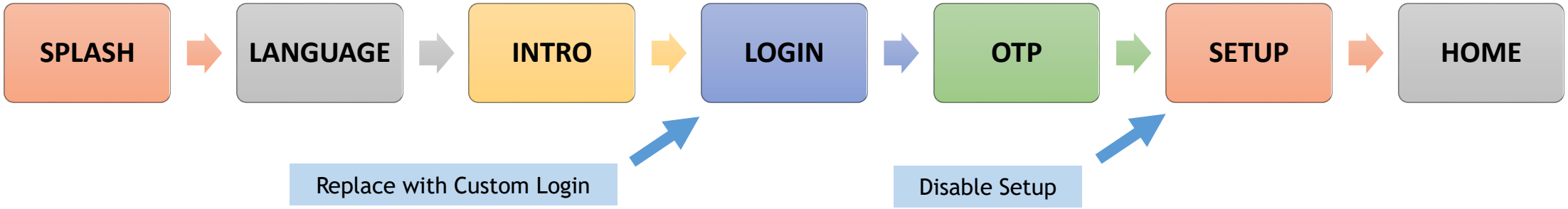



```
AppOnboardRF({
  this.language,
  this.splash,
  @required this.intro,
  this.login,
  this.otp,
  this.setup,
  @required this.home,
});
```

```
PageOutlineRF({
  this.pageHeader,
  this.pageHeaderPanel,
  this.pageFooter,
  this.template,
});
```

```
ThemeData()
```

```
StyleContainerRF({
  this.alignment,
  this.padding,
  this.margin,
  this.height,
  this.decoration,
});
```




```
rf.page.replace(context, rf.onboard.next());
```

Minimum

```
@override
AppOnboardRF customOnboard() {
  return AppOnboardRF(home: Home())
}
```

Maximum

```
@override
AppOnboardRF customOnboard() {
  return AppOnboardRF(
    splash: SplashRF(duration: 10, rippleColor: Colors.blue),
    language: null,
    intro: introItems,
    login: MyLogin(),
    otp: EmptyRF(),
    setup: EmptyRF(),
    home: Home(),
  );
}
```

```
SplashRF({
  title,
  subTitle,
  color,
  font,
  titleColor,
  subTitleColor,
  rippleColor,
  rippleImagePath,
  orgLogoPath,
  duration = 5,
});
```

Standard Page

Leading, Drawer, Title

Page Header

Actions

Header Panel

Profile

Page Title

Segment Title

Label

THEME

General Information

First Name

Last Name

Sex

Male

Female

WidgetsRF

RapidFireAppRF
PageRF
ContainerRF
TextRF
Etc…

Segment

Segment Model

Page Model

Model

Page Footer

PageHeaderRF
Drawer, Leading
Actions, Style, *Title…*

Tab Bar
Static Bar

Body

Page Border

Page Background

Fab, FabEx

Tab Bar
Bottom Navigation Panel
Static Bar

OUTLINE

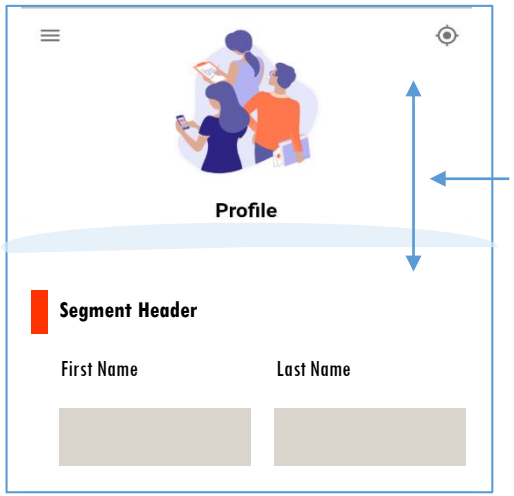
ThemeRF

```
appTitle
appSubTitle
pageTitle
segmentTitle
segmentSubTitle
label
error
info
warning
headline1
headline6
buttonSuccess
buttonDanger
button3
```

PageRF

```
title
outline
model
flex
body
fab
fabEx
filter
onBack
backgroundColor
stopBodyScroll
template
```

Flexible Page



Flexible Bar

Page Class Structure

```
PageRF({
  title,
  outline,
  model,
  flex,
  stickyHeader,
  body,
  fab*,
  fabEx,
  filter*,
  onBack,
  backgroundColor,
  disabledPageScroll = false,
  osBarColors,
  template,
});
```



```
PageOutlineRF({
  pageHeader,
  pageHeaderPanel,
  pageFooter,
});
```



```
PageHeaderRF({
  leading,
  actions,
  drawer,
  elevation,
  backgroundColor,
  foregroundColor,
});
```

tb.moveBefore()

tb.moveAfter()

Step 1

```dart
@override
void initState() {

TabBuilderRF tb = TabBuilderRF(
  tickerProvider: this,
  tabBuilder: (selected) {
    return [
      Tab(child: TabDesignerRF(0, selected, label: 'Options ${1}')),
      Tab(child: TabDesignerRF(1, selected, label: 'Options ${2}')),
      Tab(child: TabDesignerRF(2, selected, label: 'Options ${3}')),
      Tab(child: TabDesignerRF(3, selected, label: 'Options ${4}')),
      Tab(child: TabDesignerRF(4, selected, label: 'Options ${5}'))
    ];
  },
  views: [Text('Option 1'), Text('Option 2'), Text('Option 3'), Text('Option
          4'), Text('Option 5')],
);
```

```dart
TabDesignerRF(
  this.index,
  this.selected, {
  this.label,
  this.icon,
  this.decoration,
});
```

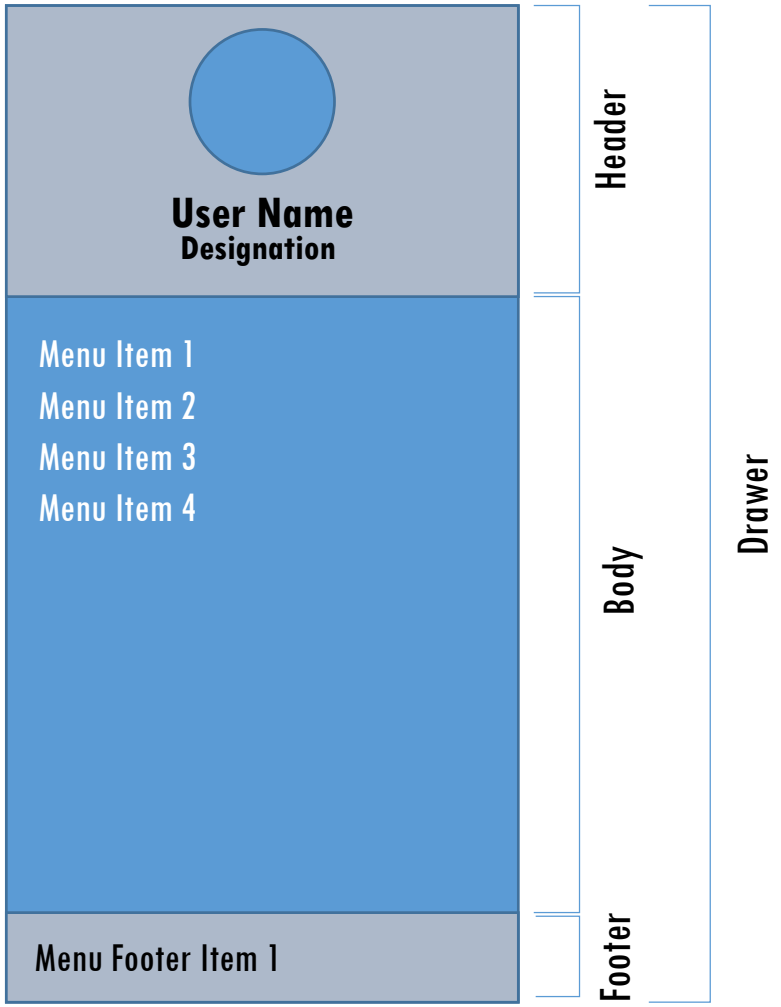
Or

STORE

```dart
StoreRF.scaffold().widgets.tabB
ar.getStyle(widget.index,
widget.selected, label:
widget.label)
```

Step 2

```dart
@override
Widget build(BuildContext context) {
return PageRF(
  title: 'My Page',
  outline: PageOutlineRF(pageHeaderPanel: TabBarRF(tab: tb),
  pageFooter: PageOutlineRF(pageHeaderPanel: TabBarRF(tab: tb),
  body: PageOutlineRF(pageHeaderPanel: TabBarRF(view: tb),
)
```




```
DrawerRF({
  this.menuHeader,
  this.menuBody,
  this.menuFooter,
  this.pageBuilder,
  this.background,
  this.clipper,
  this.headerInfoDefaultType = DrawerHeaderInfoDefaultType.user/app,
  this.template,
});

DrawerHeaderItemsRF()
DrawerItemRF()
```

User Name
Designation

App Title
App Sub Title

Flutter Regular Widgets like Container()

OR

DrawerRF (

Header = null THEN STORE HEADER (T)

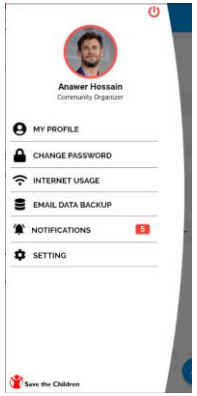
Body = null THEN STORE BODY (T)

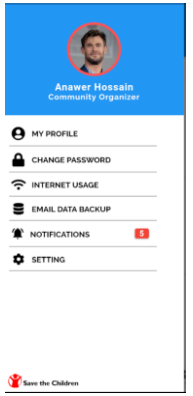
Footer = null THEN STORE FOOTER (T)

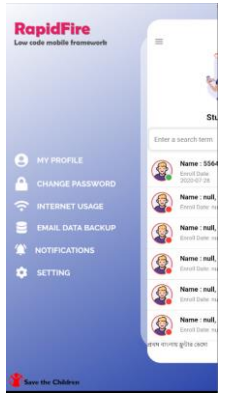
Clipper = null THEN STORE CLIPPER (T)

Template = null THEN STORE DEFAULT TEMPLATE

)





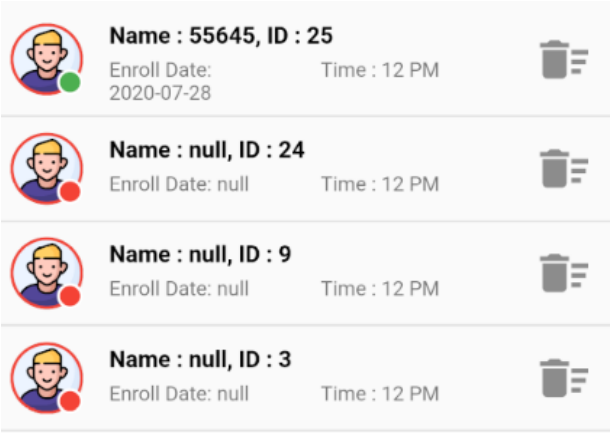


List View and Filters

```
var dataSource = DataSourceRF(itemList: List<Student>(), sql: 'select *');
```

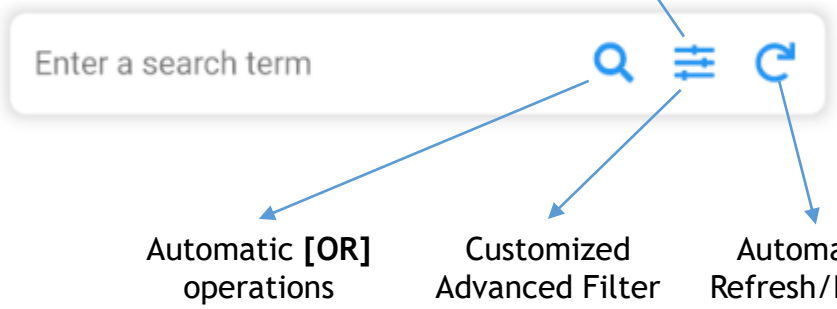
```
ListViewRF(
  dataSource: dataSource,
  builder: (model, index) {
    model = model as Student;
    return CardRF();
  },
  onTap: (model) {
    rf.page.call(context, StudentEntry(model.StudentId), refresh: true);
  },
)
```

```
FilterRF(
  dataSource: dataSource,
  onAdvanceFilter: () {
    DialogRF(
      context,
      title: 'Student Filters',
      body: filterUI(),
      type: DialogType.full/popup/bottom,
    );
  },
)
```



DataSourceRF

1. Auto Local Paging, default 15
2. Auto API Hooking with Query Customization
3. Auto Remote Paging, default 15
4. Automatic Swipe or Tap Delete with Confirmation
5. Quick Customize CardRF for various card tile design



Automatic **[OR]** operations

Customized Advanced Filter

Automatic Refresh/Reset

Text, Text Field & Container

```
TextRF(
  this.text, {
  this.size,
  this.weight,
  this.color,
  this.visible = true,
  this.style,
  this.template,
});
```

```
TextFieldRF({
  this.model,
  this.field,
  this.inputType,
  this.maxLength,
  this.hintText,
  this.hint,
  this.caption,
  this.onSaved,
  this.validator,
  this.onFieldSubmitted,
  this.validator1,
  this.onChange,
});
```

```
ContainerRF({
  this.alignment,
  this.padding,
  this.margin,
  this.height,
  this.decoration,
  this.child,
  this.template,
  this.style,
});
```

Step 1

```
var student = Student(); //declare model
```

Step 2

```
//load model
loadModel(){
  rf.db.get(student, 'select * from Student where StudentId = ?', [widget.studentId], (model) {
    setState(() {
      student = model.first;
    });
  });
}
```

Step 3

```
// design UI (automatic validation & multilingual maping)
return PageRF(
  title: 'SJ - MIS',
  model: student,
  body: ContainerRF(
    child: Column(
      children: <Widget>[
        RowRF([PageTitleRF("Student Profile")]),
        RowRF([PageSubTitleRF("General Information")]),
        RowRF([TextRF("Name")]),
        RowRF([TextFieldRF(field: 'StudentName')]),
        RowRF([TextRF("Enroll Date"), TextRF("Enroll Time")]),
        RowRF([DatePickerRF(field: 'EnrollDate'), TimePickerRF(field: 'EnrollTime')]),
      ],
    ),
  )
)
```

Step 4

```
// automatic add or update
FabRF(
  type: FabType.add,
  onPressed: () async {
    if (PageRF.isModelValid(context)) {
      var result = await rf.db.save(student);
      rf.page.back(context);
    }
  },
)
```

Step 1

```
var student = Student(); //declare model
```

Step 2

```
//fetch data
var data = await api.fetch(student, pageNo: pageNo);

dynamic fetch(
  Object model, {
  String processAlertText,
  dynamic onComplete,
});
```

Step 3

```
// sync data
var result = await api.sync(student);

void sync(
  Object model, {
  String processAlertText,
  dynamic onComplete,
})
```

Step 4

1. automatic periodic sync fetch added in scheduler
2. automatic add sync/fetch item to manual operation list