```
import { createSlice, createAsyncThunk } from "@reduxjs/toolkit";
import { cloneDeep } from "lodash";
import QLCVService from "services/QLCVService";

//#region  công văn văn bản
export const getCongVanById = createAsyncThunk(
  "congvan/getCongVanById",
  async (id, { rejectWithValue }) => {
    try {
      const response = await QLCVService.getQlCvById(id);
      return response.data;
    } catch (err) {
      return rejectWithValue(err.message || "Error");
    }
  }
);
export const deleteCongVanById = createAsyncThunk(
  "congvan/deleteCongVanById",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      const response = await QLCVService.deleteQlCvById(payload.id);
      if (response.data.isOk) {
        if (onSuccess) onSuccess(response);
        return payload.id;
      } else {
        return rejectWithValue({ message: response.data.description });
      }
    } catch (err) {
      return rejectWithValue(err.message || "Error");
    }
  }
);
export const getCongVanAll = createAsyncThunk(
  "congvan/getCongVanAll",
  async (data, { rejectWithValue }) => {
    try {
      const response = await QLCVService.getAllCongVanDen(data);
      return response;
    } catch (err) {
      return rejectWithValue(err.message || "Error");
    }
  }
);
export const getCongVanAllDi = createAsyncThunk(
  "congvan/getCongVanAllDi",
  async (data, { rejectWithValue }) => {
    try {
      const response = await QLCVService.getAllCongVanDen(data);
      return response;
    } catch (err) {
      return rejectWithValue(err.message || "Error");
    }
  }
);

export const updateCongVan = createAsyncThunk(
  "congvan/updateCongVan",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      const response = await QLCVService.updateCongVanDen(payload);
      if (response.data.isOk) {
        if (onSuccess) onSuccess(response);
        return payload;
      } else {
        return rejectWithValue({ message: response.data.description });
      }
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);

export const insertCongVan = createAsyncThunk(
  "congvan/insertCongVan",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      var response = await QLCVService.insertVanBan(payload);
      if (response.data.isOk) {
        if (onSuccess) onSuccess(response.data.data);
        return response.data.data;
      } else {
        return rejectWithValue({ message: response.data.description });
      }
    } catch (err) {
      return rejectWithValue({ message: err.response.data.description });
    }
  }
);

  export const updateDaXemCongVan = createAsyncThunk(
    "congvan/updateDaXemCongVan",
    async (data, { rejectWithValue }) => {
      try {
        const { onSuccess } = data;
        const payload = cloneDeep(data);
        delete payload.onSuccess;
        const response = await QLCVService.updateDaXemCongVan(payload);
        if(response.data.isOk)
          {
            if (onSuccess) onSuccess(response);
            return payload;
          }
           else {
            return rejectWithValue({message:response.data.description})
           }
      } catch (err) {
        return rejectWithValue({message:err.message});
      }
    }
  );

export const getDmDoUuTien = createAsyncThunk(
  "congvan/dmdouutien",
  async ({ pageNumber, pageSize }) => {
    const response = await QLCVService.getDmDoUuTien(pageNumber, pageSize);
    return response;
  }
);

export const deleteDmDoUuTienById = createAsyncThunk(
  "congvan/deleteDmDoUuTienById",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      await QLCVService.deleteDmDoUuTienById(payload.id).then((response) => {
        if (response.data.isOk) {
          onSuccess(true);
        }
      });

      return data;
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);

export const getDmDoUuTienById = createAsyncThunk(
  "congvan/getDmDoUuTienById",
  async (data, { rejectWithValue }) => {
    try {
      var res = await QLCVService.getDmDoUuTienById(data);
      return res.data;
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);

export const insertDmDoUuTien = createAsyncThunk(
  "congvan/insertDmDoUuTien",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      var response = await QLCVService.insertDmDoUuTien(payload);

      if (response.data.isOk) {
        onSuccess(response);
        return response.data.data;
      } else {
        return rejectWithValue({ message: response.data.description });
      }
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);
export const updateDmDoUuTien = createAsyncThunk(
  "congvan/updateDmDoUuTien",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      const response = await QLCVService.updateDmDoUuTien(data);
      if (onSuccess) onSuccess(response);
      return data;
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);

//#endregion
//#region  dmvanban
export const getDmVanBan = createAsyncThunk(
  "congvan/dmvanban",
  async ({ pageNumber, pageSize, rejectWithValue }) => {
    try {
      const response = await QLCVService.getDmVanBan(pageNumber, pageSize);
      return response;
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);
export const getDmVanBanById = createAsyncThunk(
  "congvan/getDmVanBanById",
  async (data, { rejectWithValue }) => {
    try {
      var res = await QLCVService.getDmVanBanById(data);
      return res.data;
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
```

```
);

export const insertDmVanBan = createAsyncThunk(
  "congvan/insertDmVanBan",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      var response = await QLCVService.insertDmVanBan(payload);
      if (response.data.isOk) {
        onSuccess(response);
        return response.data.data;
      } else {
        return rejectWithValue({ message: response.data.description });
      }
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);
export const updateDmVanBan = createAsyncThunk(
  "congvan/updateDmVanBan",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      const response = await QLCVService.updateDmVanBan(payload);
      if (onSuccess) onSuccess(response);
      return payload;
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);

export const deleteDmVanBanById = createAsyncThunk(
  "congvan/deleteDmVanBanById",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      await QLCVService.deleteDmVanBanById(payload.id).then((response) => {
        if (response.data.isOk) {
          onSuccess(true);
        }
      });
      return data;
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);

//#endregion
//#region  dmhình thức
export const getDmHinhThucGuiCongVan = createAsyncThunk(
  "congvan/dmhinhthucguicongvan",
  async ({ pageNumber, pageSize, rejectWithValue }) => {
    try {
      const response = await QLCVService.getDmHinhThucGuiCongVan({
        pageNumber,
        pageSize,
      });
      return response;
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);
export const deleteDmHinhThucGuiCongVanById = createAsyncThunk(
  "congvan/deleteDmHinhThucGuiCongVanById",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      await QLCVService.deleteDmHinhThucGuiVanBanById(payload.id).then(
        (response) => {
          if (response.data.isOk) {
            onSuccess(true);
          }
        }
      );
      return data;
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);

export const insertDmHinhThucGuiVanBan = createAsyncThunk(
  "congvan/insertDmHinhThucGuiVanBan",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      var response = await QLCVService.insertDmHinhThucGuiVanBan(payload);
      if (response.data.isOk) {
        if (onSuccess) onSuccess(response);
        return response.data.data;
      } else {
        return rejectWithValue({ message: response.data.description });
      }
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);
export const updateDmHinhThucGuiVanBan = createAsyncThunk(
  "congvan/updateDmHinhThucGuiVanBan",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      var response = await QLCVService.updateDmHinhThucGuiVanBan(payload);
      if (onSuccess) onSuccess(response);
      if (response.data.isOk) {
        if (onSuccess) onSuccess(response);
        return payload;
      } else {
        return rejectWithValue({ message: response.data.description });
      }
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);
export const getDmHinhThucGuiVanBanById = createAsyncThunk(
  "congvan/getDmHinhThucGuiVanBanById",
  async (data, { rejectWithValue }) => {
    try {
      const response = await QLCVService.getDmHinhThucGuiVanBanById(data);
      return response.data;
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);

//#endregion
//#region dm noi gửi nhận công văn

export const getDmNoiGuiNhanCongVan = createAsyncThunk(
  "congvan/dmnoiguinhancongvan",
  async ({ pageNumber, pageSize }) => {
    const response = await QLCVService.getDmNoiGuiNhanCongVan(
      pageNumber,
      pageSize
    );
    return response;
  }
);

export const insertDmNoiGuiNhanCongVan = createAsyncThunk(
  "congvan/insertDmNoiGuiNhanCongVan",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      const response = await QLCVService.insertDmNoiGuiNhanCongVan(payload);
      if (onSuccess) onSuccess(response);

      if (response.data.isOk) return response.data.data;
      else {
        return rejectWithValue({ message: response.data.description });
      }
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);

export const updateDmNoiGuiNhanCongVan = createAsyncThunk(
  "congvan/updateDmNoiGuiNhanCongVan",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      const response = await QLCVService.updateDmNoiGuiNhanCongVan(payload);
      if (onSuccess) {
        onSuccess(response);
        return payload;
      }
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);
export const deleteDmNoiGuiNhanVanBannById = createAsyncThunk(
  "congvan/deleteDmNoiGuiNhanVanBannById",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      await QLCVService.deleteDmNoiGuiNhanVanBanById(payload.id).then(
        (response) => {
          if (response.data.isOk) {
            onSuccess(true);
          }
        }
      );
      return data;
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);
export const getDmNoiGuiNhanVanBanById = createAsyncThunk(
  "congvan/getDmNoiGuiNhanVanBanById",
  async (data, { rejectWithValue }) => {
    try {
      var res = await QLCVService.getDmNoiGuiNhanVanBanById(data);
      return res.data;
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);
//#endregion
//#region  dmnoiluutru công văn văn bản

export const getDmNoiLuuTruCongVan = createAsyncThunk(
  "congvan/dmnoiluutrucongvan",
  async ({ pageNumber, pageSize }) => {
    const response = await QLCVService.getDmNoiLuuTruCongVan(
      pageNumber,
      pageSize
    );
    return response;
  }
);

export const updateDmNoiLuuTru = createAsyncThunk(
  "congvan/updateDmNoiLuuTru",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      const response = await QLCVService.updateDmNoiLuuTru(payload);
      if (onSuccess) onSuccess(response);

      return payload;
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);

export const getDmNoiLuuTruById = createAsyncThunk(
  "congvan/getDmNoiLuuTruById",
  async (data, { rejectWithValue }) => {
    try {
      var res = await QLCVService.getDmNoiLuuTruById(data);
      return res.data;
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);

export const insertDmNoiLuuTru = createAsyncThunk(
  "congvan/insertDmNoiLuuTru",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      const response = await QLCVService.insertDmNoiLuuTru(payload);
      if (response.data.isOk) {
        onSuccess(response);
        return response.data.data;
      } else {
        return rejectWithValue({ message: response.data.description });
      }
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);

export const deleteDmNoiLuuTruById = createAsyncThunk(
  "congvan/deleteDmNoiLuuTruById",
  async (data, { rejectWithValue }) => {
    try {
      const { onSuccess } = data;
      const payload = cloneDeep(data);
      delete payload.onSuccess;
      await QLCVService.deleteDmNoiLuuTruById(payload.id).then((response) => {
        if (response.data.isOk) {
          onSuccess(true);
        }
      });
      return data;
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);
export const getAllDmNoiLuuTruByBranchId = createAsyncThunk(
  "congvan/getAllDmNoiLuuTruByBranchId",
  async (data, { rejectWithValue }) => {
    try {
      const response = await QLCVService.getDmNoiLuuTruByBranchId(data);
      return response;
    } catch (err) {
      return rejectWithValue({ message: err.message });
    }
  }
);
export const themdongnoiluutru = createAsyncThunk(
  "congvan/themdongnoiluutru",
  async (data) => {
    return data;
  }
);

//#endregion

const initialState = {
  loading: false,
  congVanDenDeTail: {},
  CongVanList: [],
  error: "",
  dmdouutien: [],
  dmdouutiendetail: {},
  dmvanban: [],
  dmvanbanDeTail: {},
  dmhinhthucguicongvan: [],
  dmhinhthucguicongvanDeTail: {},
  dmnoiguinhancongvan: [],
  dmNoiGuiNhanCongVanDeTail: {},
  dmnoiluutrucongvan: [],
  dmNoiLuuTruDeTail: {},
  pageNumber: 1,
  pageSize: 10,
  totalItems: 0,
  items: [],
  isThem: false,
  dataGiaoViec: [],
};

export const quanLyCongVanSlice = createSlice({
  name: "congvan",
  initialState,
  reducers: {
    showLoading: (state) => {
      state.loading = true;
    },
    setPageNumber: (state, action) => {
      state.pageNumber = action.payload;
    },
    setPageSize: (state, action) => {
      state.pageSize = action.payload;
    },
    clearError: (state) => {
      state.error = null;
    },
    setTabKey: (state, action) => {
      state.items = [];
      state.tabKey = action.payload;
    },
    ResetFielsValue: (state, action) => {
      state.isThem = action.payload;
    },
  },
  extraReducers: (builder) => {
    builder
      .addCase(getCongVanById.pending, (state) => {
        state.loading = true;
      })
      .addCase(getCongVanById.fulfilled, (state, action) => {
        state.loading = false;
        state.congVanDenDeTail = action.payload;
      })
      .addCase(getCongVanById.rejected, (state, action) => {
        state.loading = false;
        state.error = action.error.message;
      })
      .addCase(getCongVanAll.pending, (state) => {
        state.loading = true;
      })
      .addCase(getCongVanAll.fulfilled, (state, action) => {
        state.loading = false;
        state.CongVanList = action.payload.data.data;
        state.totalItems = action.payload.data.totalCount;
      })
      .addCase(getCongVanAll.rejected, (state, action) => {
        state.loading = false;
      })

      .addCase(getCongVanAllDi.pending, (state) => {
        state.loading = true;
      })
      .addCase(getCongVanAllDi.fulfilled, (state, action) => {
        state.loading = false;
        state.congVanDiList = action.payload.items;
        state.totalItems = action.payload.totalItems;
      })
      .addCase(getCongVanAllDi.rejected, (state, action) => {
        state.loading = false;
      })
      .addCase(insertCongVan.pending, (state) => {
        state.loading = true;
      })
      .addCase(insertCongVan.fulfilled, (state, action) => {
        state.loading = false;
        // state.CongVanList.push(action.payload)
      })
      .addCase(insertCongVan.rejected, (state, action) => {
        state.loading = false;
        state.error = action.payload.message;
      })

      .addCase(updateCongVan.pending, (state) => {
        state.loading = true;
      })
      .addCase(updateCongVan.fulfilled, (state, action) => {
        state.loading = false;
      })
      .addCase(updateCongVan.rejected, (state, action) => {
        state.loading = false;
      })
      .addCase(deleteCongVanById.pending, (state) => {
        state.loading = true;
      })
      .addCase(deleteCongVanById.fulfilled, (state, action) => {
        state.loading = false;
      })
      .addCase(deleteCongVanById.rejected, (state, action) => {
        state.loading = false;
      })

      .addCase(updateDaXemCongVan.pending, (state) => {
        state.loading = true;
      })
      .addCase(updateDaXemCongVan.fulfilled, (state, action) => {
        state.loading = false;

      })
      .addCase(updateDaXemCongVan.rejected, (state, action) => {
        state.loading = false;
      })

      .addCase(getDmDoUuTien.pending, (state) => {
        state.loading = true;
      })
      .addCase(getDmDoUuTien.fulfilled, (state, action) => {
        state.loading = false;
        state.dmdouutien = action.payload.items;
        state.totalItems = action.payload.totalItems;
      })
      .addCase(getDmDoUuTien.rejected, (state, action) => {
        state.loading = false;
      })
      .addCase(deleteDmDoUuTienById.pending, (state) => {
        state.loading = true;
      })
      .addCase(deleteDmDoUuTienById.fulfilled, (state, action) => {
        state.loading = false;


      })
      .addCase(deleteDmDoUuTienById.rejected, (state, action) => {
        state.loading = false;
      })
      .addCase(insertDmDoUuTien.pending, (state) => {
        state.loading = true;
      })
      .addCase(insertDmDoUuTien.fulfilled, (state, action) => {
        state.loading = false;

      })
      .addCase(insertDmDoUuTien.rejected, (state, action) => {
        state.loading = false;
        state.error = action.payload.message;
      })
      .addCase(updateDmDoUuTien.pending, (state) => {
        state.loading = true;
      })
      .addCase(updateDmDoUuTien.fulfilled, (state, action) => {
        state.loading = false;

      })
      .addCase(updateDmDoUuTien.rejected, (state, action) => {
        state.loading = false;
      })
      .addCase(getDmDoUuTienById.pending, (state) => {
        state.loading = true;
      })
      .addCase(getDmDoUuTienById.fulfilled, (state, action) => {
        state.loading = false;
        state.dmdouutiendetail = action.payload;
      })
      .addCase(getDmDoUuTienById.rejected, (state, action) => {
        state.loading = false;
      })

      .addCase(getDmVanBan.pending, (state) => {
        state.loading = true;
      })
      .addCase(getDmVanBan.fulfilled, (state, action) => {
        state.loading = false;
        state.dmvanban = action.payload.items;
        state.totalItems = action.payload.totalItems;
      })
      .addCase(getDmVanBan.rejected, (state, action) => {
        state.loading = false;
      })
      .addCase(insertDmVanBan.pending, (state) => {
        state.loading = true;
      })
      .addCase(insertDmVanBan.fulfilled, (state, action) => {
        state.loading = false;

      })
      .addCase(insertDmVanBan.rejected, (state, action) => {
        state.loading = false;
        state.error = action.payload.message;
      })
      .addCase(updateDmVanBan.pending, (state) => {
        state.loading = true;
      })
      .addCase(updateDmVanBan.fulfilled, (state, action) => {
        state.loading = false;

      })
      .addCase(updateDmVanBan.rejected, (state, action) => {
        state.loading = false;
      })
      .addCase(deleteDmVanBanById.pending, (state) => {
        state.loading = true;
      })
      .addCase(deleteDmVanBanById.fulfilled, (state, action) => {
        state.loading = false;

      })
      .addCase(deleteDmVanBanById.rejected, (state, action) => {
        state.loading = false;
      })
      .addCase(getDmVanBanById.pending, (state) => {
        state.loading = true;
      })
      .addCase(getDmVanBanById.fulfilled, (state, action) => {
        state.loading = false;
        state.dmvanbanDeTail = action.payload;
      })
      .addCase(getDmVanBanById.rejected, (state, action) => {
        state.loading = false;
      })

      .addCase(getDmHinhThucGuiCongVan.pending, (state) => {
        state.loading = true;
      })
      .addCase(getDmHinhThucGuiCongVan.fulfilled, (state, action) => {
        state.loading = false;
        state.dmhinhthucguicongvan = action.payload.items;
        state.totalItems = action.payload.totalItems;
      })
      .addCase(getDmHinhThucGuiCongVan.rejected, (state, action) => {
        state.loading = false;
      })
      .addCase(deleteDmHinhThucGuiCongVanById.pending, (state) => {
        state.loading = true;
      })
      .addCase(deleteDmHinhThucGuiCongVanById.fulfilled, (state, action) => {
        state.loading = false;

      })
      .addCase(deleteDmHinhThucGuiCongVanById.rejected, (state, action) => {
        state.loading = false;
      })
      .addCase(insertDmHinhThucGuiVanBan.pending, (state) => {
        state.loading = true;
      })
      .addCase(insertDmHinhThucGuiVanBan.fulfilled, (state, action) => {
        state.loading = false;

      })
      .addCase(insertDmHinhThucGuiVanBan.rejected, (state, action) => {
        state.loading = false;
        state.error = action.payload.message;
      })
      .addCase(updateDmHinhThucGuiVanBan.pending, (state) => {
        state.loading = true;
      })
      .addCase(updateDmHinhThucGuiVanBan.fulfilled, (state, action) => {
        state.loading = false;

      })
      .addCase(updateDmHinhThucGuiVanBan.rejected, (state, action) => {
        state.loading = false;
      })

      .addCase(getDmHinhThucGuiVanBanById.pending, (state) => {
        state.loading = true;
      })
      .addCase(getDmHinhThucGuiVanBanById.fulfilled, (state, action) => {
        state.loading = false;
        state.dmhinhthucguicongvanDeTail = action.payload;
      })
      .addCase(getDmHinhThucGuiVanBanById.rejected, (state, action) => {
        state.loading = false;
      })

      .addCase(getDmNoiGuiNhanCongVan.pending, (state) => {
        state.loading = true;
      })
      .addCase(getDmNoiGuiNhanCongVan.fulfilled, (state, action) => {
        state.loading = false;
        state.dmnoiguinhancongvan = action.payload.items;
        state.totalItems = action.payload.totalItems;
      })
      .addCase(getDmNoiGuiNhanCongVan.rejected, (state, action) => {
        state.loading = false;
      })
      .addCase(updateDmNoiGuiNhanCongVan.pending, (state) => {
        state.loading = true;
      })
      .addCase(updateDmNoiGuiNhanCongVan.fulfilled, (state, action) => {
        state.loading = false;

      })

      .addCase(getDmNoiGuiNhanVanBanById.rejected, (state, action) => {
        state.loading = false;
      })
      .addCase(getDmNoiGuiNhanVanBanById.pending, (state) => {
        state.loading = true;
      })
      .addCase(getDmNoiGuiNhanVanBanById.fulfilled, (state, action) => {
        state.loading = false;
        state.dmNoiGuiNhanCongVanDeTail = action.payload;
      })

      .addCase(updateDmNoiGuiNhanCongVan.rejected, (state, action) => {
        state.loading = false;
        state.error = action.payload;
      })
      .addCase(insertDmNoiGuiNhanCongVan.pending, (state) => {
        state.loading = true;
      })
      .addCase(insertDmNoiGuiNhanCongVan.fulfilled, (state, action) => {
        state.loading = false;

      })
      .addCase(insertDmNoiGuiNhanCongVan.rejected, (state, action) => {
        state.loading = false;
        state.error = action.payload.message;
      })

      .addCase(deleteDmNoiGuiNhanVanBannById.pending, (state) => {
        state.loading = true;
      })
      .addCase(deleteDmNoiGuiNhanVanBannById.fulfilled, (state, action) => {
        state.loading = false;

      })
```

```
    .addCase(deleteDmNoiGuiNhanVanBannById.rejected, (state, action) => {
      state.loading = false;
      state.error = action.payload;
    })

    .addCase(getDmNoiLuuTruCongVan.pending, (state) => {
      state.loading = true;
    })
    .addCase(getDmNoiLuuTruCongVan.fulfilled, (state, action) => {
      state.loading = false;
      state.dmnoiluutrucongvan = action.payload.items;
      state.totalItems = action.payload.totalItems;
    })
    .addCase(getDmNoiLuuTruCongVan.rejected, (state, action) => {
      state.loading = false;
    })

    .addCase(getDmNoiLuuTruById.pending, (state) => {
      state.loading = true;
    })
    .addCase(getDmNoiLuuTruById.fulfilled, (state, action) => {
      state.loading = false;
      state.dmNoiLuuTruDeTail = action.payload;
    })
    .addCase(getDmNoiLuuTruById.rejected, (state, action) => {
      state.loading = false;
    })

    .addCase(insertDmNoiLuuTru.pending, (state) => {
      state.loading = true;
    })
    .addCase(insertDmNoiLuuTru.fulfilled, (state, action) => {
      state.loading = false;

    })
    .addCase(insertDmNoiLuuTru.rejected, (state, action) => {
      state.loading = false;
      state.error = action.payload.message;
    })
    .addCase(updateDmNoiLuuTru.pending, (state) => {
      state.loading = true;
    })
    .addCase(updateDmNoiLuuTru.fulfilled, (state, action) => {
      state.loading = false;

    })
    .addCase(updateDmNoiLuuTru.rejected, (state, action) => {
      state.loading = false;
    })
    .addCase(getAllDmNoiLuuTruByBranchId.pending, (state) => {
      state.loading = true;
    })
    .addCase(getAllDmNoiLuuTruByBranchId.fulfilled, (state, action) => {
      state.loading = false;
      state.dmnoiluutrucongvan = action.payload;
    })
    .addCase(getAllDmNoiLuuTruByBranchId.rejected, (state, action) => {
      state.loading = false;
    })

    .addCase(deleteDmNoiLuuTruById.pending, (state) => {
      state.loading = true;
    })
    .addCase(deleteDmNoiLuuTruById.fulfilled, (state, action) => {
      state.loading = false;

    })
    .addCase(deleteDmNoiLuuTruById.rejected, (state, action) => {
      state.loading = false;
      state.error = action.payload;
    });
  builder.addCase(themdongnoiluutru.fulfilled, (state, action) => {
    state.items = [...state.dmnoiluutrucongvan, action.payload];
  });
  },
});

export const {
  showLoading,
  setPageNumber,
  setPageSize,
  totalItems,
  clearError,
  setTabKey,
  ResetFielsValue,
} = quanLyCongVanSlice.actions;

export default quanLyCongVanSlice.reducer;
```