SWINBURNE UNIVERSITY OF TECHNOLOGY

COS20031

Computing Technology Design Project

Week 10-11: Major-specific topics

Software Development

System Development Life Cycles

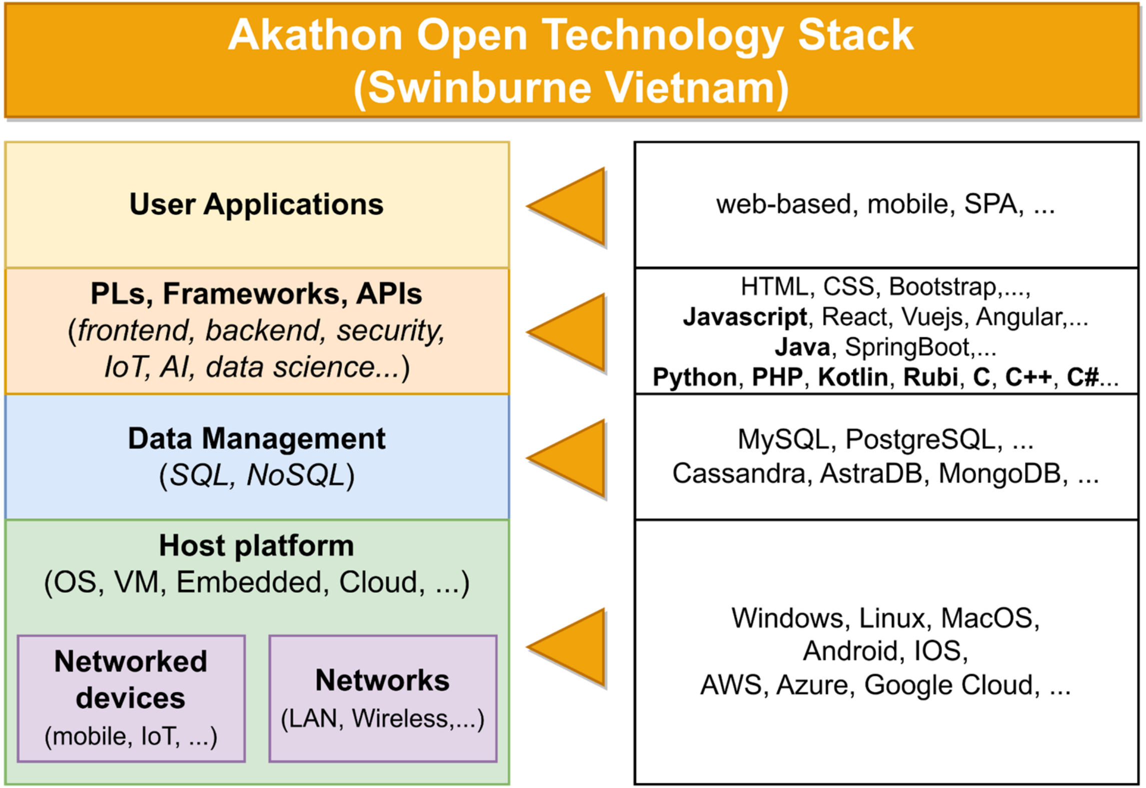
Database

1. Planning
2. Requirement gathering
3. Conceptual design
4. Logical design
5. Physical design
6. **Construction**
7. **Implementation** & rollout
8. Ongoing support

Software

1. Planning
2. Analysis
3. Design
   a. Conceptual
   b. Logical
   c. Physical
4. **Implementation**
5. Deployment
6. Maintenance

Software Development Technology Stack



Akathon Open Technology Stack
(Swinburne Vietnam)

User Applications

web-based, mobile, SPA, ...

PLs, Frameworks, APIs
(*frontend, backend, security, IoT, AI, data science...*)

HTML, CSS, Bootstrap,...,
Javascript, React, Vuejs, Angular,...
Java, SpringBoot,...
Python, **PHP**, **Kotlin**, **Rubi**, **C**, **C++**, **C#**...

Data Management
(*SQL, NoSQL*)

MySQL, PostgreSQL, ...
Cassandra, AstraDB, MongoDB, ...

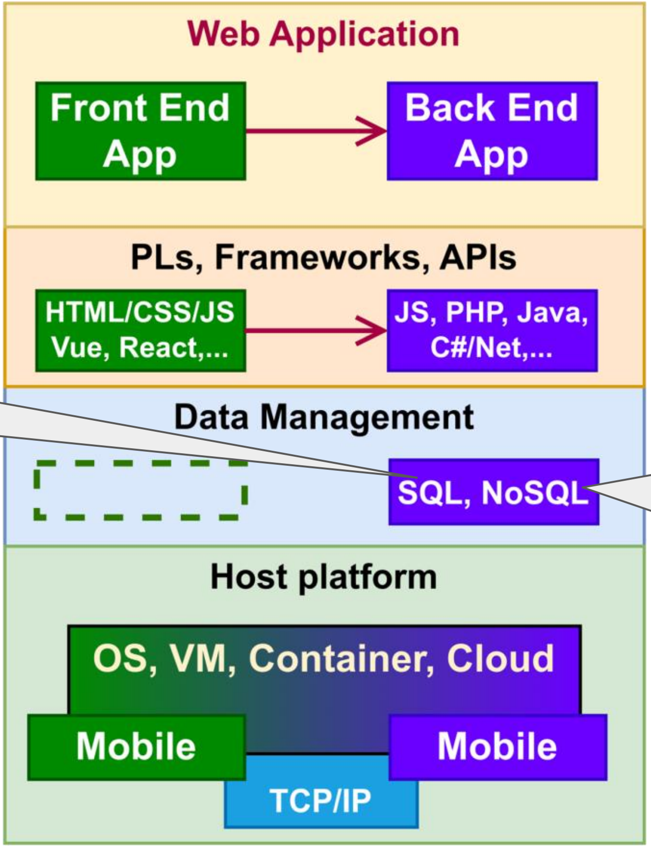
Host platform
(OS, VM, Embedded, Cloud, ...)

Networked devices
(mobile, IoT, ...)

Networks
(LAN, Wireless,...)

Windows, Linux, MacOS,
Android, IOS,
AWS, Azure, Google Cloud, ...

Web-based software: front end & back end



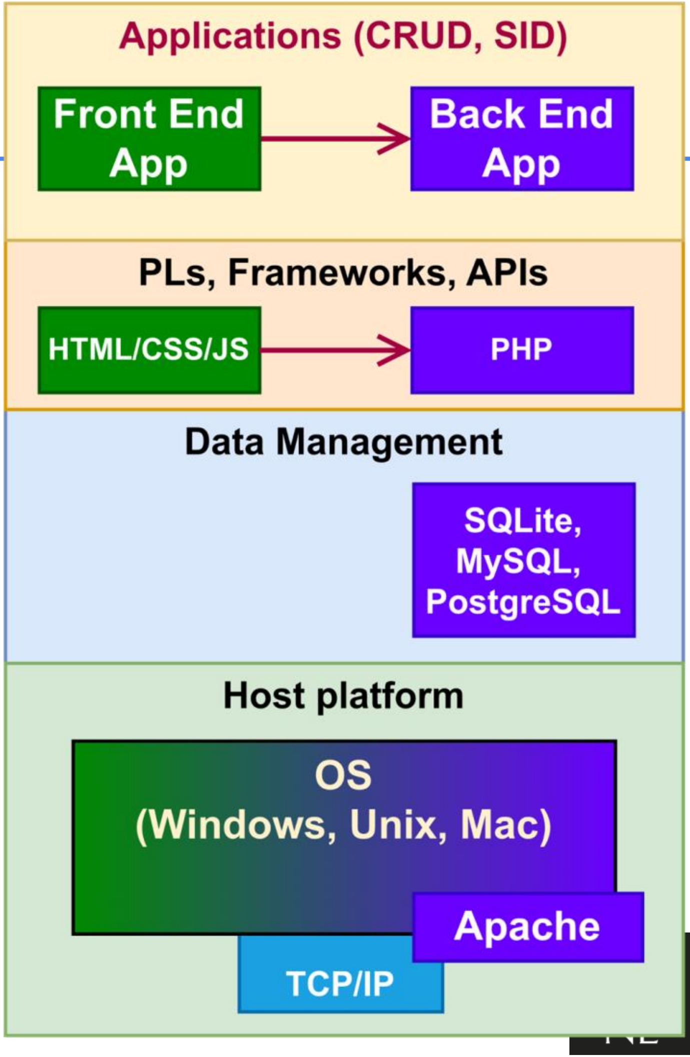
COS20031

Cassandra/
AstraDB
Workshop
(10/11/2023)

(D) CRUD applications

App: CRUD (Album Manager)

- URL: Embedding SQL
- Code set: in the Exercise File
- Language: PHP
- DBMS: SQLite
- Deploy and Demo: on XAMPP



Applications (CRUD, SID)

Front End App → Back End App

PLs, Frameworks, APIs

HTML/CSS/JS → PHP

Data Management

SQLite, MySQL, PostgreSQL

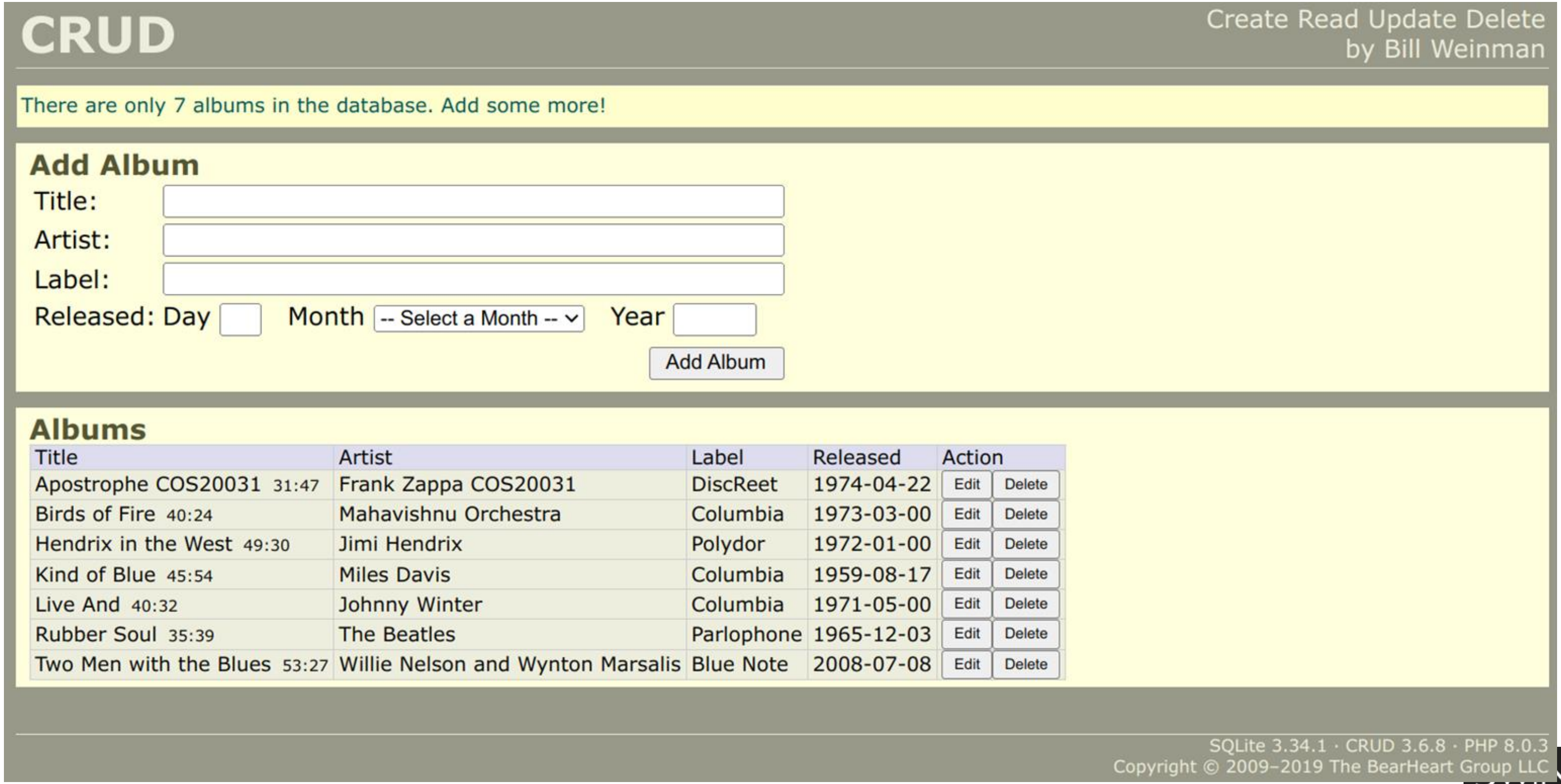
Host platform

OS (Windows, Unix, Mac)

Apache

TCP/IP

App: CRUD (Album Manager)



CRUD

Create Read Update Delete
by Bill Weinman

There are only 7 albums in the database. Add some more!

Add Album

Title:

Artist:

Label:

Released: Day Month -- Select a Month -- Year

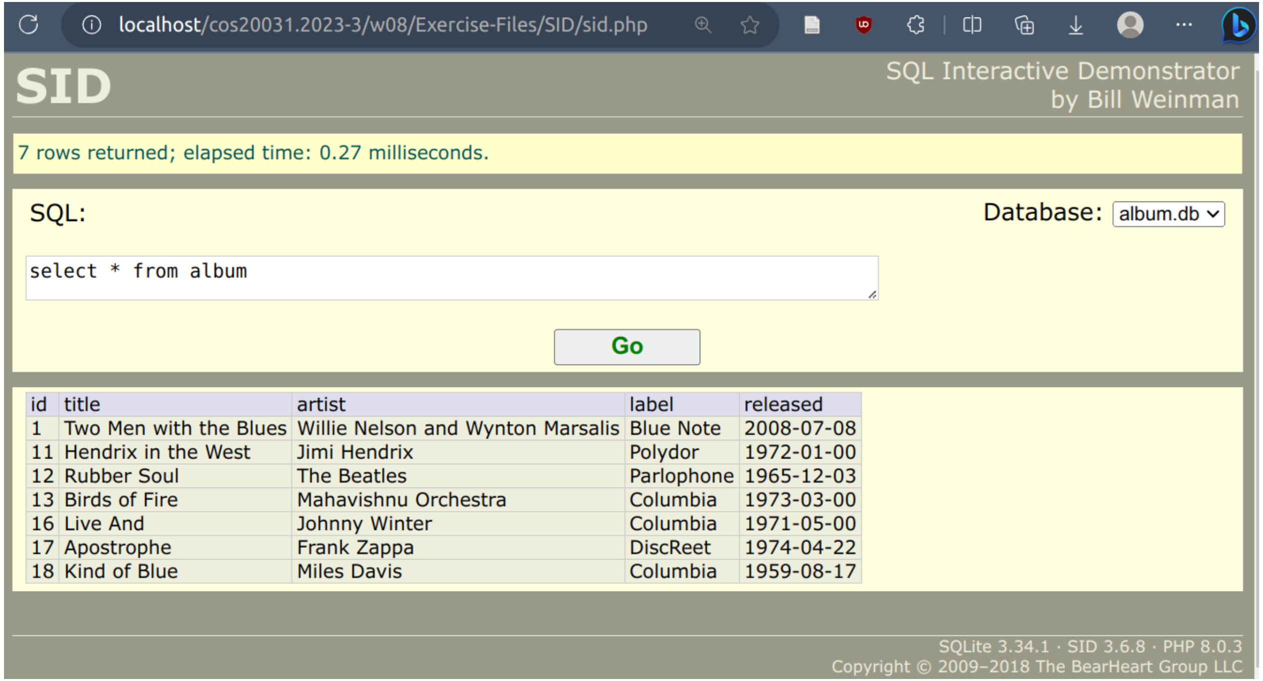
Albums

Title	Artist	Label	Released	Action
Apostrophe COS20031 31:47	Frank Zappa COS20031	DiscReet	1974-04-22	Edit Delete
Birds of Fire 40:24	Mahavishnu Orchestra	Columbia	1973-03-00	Edit Delete
Hendrix in the West 49:30	Jimi Hendrix	Polydor	1972-01-00	Edit Delete
Kind of Blue 45:54	Miles Davis	Columbia	1959-08-17	Edit Delete
Live And 40:32	Johnny Winter	Columbia	1971-05-00	Edit Delete
Rubber Soul 35:39	The Beatles	Parlophone	1965-12-03	Edit Delete
Two Men with the Blues 53:27	Willie Nelson and Wynton Marsalis	Blue Note	2008-07-08	Edit Delete

SQLite 3.34.1 · CRUD 3.6.8 · PHP 8.0.3
Copyright © 2009–2019 The BearHeart Group LLC

App: SQL Interactive Demonstrator (SID)

- URL: Embedding SQL
- Code set: in the Exercise File of the course
- Language: PHP
- DBMS: SQLite
- Deploy and Demo: on XAMPP



localhost/cos20031.2023-3/w08/Exercise-Files/SID/sid.php
SID
SQL Interactive Demonstrator
by Bill Weinman

7 rows returned; elapsed time: 0.27 milliseconds.

SQL:

Database: album.db

```
select * from album
```

id	title	artist	label	released
1	Two Men with the Blues	Willie Nelson and Wynton Marsalis	Blue Note	2008-07-08
11	Hendrix in the West	Jimi Hendrix	Polydor	1972-01-00
12	Rubber Soul	The Beatles	Parlophone	1965-12-03
13	Birds of Fire	Mahavishnu Orchestra	Columbia	1973-03-00
16	Live And	Johnny Winter	Columbia	1971-05-00
17	Apostrophe	Frank Zappa	DiscReet	1974-04-22
18	Kind of Blue	Miles Davis	Columbia	1959-08-17

SQLite 3.34.1 · SID 3.6.8 · PHP 8.0.3
Copyright © 2009–2018 The BearHeart Group LLC

Set up the app

- Install XAMPP to a folder, denote by $XAMPP:
  - e.g. `$XAMPP = c:\Program files\xampp`
- Start the Apache web server
- Copy the app source code folder (e.g. `CRUD`), from the Exercise folder to `$XAMPP/htdocs`
- Access the app on the browser:
  - e.g. `http://localhost/CRUD/crud.php`

How do you use MySQL?

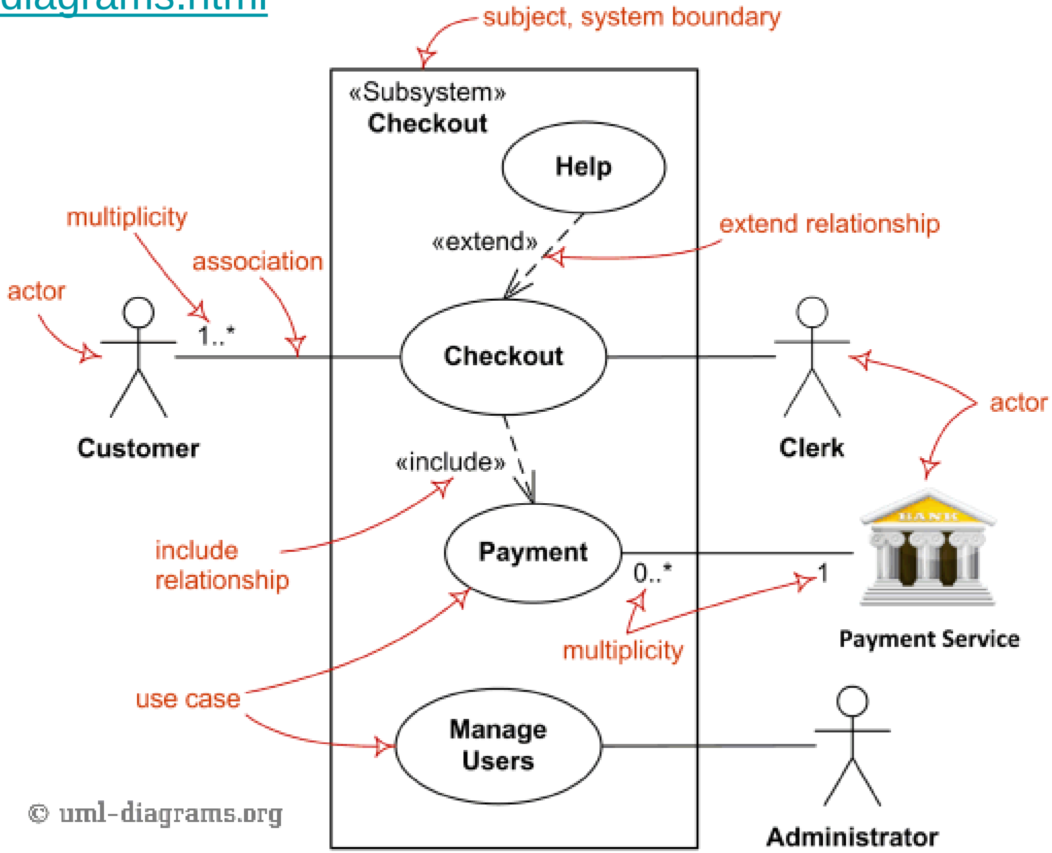
- From XAMPP, start the MySQL server
- Read the header of the the main application script
  - `crud.php`
  - `sid.php`
- Create the MySQL user account and database specified in the header (e.g. album)
- Populate the database with initial data:
  - use the corresponding SQL script provided in the `SQL` subfolder
- Access the app on the browser:
  - e.g. `http://localhost/CRUD/crud.php`

Analysis: Use Case Diagram

https://www.uml-diagrams.org/use-case-diagrams.html

- Model the **behaviours** of a system:
  - business system
  - computing system
- **Use case** is either:
  - a business function OR
  - a system function

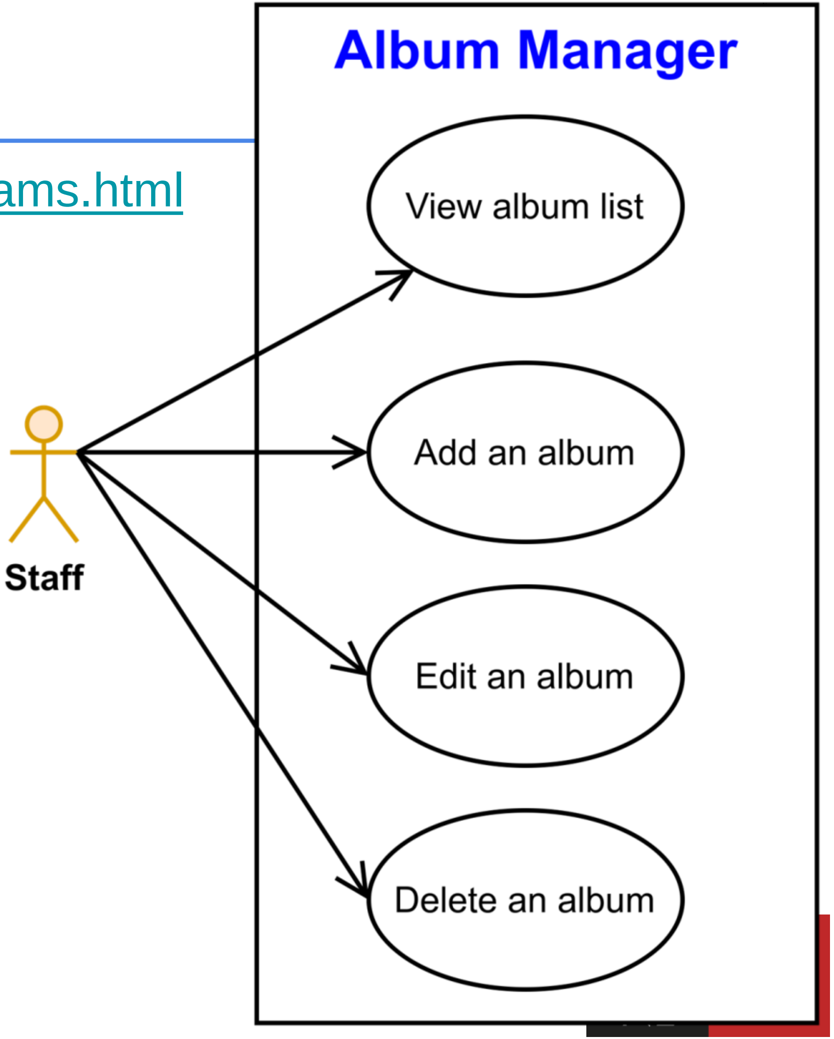
  provided for an interaction by an **actor**



Example: Album Manager

https://www.uml-diagrams.org/use-case-diagrams.html

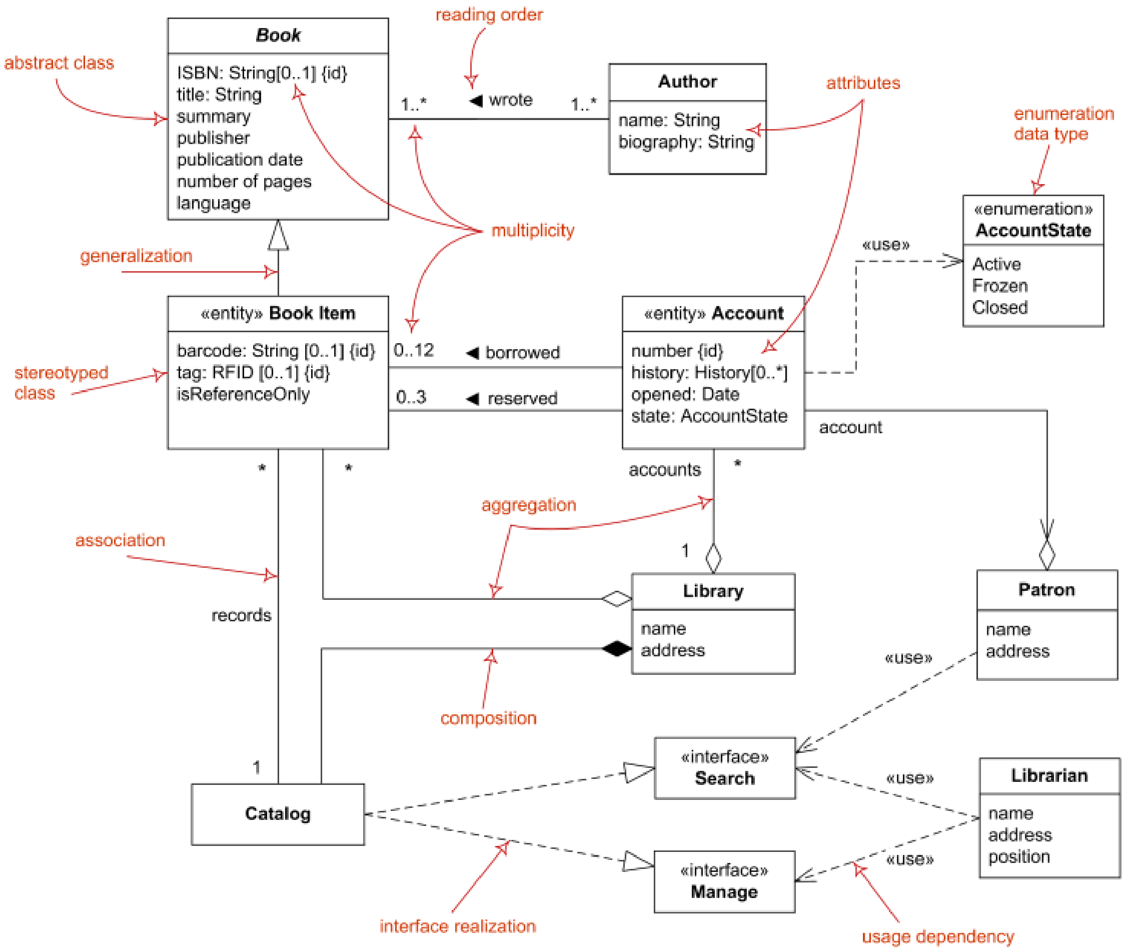
- **Actor**: (Sales) staff
- **System functions**:
  - view album list
  - add, edit, delete an album
- Other functions:
  - what are they?



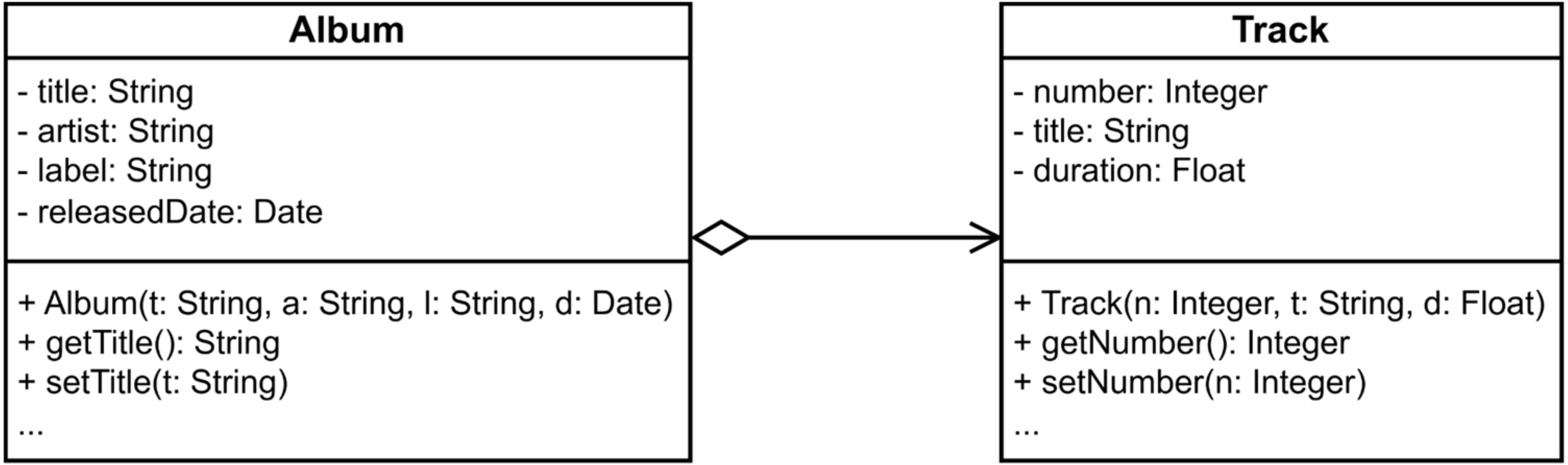
Analysis/Design: Class Diagram

https://www.uml-diagrams.org/class-diagrams-overview.html

- Model the system structure
  - analysis & design
- Elements:
  - class
  - interface
  - property (attribute)
  - operation
  - association
  - constraint



Example: Class Diagram



Album

- title: String
- artist: String
- label: String
- releasedDate: Date

+ Album(t: String, a: String, l: String, d: Date)
+ getTitle(): String
+ setTitle(t: String)
...

Track

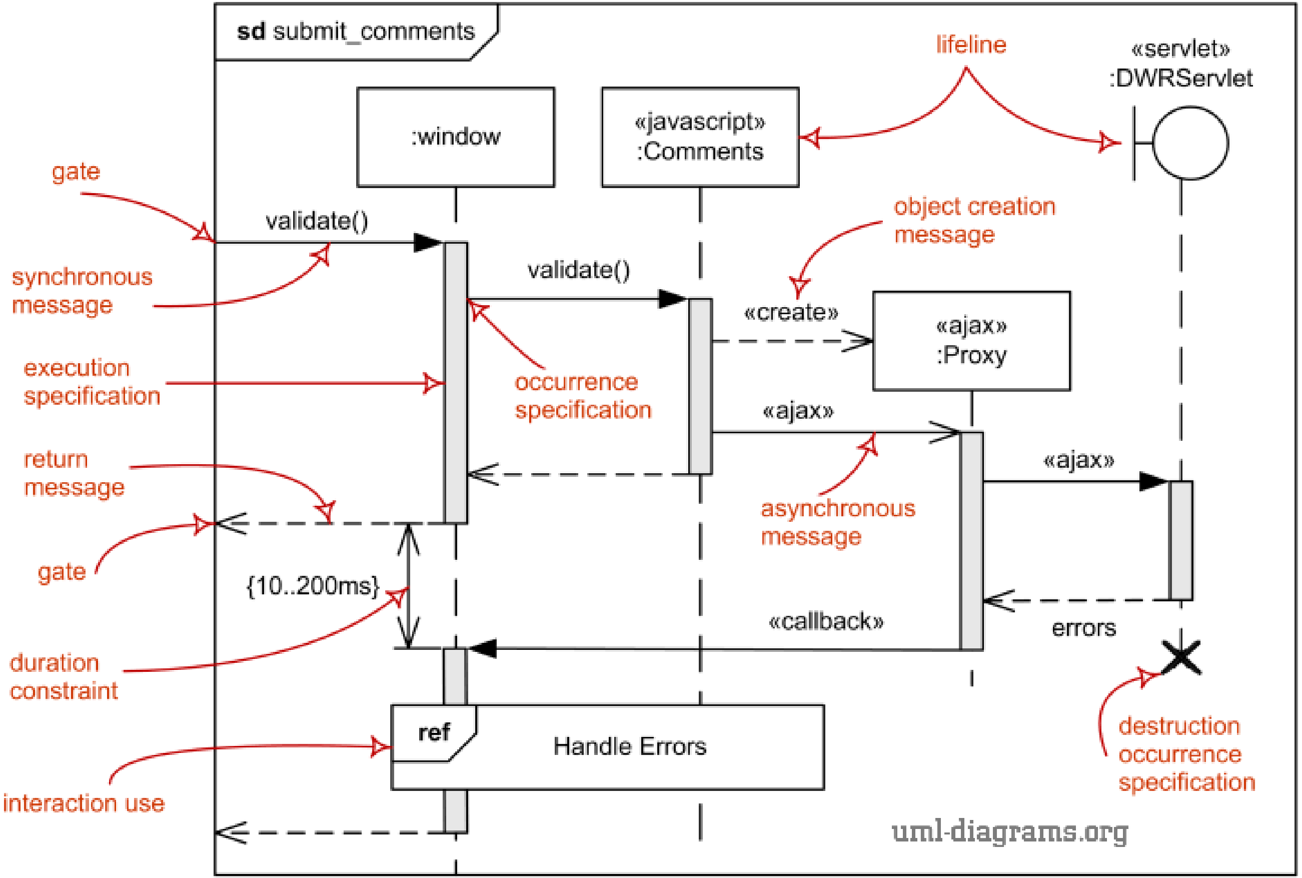
- number: Integer
- title: String
- duration: Float

+ Track(n: Integer, t: String, d: Float)
+ getNumber(): Integer
+ setNumber(n: Integer)
...

Design: Sequence Diagram

https://www.uml-diagrams.org/sequence-diagrams.html

- Describes object interactions in detail
- Key elements:
  - object lifelines
  - messages



Example: Sequence Diagram

Domain Driven Software Development

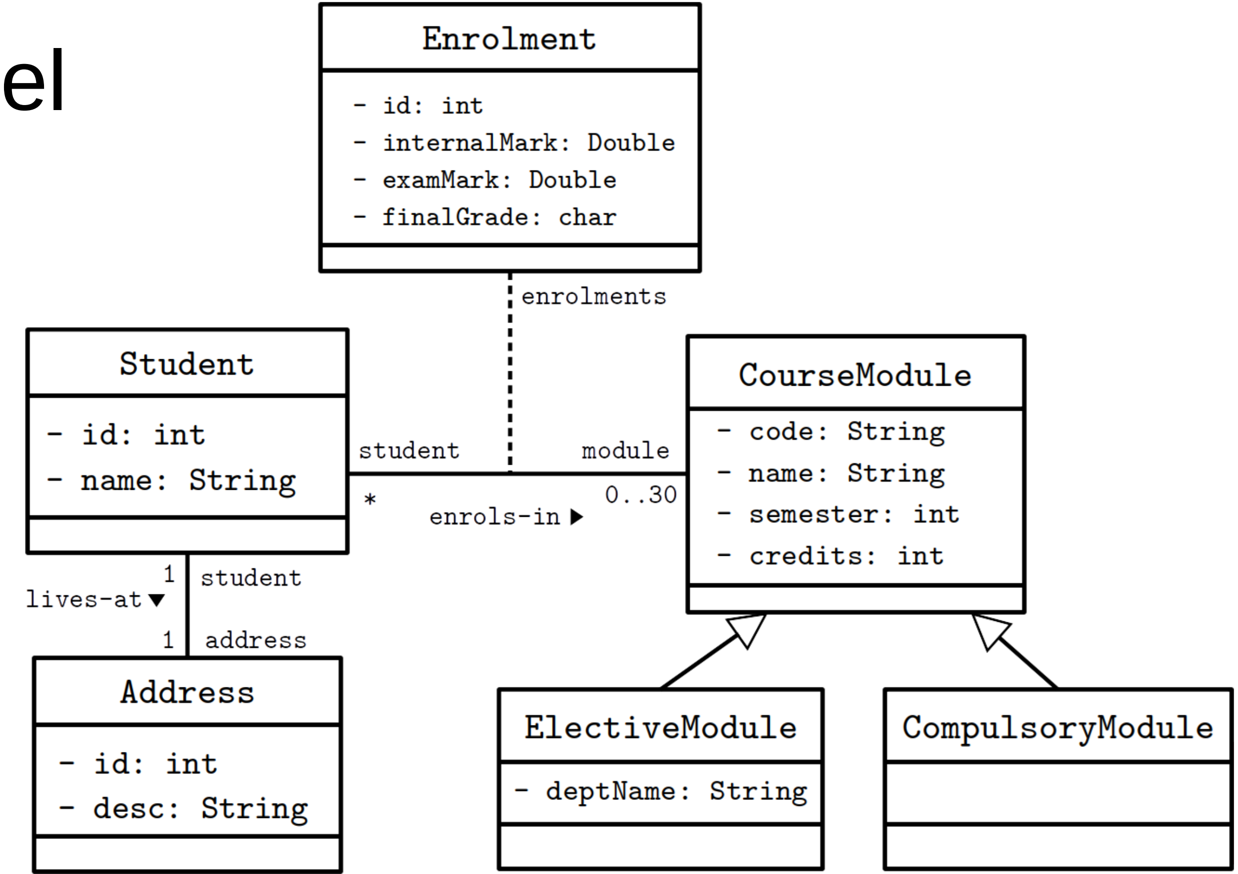
- Domain Driven Design (Eric Evans, 2003)
- Domain model is the core (“heart”) of the software
  - the business logic
  - designed with domain experts
  - implementation feasible
- Software (UI, database, etc.) is designed around the domain model

JDA Software framework

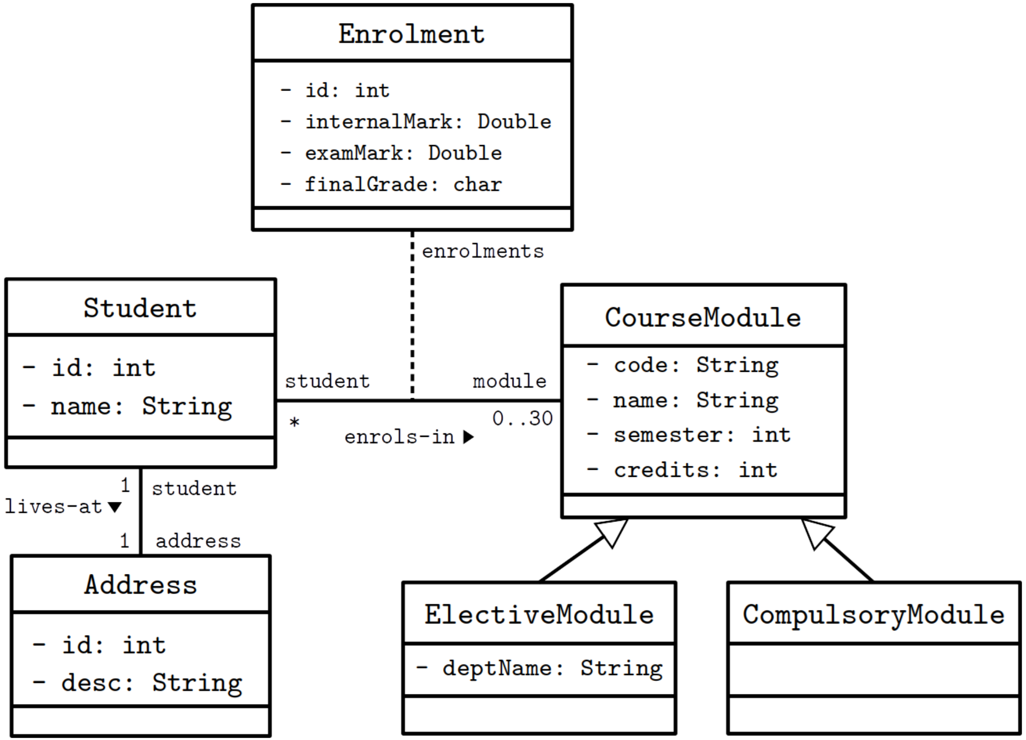
- **JDA: Java-based software framework**
  - https://github.com/jdomainapp/jda
- Features:
  - **Domain Driven Design (DDD)**
  - Java (>= 8)
  - **Frontend**:
    - web frameworks (Vuejs, React, Angular)
    - Mobile: ReactNative
  - **Backend**: SpringBoot (Nodejs: todo)
  - Automatic **software generation** from domain model
  - Maven: multi-project configuration
    - core, common, modules
  - Open source
- **On-going student projects**

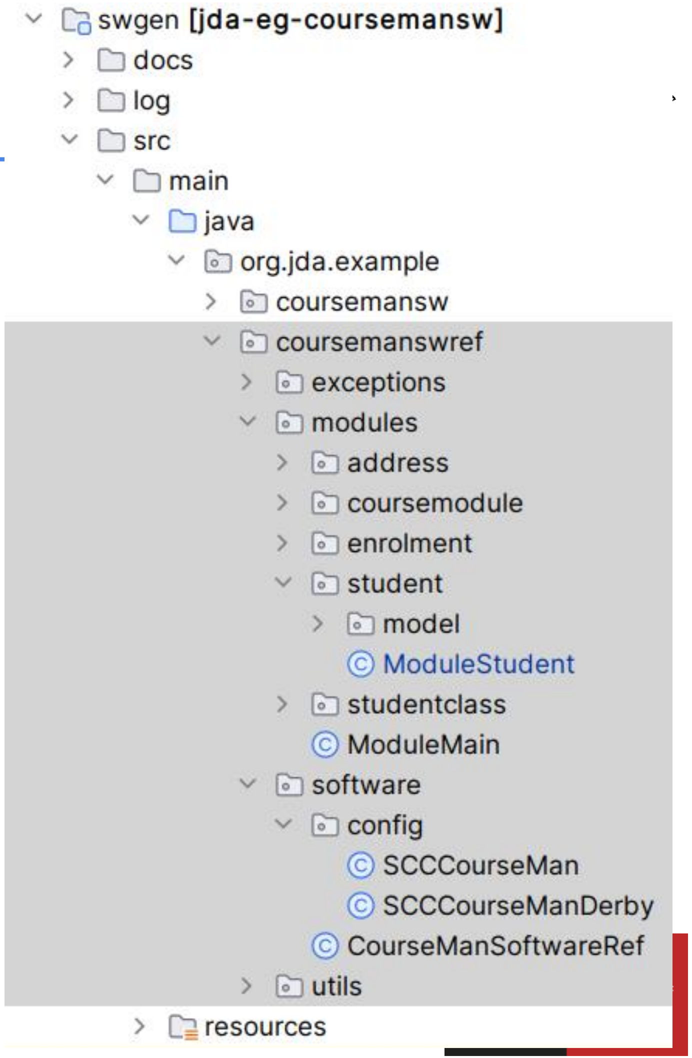
Example: CourseMan

- Domain model



Example: Software design

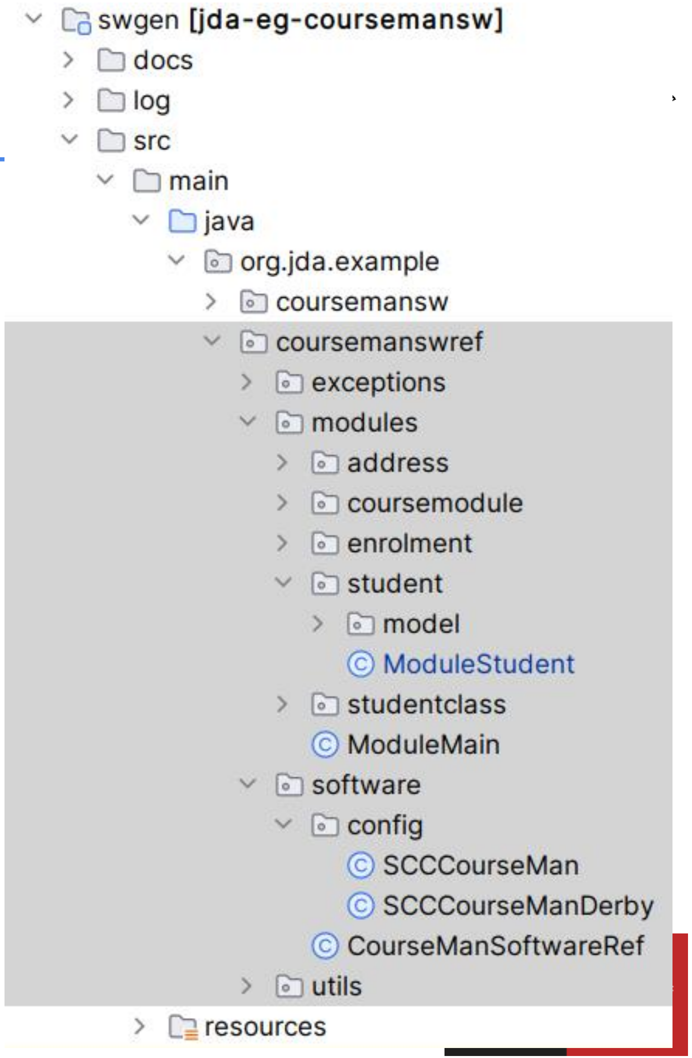




```
swgen [jda-eg-coursemansw]
  docs
  log
  src
    main
      java
        org.jda.example
          coursemansw
          coursemanswref
            exceptions
            modules
              address
              coursemodule
              enrolment
              student
                model
                ModuleStudent
              studentclass
              ModuleMain
            software
              config
                SCCCourseMan
                SCCCourseManDerby
              CourseManSoftwareRef
            utils
      resources
```

Example: Software design

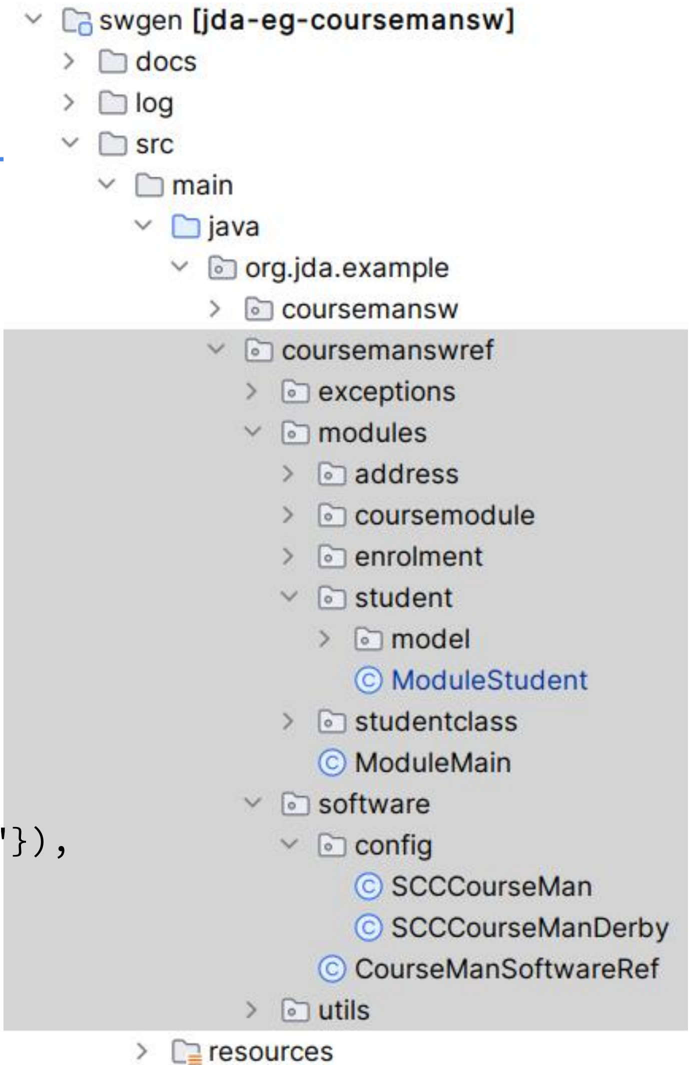
```
@ModuleDescriptor(name = "ModuleStudent",
        modelDesc = @ModelDesc(
                model = Student.class),
        viewDesc = @ViewDesc(
            formTitle = "Form: Student",
            imageIcon = "Student.png",
            domainClassLabel = "Student",
            viewType= RegionType.Data,
            parent= RegionName.Tools,
            view = View.class),
        controllerDesc = @ControllerDesc(
            controller= Controller.class,
            isDataFieldStateListener = true)
    ,isPrimary=true
    ,setUpDesc = @SetUpDesc(postSetUp =
CopyResourceFilesCommand.class)
)
public class ModuleStudent { ... }
```



```
swgen [jda-eg-coursemansw]
  docs
  log
  src
    main
      java
        org.jda.example
          coursemansw
          coursemanswref
            exceptions
            modules
              address
              coursemodule
              enrolment
              student
                model
                ModuleStudent
              studentclass
              ModuleMain
            software
              config
                SCCCourseMan
                SCCCourseManDerby
              CourseManSoftwareRef
            utils
      resources
```

Example: Software design

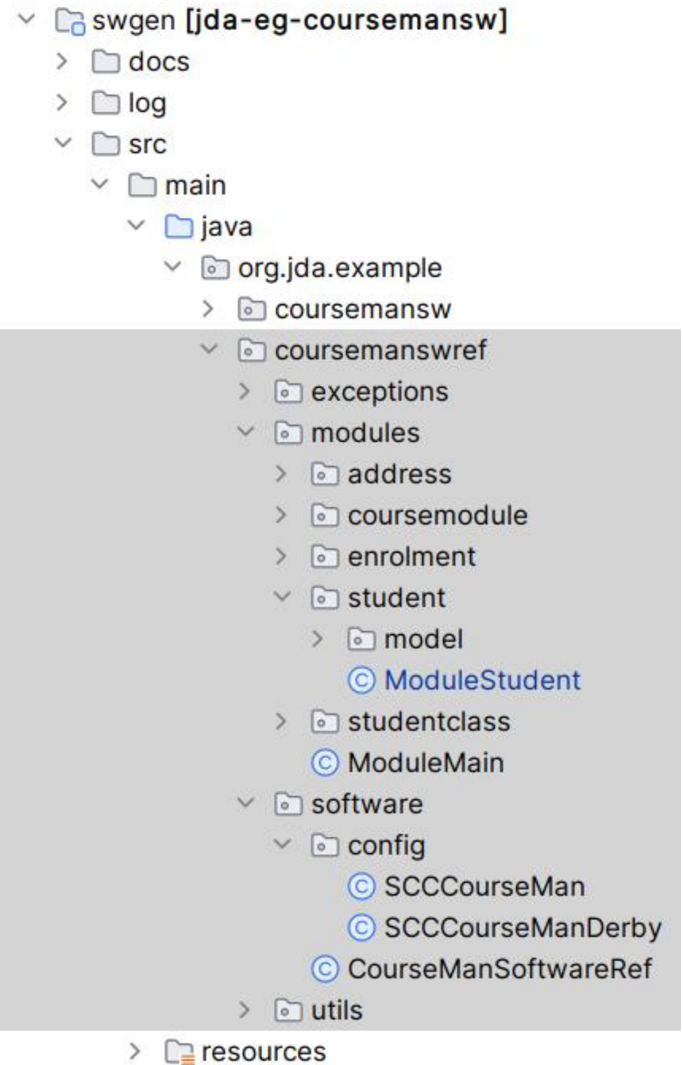
```
@ModuleDescriptor(...)
public class ModuleStudent {
  @AttributeDesc(label = "Manage Students")
  private String title;

  @AttributeDesc(label = "Student ID")
  private int id;

  @AttributeDesc(label = "Full Name")
  private String name;

  ...

  @AttributeDesc(label = "Current Address",
      type = JComboField.class,
      ref=@Select(clazz= Address.class,attributes={"name"}),
      loadOidWithBoundValue=true,
      ,isStateEventSource=true)
  private Address address;
}
```



```
swgen [jda-eg-coursemansw]
  docs
  log
  src
    main
      java
        org.jda.example
          coursemansw
          coursemanswref
            exceptions
            modules
              address
              coursemodule
              enrolment
              student
                model
                ModuleStudent
              studentclass
              ModuleMain
            software
              config
                SCCCourseMan
                SCCCourseManDerby
              CourseManSoftwareRef
            utils
      resources
```

Example: Software design

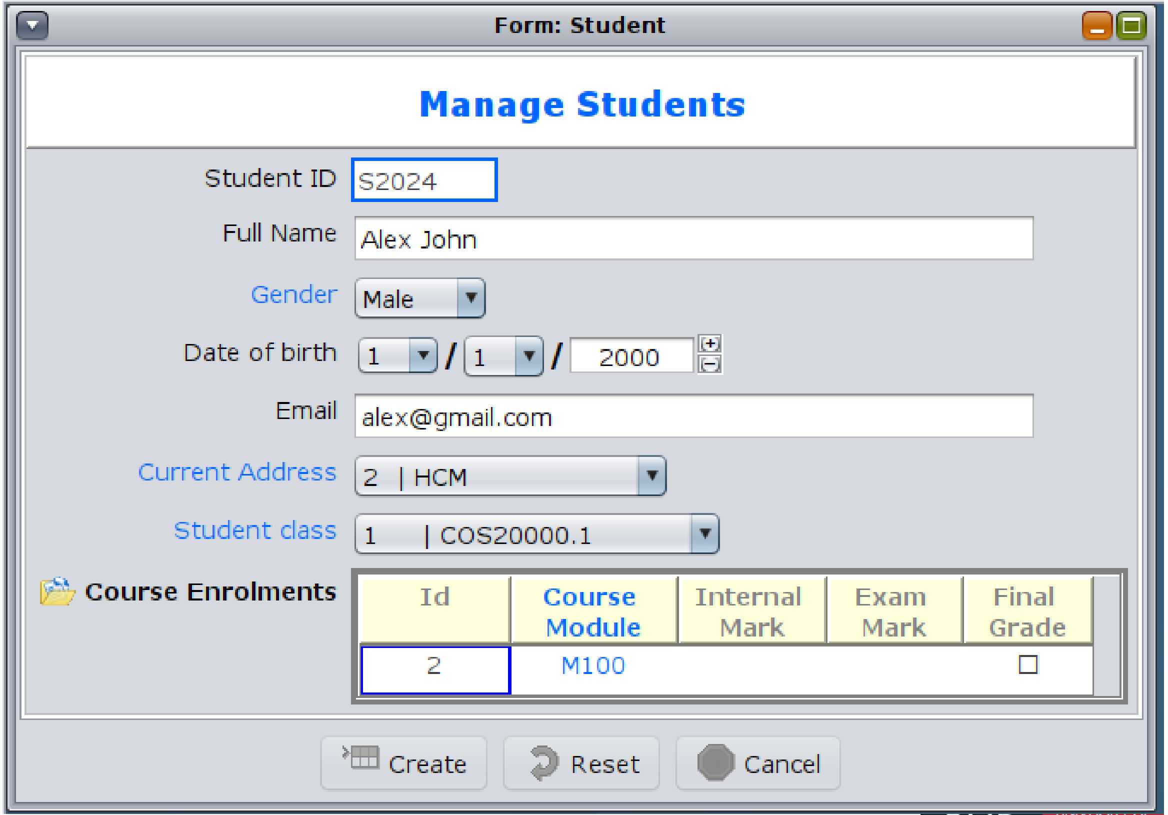
```
@ModuleDescriptor(...)
public class ModuleStudent {
  @AttributeDesc(label = "Manage Students")
  private String title;

  ...


  @AttributeDesc(label = "Course Enrolments")
  private Collection<Enrolment> enrolments;
}
```



```
swgen [jda-eg-coursemansw]
  docs
  log
  src
    main
      java
        org.jda.example
          coursemansw
          coursemanswref
            exceptions
            modules
              address
              coursemodule
              enrolment
              student
                model
                ModuleStudent
              studentclass
              ModuleMain
            software
              config
                SCCCourseMan
                SCCCourseManDerby
              CourseManSoftwareRef
            utils
      resources
```

Software generator

- Desktop



Manage Students

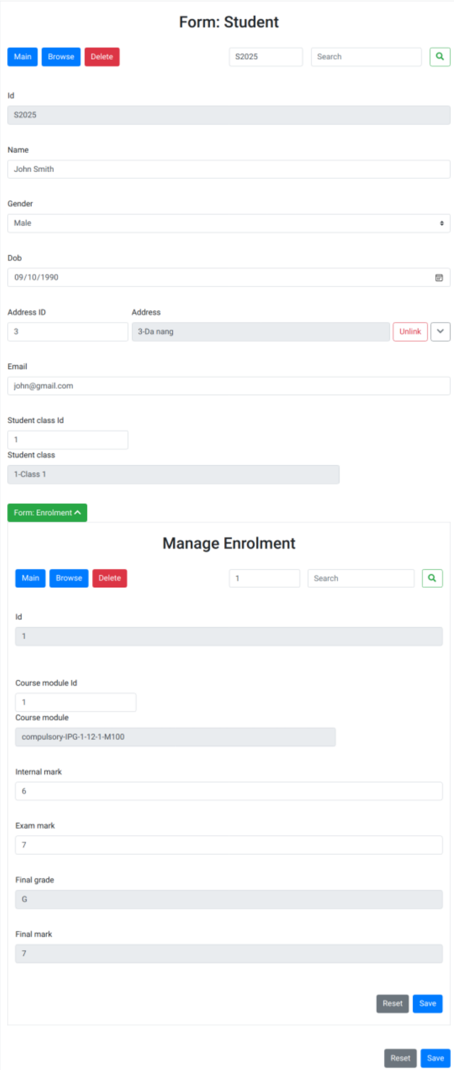
Student ID	S2024
Full Name	Alex John
Gender	Male
Date of birth	1 / 1 / 2000
Email	alex@gmail.com
Current Address	2 | HCM
Student class	1 | COS20000.1

Course Enrolments

Id	Course Module	Internal Mark	Exam Mark	Final Grade
2	M100			□

Software generator

- Web



Form: Student

S2025

Id: S2025

Name: John Smith

Gender: Male

Dob: 09/10/1990

Address ID: 3

Address: 3-Da nang

Email: john@gmail.com

Student class Id: 1

Student class: 1-Class 1

Manage Enrolment

1

Id: 1

Course module Id: 1

Course module: compulsory-IPG-1-12-1-M100

Internal mark: 6

Exam mark: 7

Final grade: G

Final mark: 7

(E) Project update

Project update

- Plan the implementation of your software prototype
  - major-specific mark in the project rubric

Tutorial & Workshop

See Canvas.