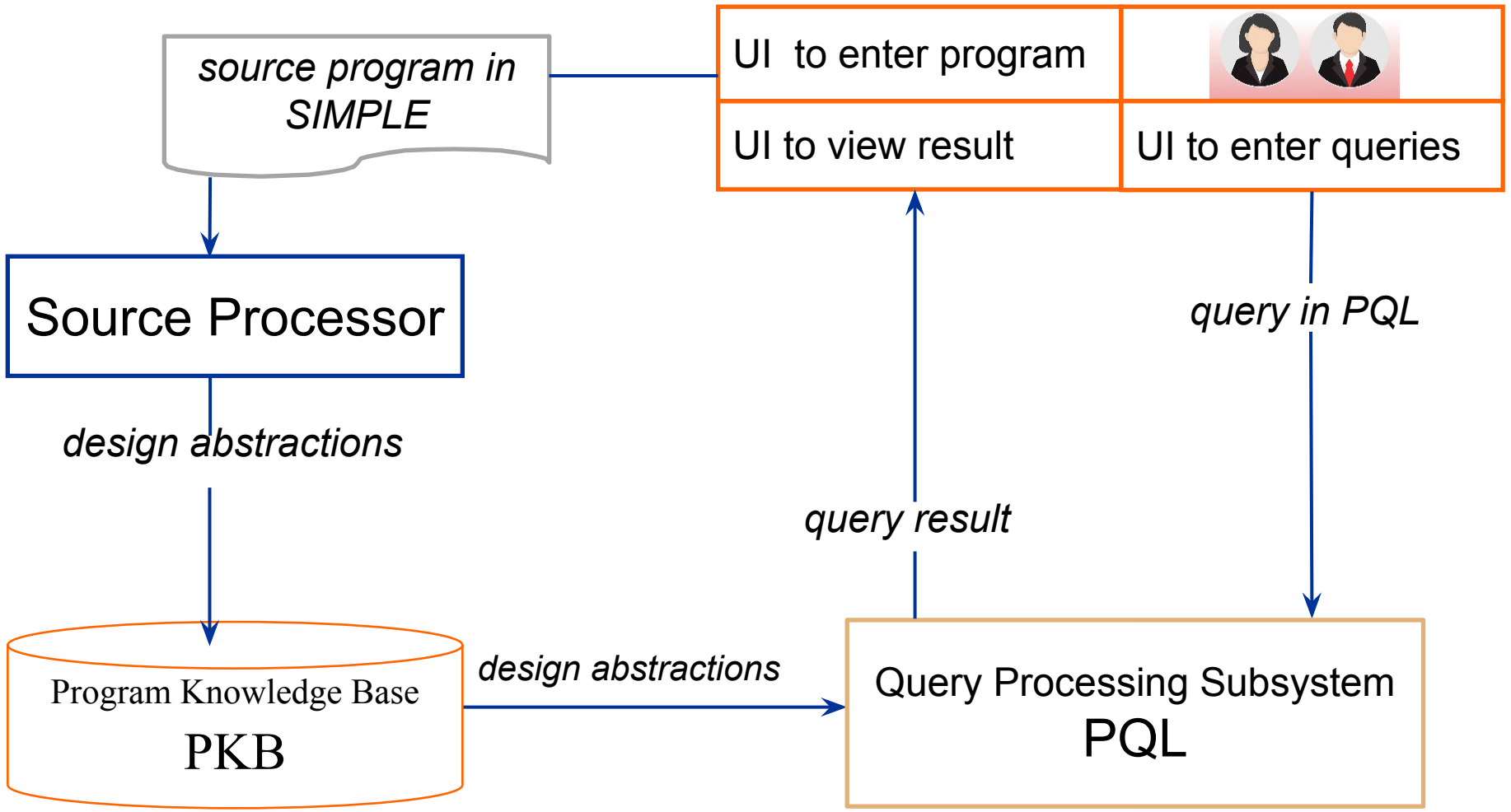
CS3203: Software Engineering Project

Tools for the Project

More info at - Wiki > Tools

SPA – Static Program Analysis Tool



Version Control System - Git

Make sure you brush up on your Git skills:

https://try.github.io/

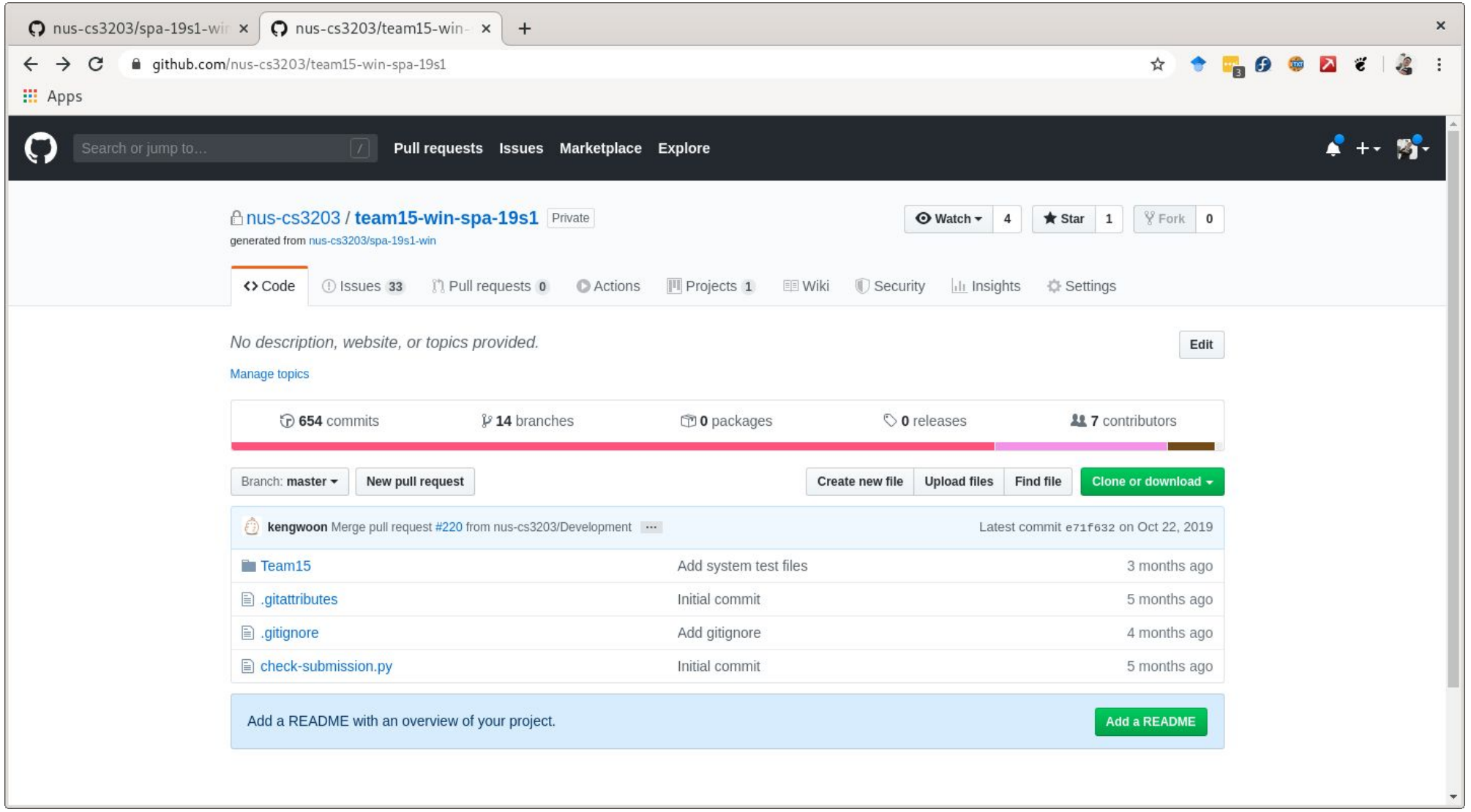
Minimally, how to do the 3 things on Git

- **Git Add**: Add files to be committed
- **Git Commit**: Commit with message
- **Git Push**: Push to remote Git server

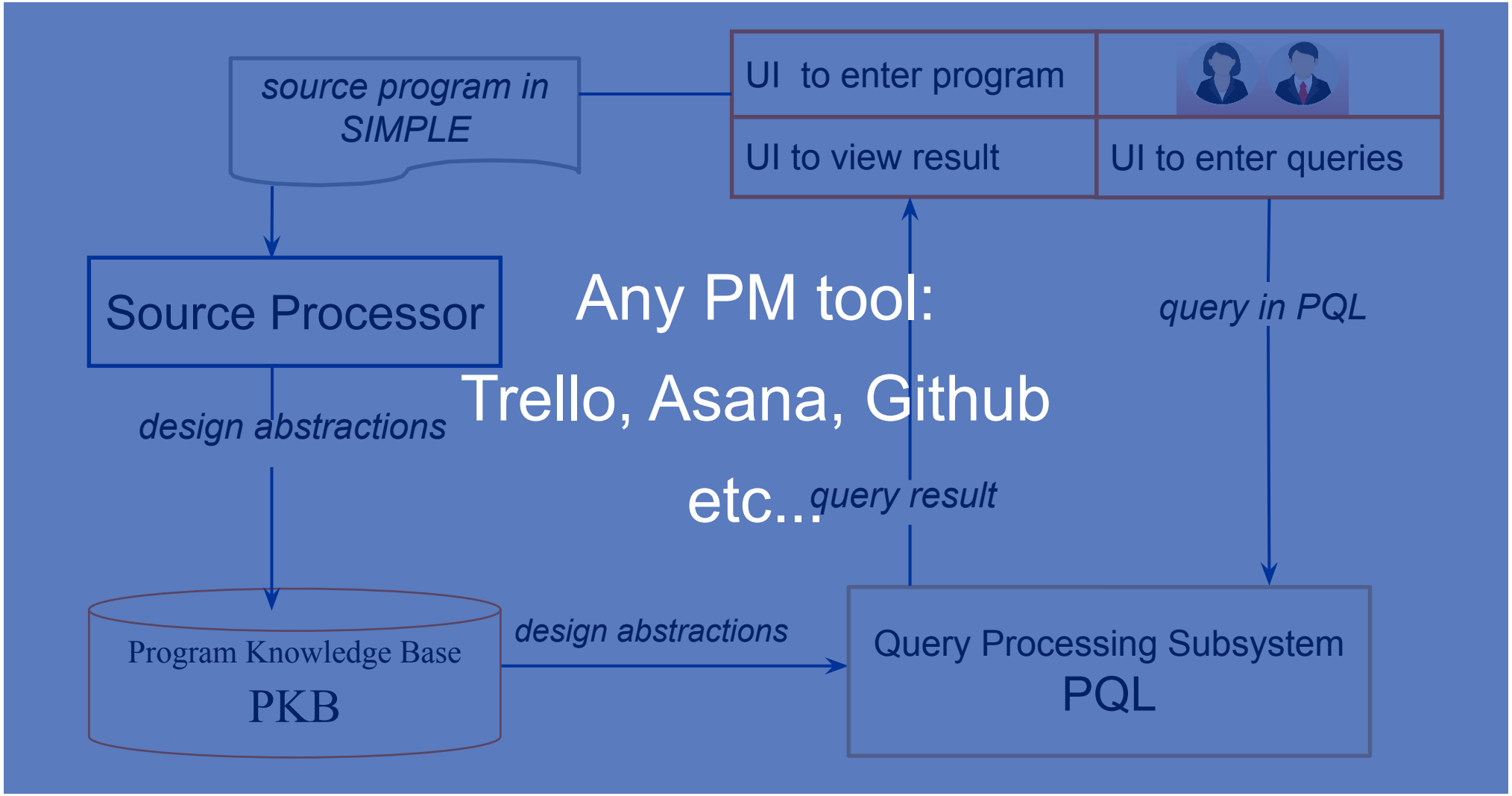
Git use in CS3203

- All teams **must** use the Git repository provided
  - Do not fork, use branches (Your tutors will be added)
- Each team must declare the info:
  - All GitHub IDs of the members of the team
  - Choice of startup solution (Windows or Cross-platform)
- Cross-platform, please also state the target environment (Windows, Mac or Linux)! See Wiki for more information!
  - **Make sure to test your system against the target env!**

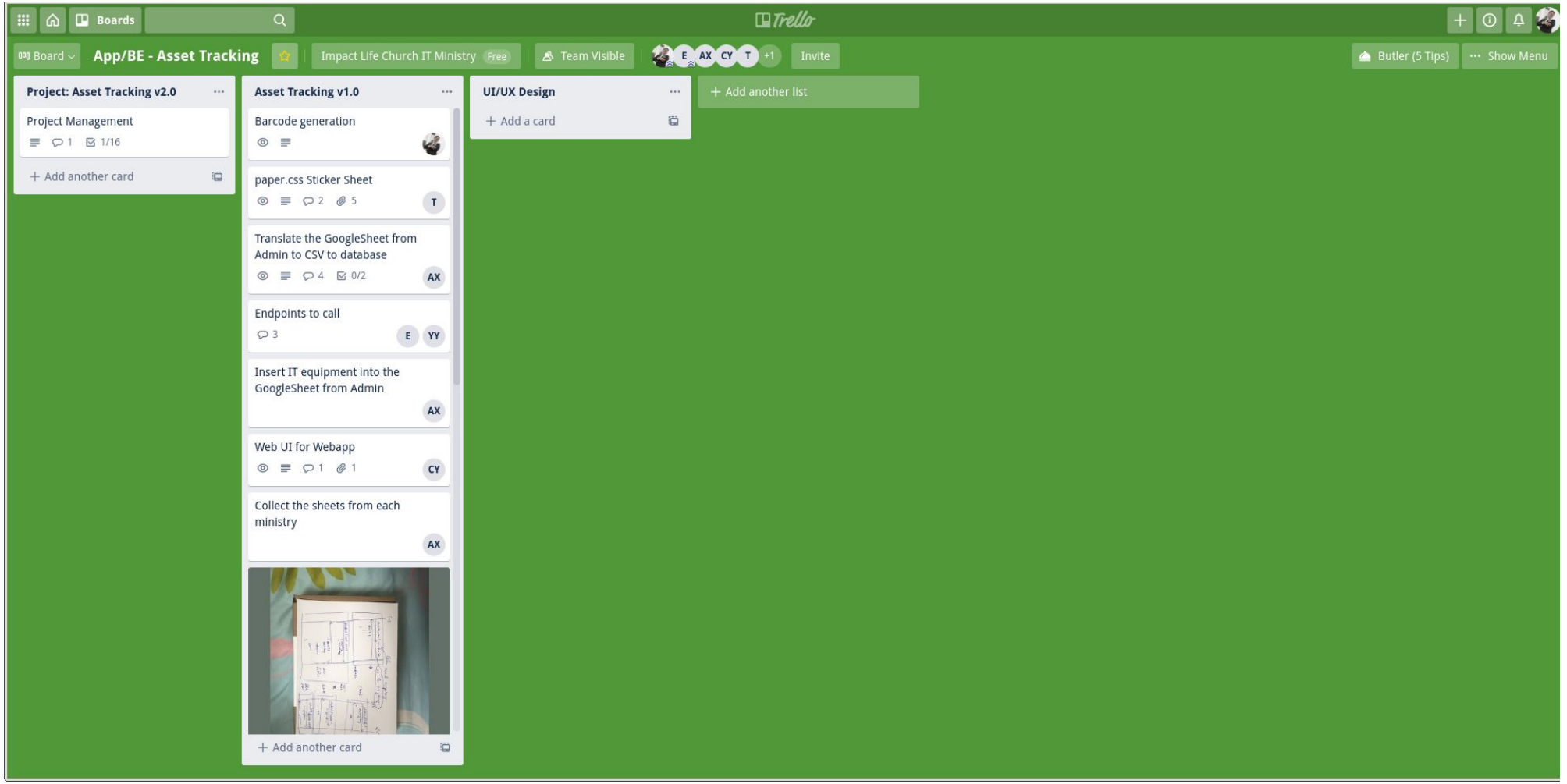
Github Education - Git



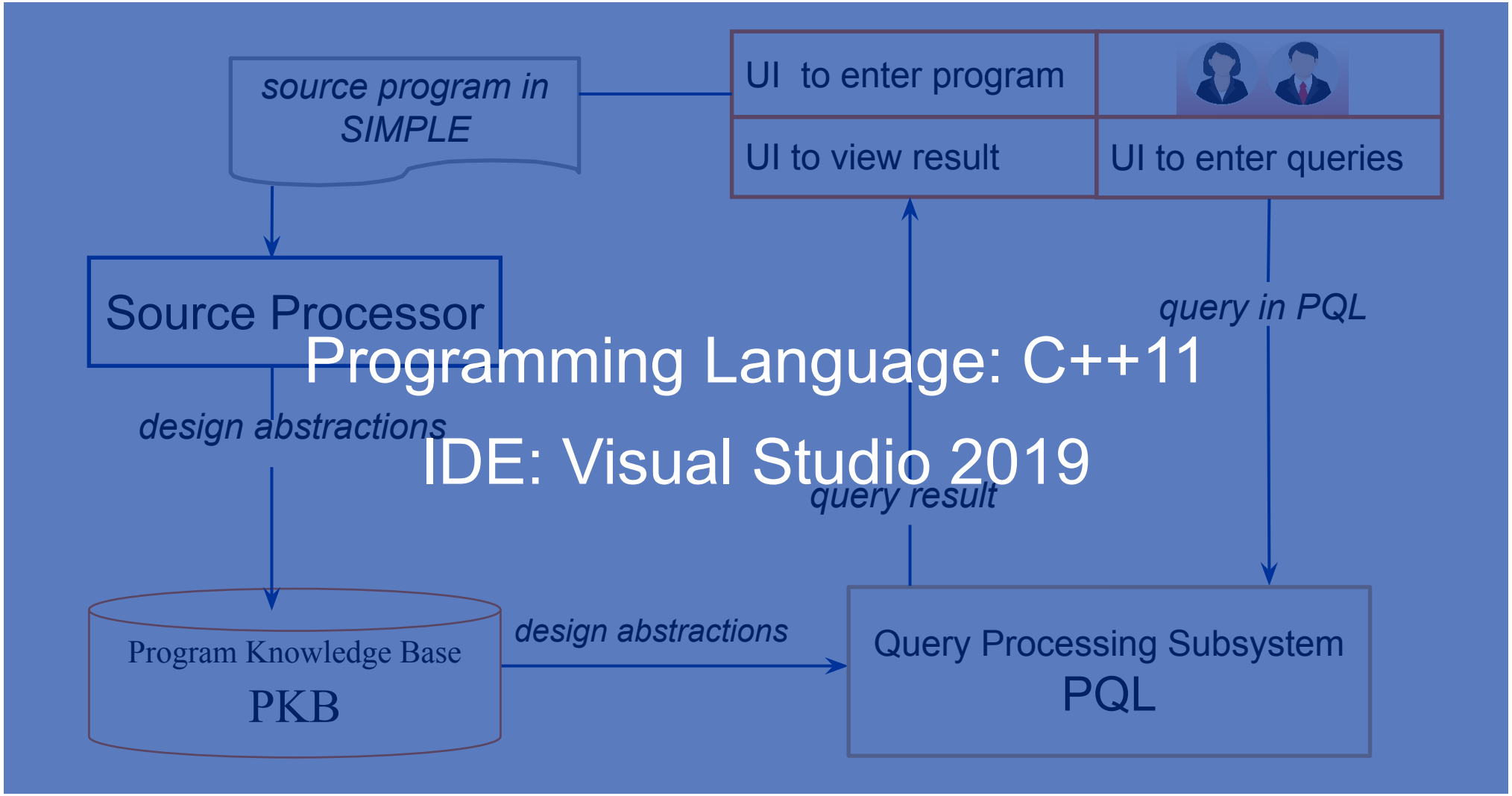
Project Management



Trello


Boards
Trello
Board
App/BE - Asset Tracking
Impact Life Church IT Ministry
Free
Team Visible
E
AX
CY
T
+1
Invite
Butler (5 Tips)
Show Menu
Project: Asset Tracking v2.0
Project Management
1
1/16
Add another card
Asset Tracking v1.0
Barcode generation
paper.css Sticker Sheet
2
5
T
Translate the GoogleSheet from Admin to CSV to database
4
0/2
AX
Endpoints to call
3
E
YY
Insert IT equipment into the GoogleSheet from Admin
AX
Web UI for Webapp
1
1
CY
Collect the sheets from each ministry
AX
Add another card
UI/UX Design
Add a card
Add another list

Development Environment



Startup SPA Development

- Windows Startup SPA Solution (Official/Recommended):
  - VS2019 Enterprise

Easy

- Cross-platform Startup SPA Solution:
  - IDE:
    - VS2019 Enterprise
    - Clion with Make
  - OS (See Target Environment):
    - Windows
    - MacOS
    - Linux

Not-so-easy

Notes on check-submission.py

Its a python tool that you should run to make sure your project satisfy the basic submission requirements (you should still check the requirements)!

$ python check-submission.py

This script will check for basic compliance with the submission requirements.

Disclaimer: you are still responsible for your submission, this check is by no means complete.

[Failed] - Team number must be valid.

Visual Studio

Windows

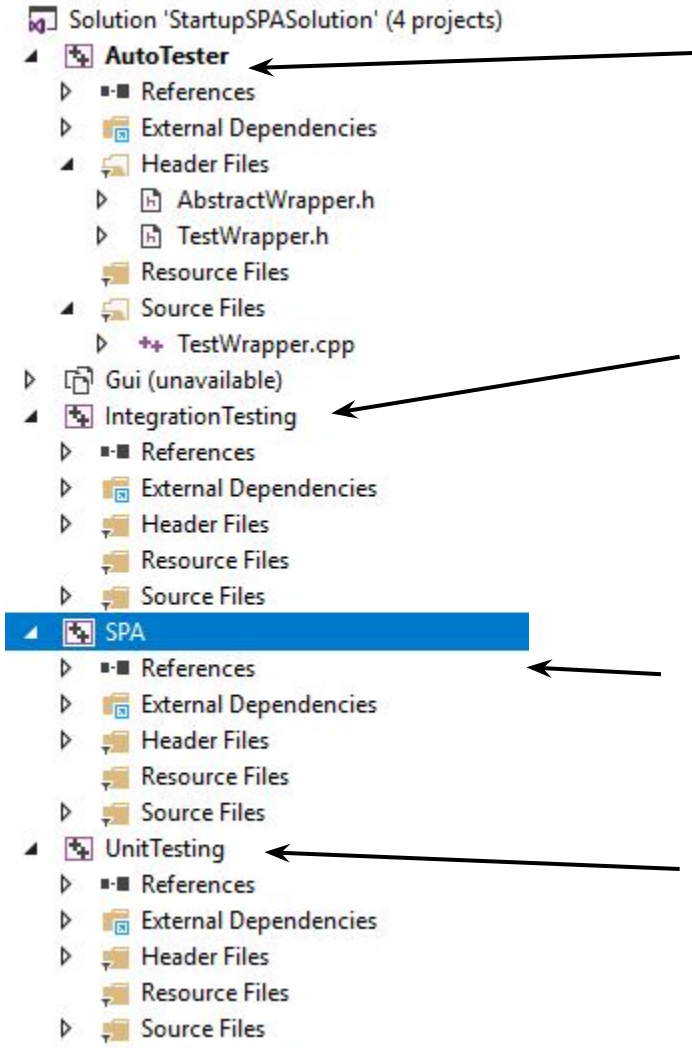
Open Startup SPA Solution

- The easy way to start!
- Available in "Tools" folder in IVLE Workbin

- Important files:
  - StartupSPASolution.sln
  - Read the documentation!

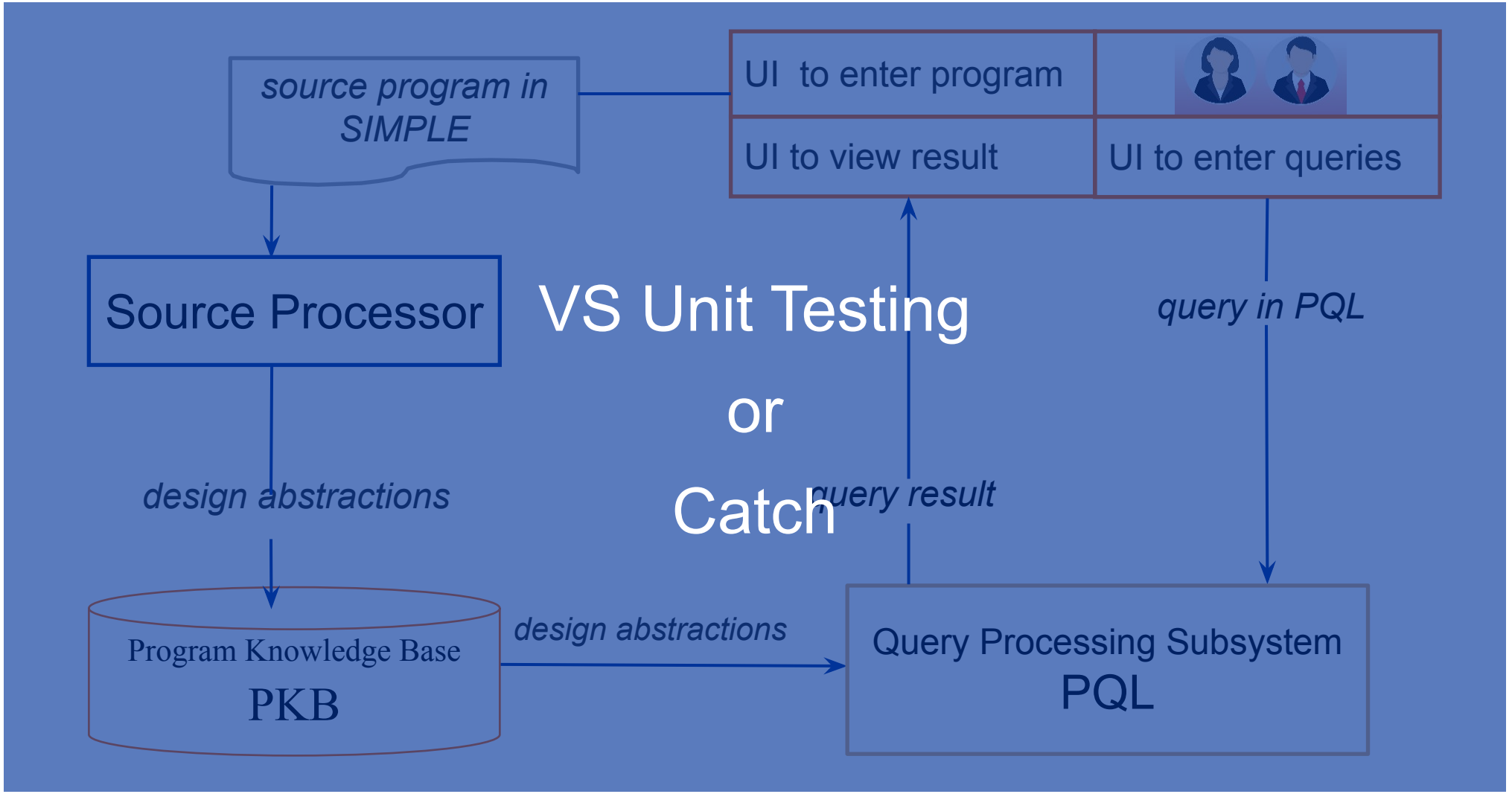
- Open and build!

Analyze the Project Properties



- **AutoTester:**
  Build to get AutoTester.exe for your SPA
- **IntegrationTesting:**
  Implement IntegrationTest.cpp and test classes, build and run for integration testing for your SPA
- **Empty SPA project:**
  Fill in the code for your SPA
- **UnitTesting:**
  Implement UnitTest.cpp and test classes, build and run for unit testing for your SPA

SPA Testing


source program in SIMPLE
UI to enter program
UI to view result
UI to enter queries
Source Processor
VS Unit Testing
or
Catch
query in PQL
design abstractions
query result
Program Knowledge Base
PKB
design abstractions
Query Processing Subsystem
PQL

Project Properties

- Compiling:
  - Turning source code into object code
  - VS2019: Additional Include Directories
- Linking:
  - Combining all the object code with the libraries into binaries
  - VS2019: Additional Dependencies
- Building:
  - The whole sequence from compiling to linking

Debug vs Release

- Debug:
  - PDB files are created: think of a lot more code added into your code to enable debugging
  - Not optimized as much
  - Autotester settings
- Release:
  - Optimized
  - What we will run our tests on

Running the AutoTester

- Running from the command line (cmd in Windows)

```
> AutoTester.exe ..\Tests\Sample_source.txt
  ..\Tests\Sample_queries.txt
  ..\Tests\out.xml
```

  - Open output.xml in Mozilla Firefox to see the results
  - Note: analysis.xsl is in the same directory with output.xml
- Running from VS in Debug mode
  - Press the green Run button
  - Open out.xml in Mozilla Firefox to see the results

GUI vs Autotester

- GUI:
  - Optional
  - For your own benefit
- Autotester
  - Implement your code into autotester
  - Mandatory
  - Will be used in grading

CMake

Windows, Mac, Linux

Cross Platform… But...

We strongly recommend you to use the same OS/IDE/Build system across every member!

- Windows / VS / CMake
- MacOS / AppleClang / CMake
- Linux(Fedora 30) / GCC / CMake

You can mix OS/IDE/Build Systems… But...

No additional marks will be given.

Support will be given on best-effort basis.

CMake

Make sure you check out the tutorial:

https://cmake.org/cmake/help/latest/guide/tutorial/index.html

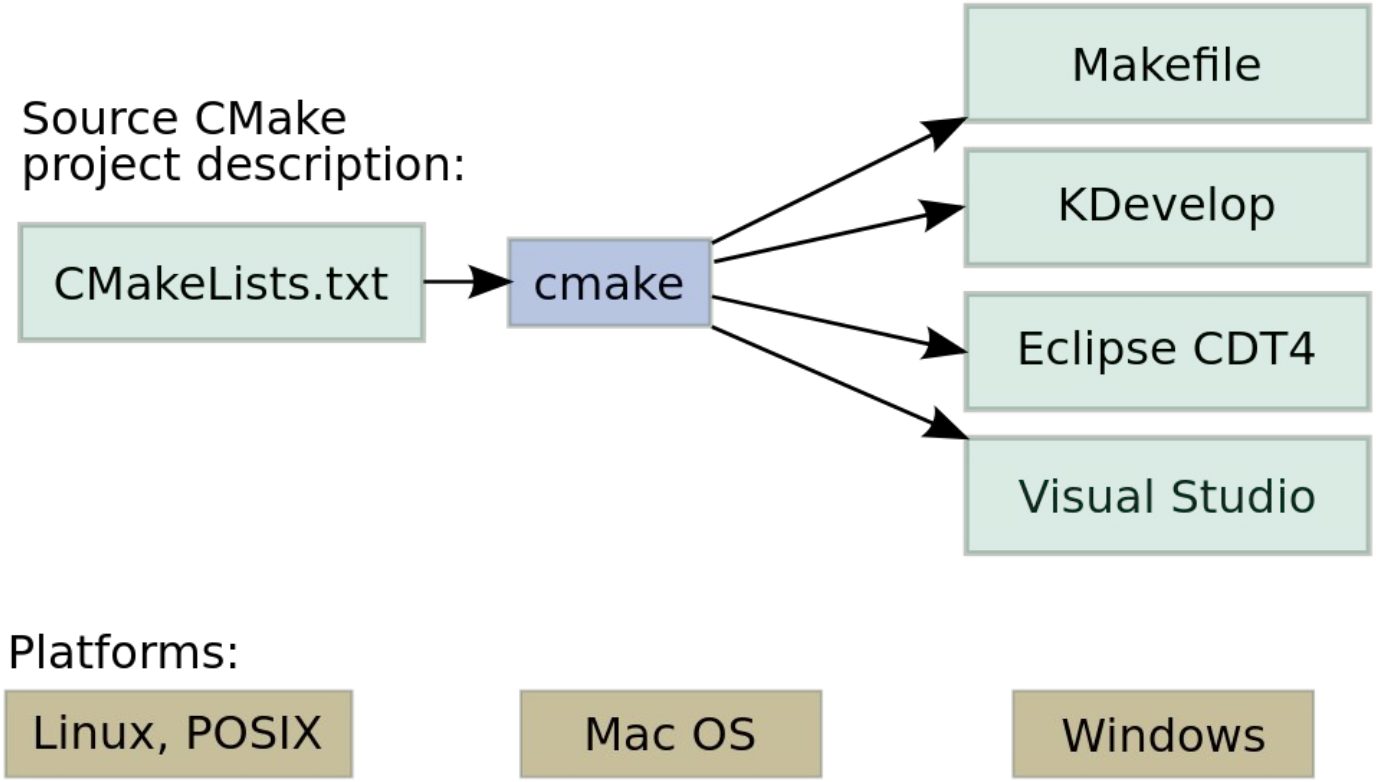
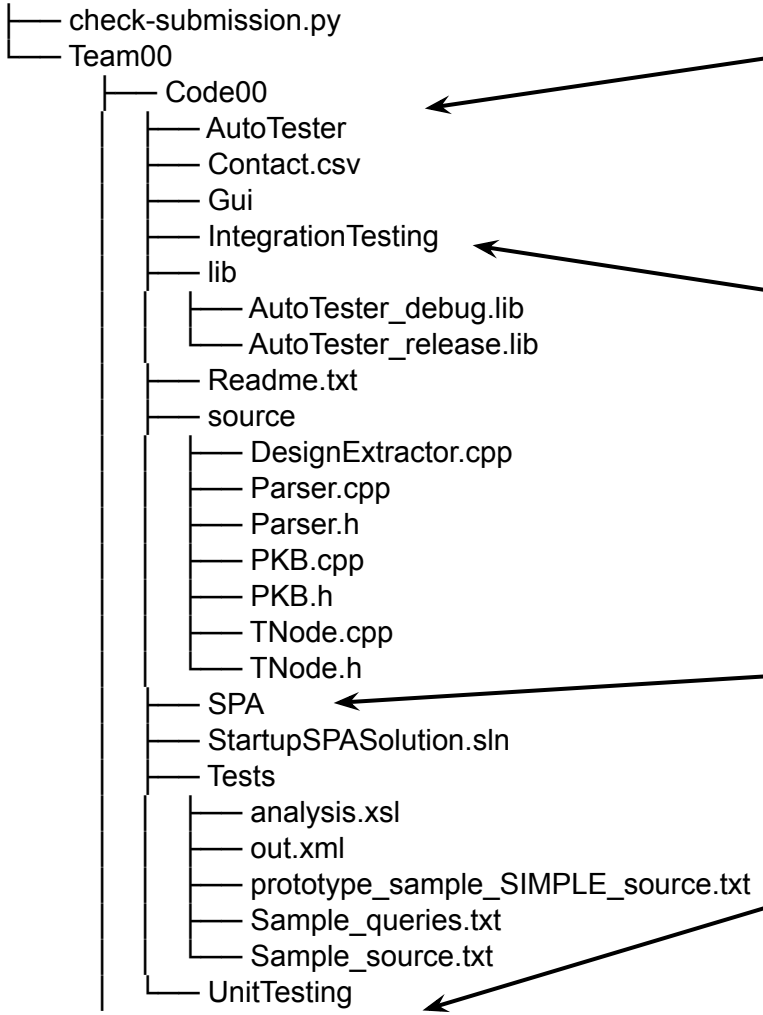


Image Credits: https://www.shlomifish.org/lecture/CMake

Analyze the Project Properties



- **AutoTester:**
  Build to get AutoTester.exe for your SPA
- **IntegrationTesting:**
  Implement IntegrationTest.cpp and test classes, build and run for integration testing for your SPA
- **Empty SPA project:**
  Fill in the code for your SPA
- **UnitTesting:**
  Implement UnitTest.cpp and test classes, build and run for unit testing for your SPA

Running the AutoTester

- Running from the command line (cmd in Windows)

```
> ./autotester ../../../tests/Sample_source.txt
    ../../../tests/Sample_queries.txt
    ../../../tests/out.xml
```

  - Open output.xml in Mozilla Firefox to see the results
  - Note: analysis.xsl is in the same directory with output.xml
- Build and Compile using your platform instructions