Multitenant Core Internals

Article Mutlitenancy

- multitenant article: https://medium.com/deviniti-technology-driven-blog/implementing-multitenancy-architecture-spring-boot-jpa-hibernate-flyway-8fb19b312a10

Setup/Config Mutlitenancy

- config
  - config SchemaMultiTenantConnectionProvider: in application.properties `spring.jpa.properties.hibernate.multi_tenant_connection_provider`
  - flyway migration: `FlywayMultitenantConfig.java`

SpringBoot bootstrap

- `HibernateContext`: trigger reading and caching of `hibernate config` from applicationX.properties

```
OkrApplication > OkrApplicationContextInitializer
               > HibernateContext.extractAndSetHibernateConfig(env):
```

- `TenantConfigProvider`: trigger reading and caching (in TenantConfigs.java) of `tenant config` from applicationX.properties

```
TenantConfigProvider.constructor()
```

- flyway migration (for each tenant): migrates db via data from `TenantConfigProvider.getTenantConfigById()`

```
FlywayMultitenantMigrationInitializer
    > migrateFlyway()
    > TenantConfigProvider.getTenantConfigById()
    > get cached TenantConfig (for flyway) from map
```

- [during bootstrap]
  - called `SchemaMultiTenantConnectionProvider.getAnyConnection()`, which is defined in super class `AbstractMultiTenantConnectionProvider`

```
    public Connection getAnyConnection() throws SQLException {
        return this.getAnyConnectionProvider()
```

```
                    .getConnection();
    }

    abstract ConnectionProvider getAnyConnectionProvider();
```

- concrete implementation in `SchemaMultiTenantConnectionProvider`

```
protected ConnectionProvider getAnyConnectionProvider() {
    return getConnectionProvider("public");
}
```

- int `getConnectionProvider()`: ConnectionProvider is not cached, so create a new one
  - get non-tenant specifiy hibernate configuraiton via `HibernateContext.getHibernateConfig()` (internally cached)
  - use `SchemaMultiTenantConnectionProvider.initConnectionProvider(properties)` to convert properties to ConnectionProvider
  - cache and return ConnectionProvider

login client

- `SchemaMultiTenantConnectionProvider.getConnection(tenantId)` -> HibernateContext.getHibernateConfig(tenantId)

```
SchemaMultiTenantConnectionProvider
    .getConnection(tenantId)
    .[via super class] selectConnectionProvider(tenantId)
    .createNewConnectionProvider(tenantId)
    .getHibernateProperties(tenantId)
    .HibernateContext.getHibernateConfig(tenantId);
```

- `HibernateContext.getHibernateConfig(tenantId)`: return patched properties and convert to ConnectionProvider using `initConnectionProvider(properties`
- cache and return ConnectionProvider