**Redux Toolkit**

* Redux is a predictable state container for JS app
* Redux core is old School ( backdated ) & toolkit is latest
* Easy to use and handle state & best developer experience
* Remove Huge file setup and management
* State immuting is to hard to manage in Core to solve this
* Switch case complex syntax, toolkit is ignore this
* Third party packages need to use thunk, saga, devtools . . .

**Pre Requirements**

* useStata, useContaxt, usereducer
* Redux core with some project
* React redux

**Redux Install & Setup**

* To install RTK   
  npm i @reduxjs/toolkit react-redux

**Redux Toolkit Structure**

* app   
  - store.js
* features   
  - featureFolder/featireSlice





**Feature Slice**

* State action types
* State action handlers
* Reducers
* Initial States

**Store Setup**

* To setup / configure redux toollit store first create a store file into app folder
* Then config store with combine reducers   
  const store = configureStore({  
   reducer : {  
   counter : counterReducer   
   },   
   Middleware : (getDefaultMiddleware) => {  
   getDefaultMiddleware({  
   // my middlewares   
   });  
   }  
  });
* After creating a store now we need to provide this store to react apps same like redux core

**Create Feature Slice**

* RTK Slices are the arrarangement of type, actions, reducers, initial state
* In a app, every state features are called slice   
    
    
    
  - To create a slice , go to features folder and then create a feature folder , now create a featureSlice   
    
  export const counterSlice = createSlice({  
   name : “type prefix name / slice name”,   
   initialState : 0,   
   reducers : {  
   Increment : (state, action) => {  
   state++;  
   },   
   decrement : (state, action) => {  
   state–;  
   },   
   }  
  });  
    
  export const { Increment, decrement } = counterSlice.actions;   
  export default counterSlice.reducer;

**Action reducer & prepare**

* Every action has reducer and prepare func
* To manage and handle advance state manage

export const counterSlice = createSlice({  
 name : “type prefix name / slice name”,   
 initialState : 0,   
 reducers : {  
 Increment : (state, action) => {  
 state++;  
 },   
 decrement : (state, action) => {  
 state–;  
 },   
 addIncrement : {  
 Reducer : (state, action) => {  
 state++;  
 },   
 Prepare : (inputs) => {  
 return {  
 Payload : {  
   
 }  
 }  
 }  
 }  
 }  
 });

**Create Async Thunk**

* To load any async api call   
  const fetchPost = createAsyncThunk(‘’, async () => {  
    
  });

**Extra Reducers**

* Extra reducer is used to handle extra features   
  export const counterSlice = createSlice({  
   name : “type prefix name / slice name”,   
   initialState : 0,   
   reducers : {  
   Increment : (state, action) => {  
   state++;  
   },   
   decrement : (state, action) => {  
   state–;  
   },   
   },   
   extraReducers(builder){  
   builder.addCase().addCase();  
   }  
  });