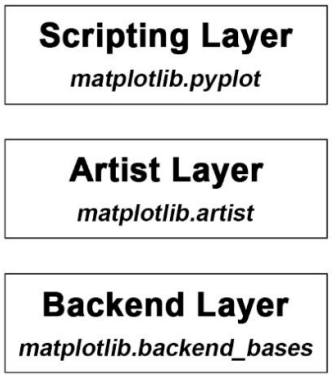
Architectural Patterns/Styles

1.1 mathplotlib

- **Propose :** เป็น library ที่ช่วยให้สามารถทำการ visualize ข้อมูลทางสถิติได้ซึ่งสามารถทำได้ทั้งรูปแบบภาพนิ่ง เป็น animation หรือแบบที่ผู้ใช้สามารถมีปฏิสัมพันธ์ด้วยได้ เพื่อที่จะทำให้สามารถวิเคราะห์ภาพรวมของข้อมูลได้ง่ายมากขึ้น
- **Architecture :** Architecture ของ mathplotlib นั้นแบ่งออกเป็นเลเยอร์ทั้งหมด 3 เลเยอร์โดยแต่ละเลเยอร์คือ Scripting Layer เป็นเลเยอร์ที่เขียน script น้อยที่สุดจากทั้ง 3 เลเยอร์มีไว้เพื่อทำให้ matplotlib ทำงานได้เหมือนกับ MATLAB เลเยอร์ต่อไปคือ Artist Layer ทำให้สามารถควบคุมและปรับแต่ง Figure ของ mathplotlib ได้ และสุดท้าย Backend Layer ทำหน้าที่สื่อสารกับเครื่องมือวาดรูปของคอมพิวเตอร์



- **Quality Attributes :**
    - **Performance**
        - **Source of Stimulus :** User request
        - **Stimulus :** User ทำการเรียกใช้งาน api เพื่อเขียนกราฟ
        - **Artifacts :** Graphing component ของทั้ง 3 layer ใน matplotlib
        - **Environment :** Normal Runtime mode
        - **Response :** ระบบทำการสร้างกราฟตาม parameter ที่ user ใส่ลงไป
        - **Response measure :** เวลาที่ใช้ในการ generate กราฟตามข้อมูลที่ input ที่ user ได้ใส่ลงไป

- Usability
  - **Source of Stimulus :** User
  - **Stimulus :** การเปลี่ยนรูปแบบกราฟที่จะออกมาทั้งสีและแกน
  - **Artifacts :** function ที่ทำงานเมื่อมีการเขียนกราฟของทั้ง 3 layer
  - **Environment :** Normal Runtime mode
  - **Response :** ระบบทำการแก้ไขรูปแบบของกราฟตาม parameter ที่ user ใส่ลงไปเพื่อเปลี่ยนแปลงรูปแบบกราฟให้ไม่เป็นรูปแบบปกติ
  - **Response measure :** จำนวนการทำงานที่ได้ผลตามที่ user ต้องการ
- Integrability
  - **Source of Stimulus :** Stakeholder (User)
  - **Stimulus :** นำ library matplotlib ไปใช้กับ python application
  - **Artifacts :** matplotlib ทั้งระบบ
  - **Environment :** Development
  - **Response :** component ของ python application ที่นำ matplotlib ไป integrate สามารถรับข้อมูลจาก matplotlib ได้
  - **Response measure :** ไม่ส่งผลต่อการทำงานอื่นๆใน application

- **แหล่งที่มา :**
  - https://www.aosabook.org/en/matplotlib.html
  - https://matplotlib.org/
  - https://medium.datadriveninvestor.com/data-visualization-with-python-matplotlib-architecture-6b05af533569

1.2 Audacity

- **Propose :** เป็นโปรแกรมที่มีไว้เพื่ออัดเสียง และตัดต่อเสียงต่างๆ โดยมีจุดประสงค์หลักคือการที่ผู้ใช้นั้นสามารถใช้งานโปรแกรม Audacity ได้โดยที่ไม่ต้องมีคู่มือการใช้งานแล้วยังสามารถค่อยๆค้นพบฟังก์ชั่นการใช้งานต่างๆที่มีอยู่เพิ่มขึ้นเรื่อยๆ
- **Architecture :** Audacity นั้นถูกแบ่งเป็น layer โดยที่แต่ละ layer นั้นก็จะมี library หลายๆตัวทำงานอยู่โดยจะมี library ที่สำคัญอยู่ 2 ตัวได้แก่ Port-Audio ที่มีหน้าที่ให้การทำ low-level audio interface และ wxWidgets ที่ทำหน้าที่ไว้สร้าง GUI component ซึ่งจะมี layere ที่ชื่อว่า OS abstraction และ Platform Specific Implementation Layer เป็นชั้นที่มี condition code ที่ตัดสินใจว่าการทำงานนั้นจะทำงานในรูปแบบใดขึ้นอยู่กับ operating system ที่ใช้ในการทำงาน

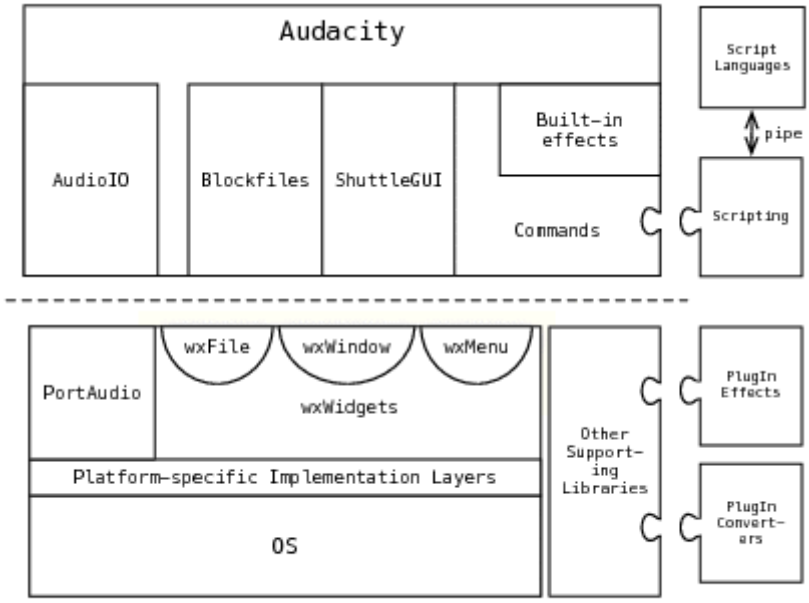


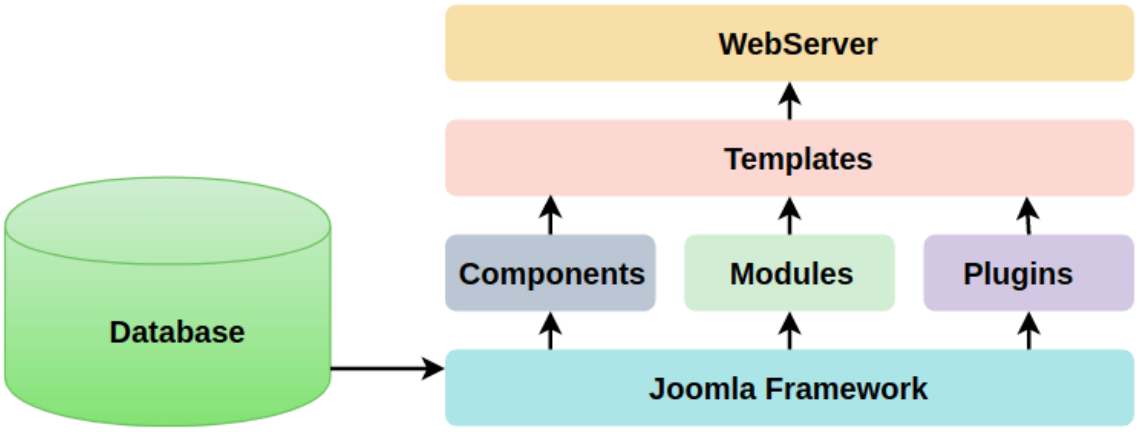
Figure 2.1: Layers in Audacity

- **Quality Attributes :**
  - **Usability**
    - **Source of Stimulus :** User
    - **Stimulus :** เรียนรู้ที่จะใช้งานซอร์ฟแวร์
    - **Artifact :** GUI
    - **Environment :** Runtime

        - **Response :** โชว์ function ต่างๆของโปรแกรมในรูปแบบที่ใช้งานง่าย
        - **Response measure :** จำนวนฟังก์ชั่นที่ user สามารถใช้งานได้โดยที่ไม่ต้องใช้คู่มือ
    - **Testability**
        - **Source of Stimulus :** Automated System Tester
        - **Stimulus :** ทดสอบความถูกต้องของระบบที่มีอยู่
        - **Artifacts :** General Function ที่มีอยู่ใน software
        - **Environment :** การใช้งาน function tester ของ user บน runtime
        - **Response :** ทำการ test และบันทึกผล
        - **Response Measure :** ความน่าจะเป็นที่จะเจอ function ที่ผิดพลาด
    - **Integrability**
        - **Source of Stimulus :** Stakeholder (User)
        - **Stimulus :** เพิ่ม plug-in ที่ต้องการ
        - **Artifacts :** component เฉพาะที่ทำหน้าที่รับ plug-in
        - **Environment :** Runtime, Integration
        - **Response :** New function จาก plug-in ที่เพิ่มเข้ามา
        - **Response Measure :** plug-in ที่เพิ่มเข้ามาไม่กระทบกับ function การทำงานหลัก
- **แหล่งที่มา :**
    - https://www.aosabook.org/en/audacity.html
    - **https://wiki.audacityteam.org/wiki/Quality**

2.1 Joomla!

- **Propose :** เป็น CMS (Control Management System) ที่ช่วยให้สามารถดูแลจัดการเนื้อหาต่างๆที่อยู่บนเว็บไซต์ ผ่านตัว Joomla! เองได้โดยไม่ต้องเขียนหน้า และระบบต่างๆ ของเว็บไซต์เพิ่มเติม โดย Joomla! นั้นถูกออกแบบมาให้ใช้งานและติดตั้งได้อย่างง่ายดาย และ ยังมี extension มากมายให้เลือกใช้เพื่อให้ได้ website ที่ตรงกับความต้องการของเรามากที่สุด
- **Architecture :** เป็นสถาปัตยกรรมรูปแบบ MVC (Model-View-Controller) ที่เขียนโดยภาษา php และใช้ MySQL เป็น database



Joomla Architecture

โดยการนำข้อมูลจากใน database มาให้ modules, components และ plugins ใช้ผ่านทาง Joomla Framework ซึ่งมี libraries และ packages ต่างๆอยู่ เพื่อที่จะสร้างเป็น templates ที่มีทั้ง backend และ frontend เพื่อให้ user เรียกไปใช้งานได้ตามสะดวก

- **Quality Attributes :**
  - **Usability**
    - **Source of Stimulus :** User
    - **Stimulus :** เรียกใช้ template ของ Joomla!
    - **Environment :** Runtime

        - **Artifact :** ทั้งระบบ
        - **Response :** User ได้รับ Templates ที่เรียกใช้
        - **Response Measure :** ความพึงพอใจของ user ต่อ Template ที่ได้รับ, เวลาให้การทำ Template เพื่อที่จะส่งให้ user
    - **Integrability**
        - **Source of Stimulus :** Developer
        - **Stimulus :** นำ plugin มาเชื่อมต่อ
        - **Artifacts :** Joomla! extension
        - **Environment :** Joomla!, extension
        - **Response :** สามารถใช้ extension นั้นร่วมกับ Joomla! ได้
        - **Response Measure :** จำนวนของ extension ที่สามารถใช้งานได้
    - **Modifiability**
        - **Source of Stimulus :** User
        - **Stimulus :** เพิ่ม function ใหม่
        - **Artifacts :** Code
        - **Environment :** design time
        - **Response :** function ที่เขียนเพิ่ม ถูกเพิ่ม
        - **Response Measure :** เวลาที่ใช้ในการสร้าง function ใหม่
- **แหล่งที่มา :**
    - https://docs.joomla.org/Portal:Learn_More
    - https://www.javatpoint.com/architecture-of-joomla
    - https://extensions.joomla.org/