MINISTRY OF EDUCATION

TECHNICAL UNIVERSITY
OF CLUJ-NAPOCA, ROMANIA

FUNDAMENTAL PROGRAMMING TECHNIQUES

Laboratory Resources

Tudor Cioara
Claudia Daniela Pop

Ionut Anghel
Mitrea Dan

Marcel Antal
Cristina Bianca Pop

2023-2024

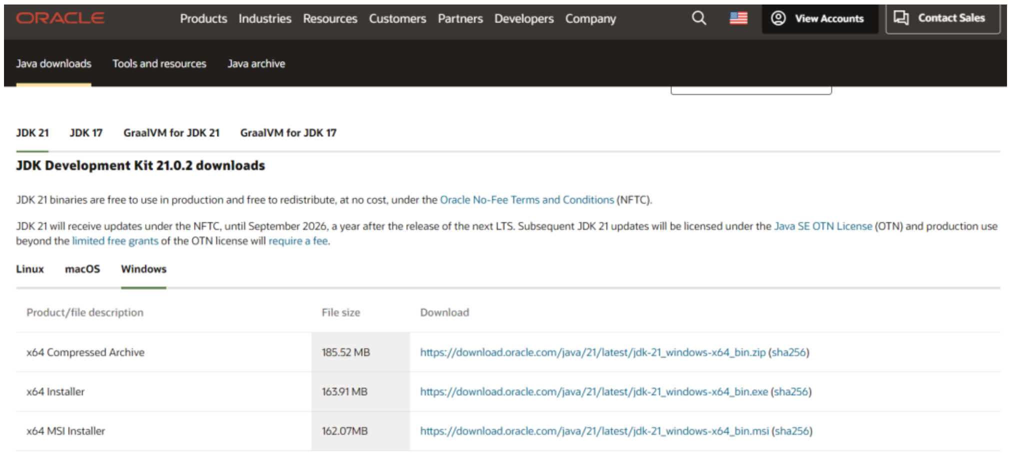
Contents

1 Java

1.1 Java JDK

1) Access the next link:

https://www.oracle.com/java/technologies/downloads/



JDK Development Kit 21.0.2 downloads

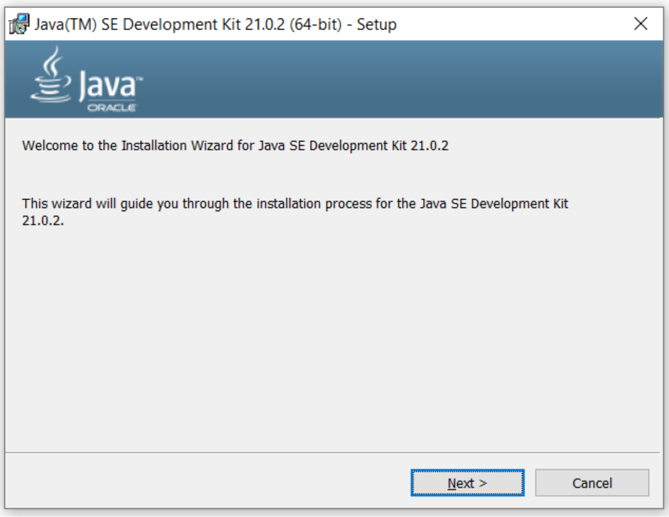
JDK 21 binaries are free to use in production and free to redistribute, at no cost, under the Oracle No-Fee Terms and Conditions (NFTC).

JDK 21 will receive updates under the NFTC, until September 2026, a year after the release of the next LTS. Subsequent JDK 21 updates will be licensed under the Java SE OTN License (OTN) and production use beyond the limited free grants of the OTN license will require a fee.

Product/file description	File size	Download
x64 Compressed Archive	185.52 MB	https://download.oracle.com/java/21/latest/jdk-21_windows-x64_bin.zip (sha256)
x64 Installer	163.91 MB	https://download.oracle.com/java/21/latest/jdk-21_windows-x64_bin.exe (sha256)
x64 MSI Installer	162.07MB	https://download.oracle.com/java/21/latest/jdk-21_windows-x64_bin.msi (sha256)

2) Click on the link which corresponds to your version of the Operating System. In the example the version which is used corresponds to Windows x64 and the file is named https://download.oracle.com/java/21/latest/jdk-21_windows-x64_bin.exe.

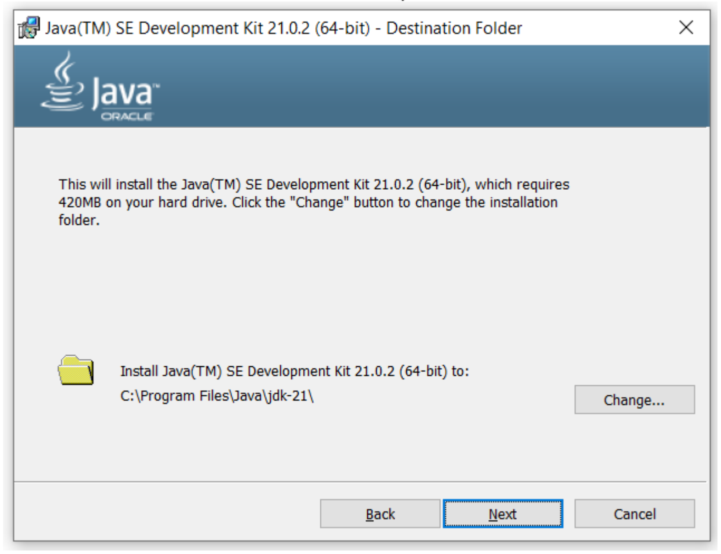
3) Click on *jdk-21_windows-x64_bin.exe*. The window below will appear. Click on the *Next* button.



Welcome to the Installation Wizard for Java SE Development Kit 21.0.2

This wizard will guide you through the installation process for the Java SE Development Kit 21.0.2.

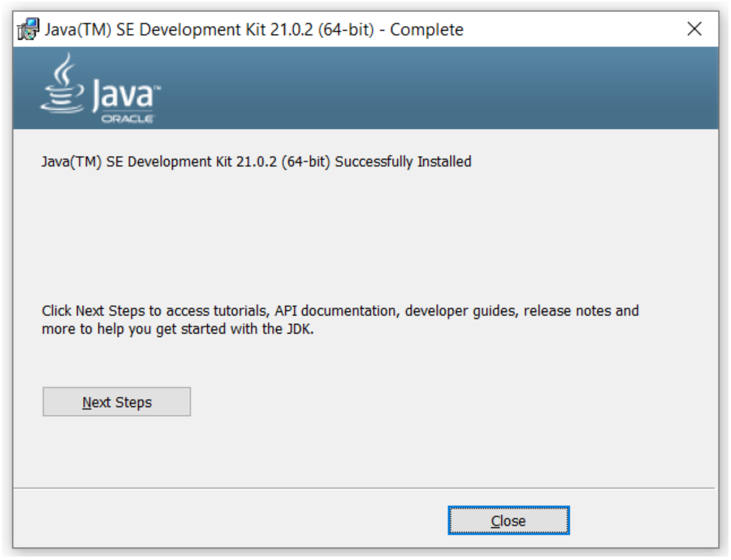
4) You will be asked where you want to install Java. Use the default location and click Next.


Java(TM) SE Development Kit 21.0.2 (64-bit) - Destination Folder

This will install the Java(TM) SE Development Kit 21.0.2 (64-bit), which requires 420MB on your hard drive. Click the "Change" button to change the installation folder.

Install Java(TM) SE Development Kit 21.0.2 (64-bit) to:
C:\Program Files\Java\jdk-21\

5) After the installation is successfully completed the window below will be displayed. Click Close.


Java(TM) SE Development Kit 21.0.2 (64-bit) - Complete

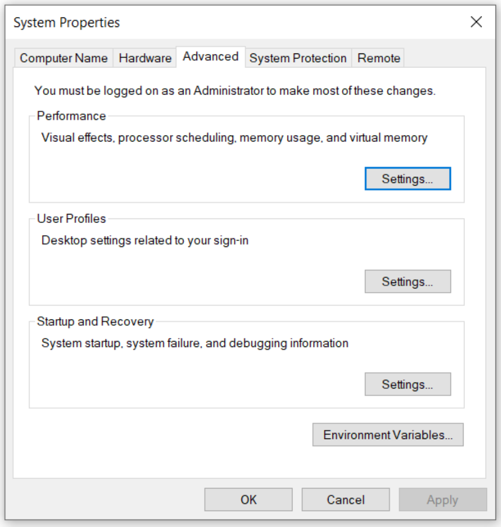
Java(TM) SE Development Kit 21.0.2 (64-bit) Successfully Installed

Click Next Steps to access tutorials, API documentation, developer guides, release notes and more to help you get started with the JDK.

1.2 Set the JAVA_HOME variable

Note: the steps may vary according to the Windows version installed on the computer

1) Write in the search bar: "Edit the system environment variables" and click on the suggested option. The window below will be displayed.



System Properties

Computer Name | Hardware | Advanced | System Protection | Remote

You must be logged on as an Administrator to make most of these changes.

Performance
Visual effects, processor scheduling, memory usage, and virtual memory

User Profiles
Desktop settings related to your sign-in

Startup and Recovery
System startup, system failure, and debugging information

2) Click on *Environment Variables*.

3) Under *System Variables* click *New*.

4) In the text field associated with the name of the variable insert JAVA_HOME and in the field associated with the value of the variable insert C:\Program Files\Java\java_version;.

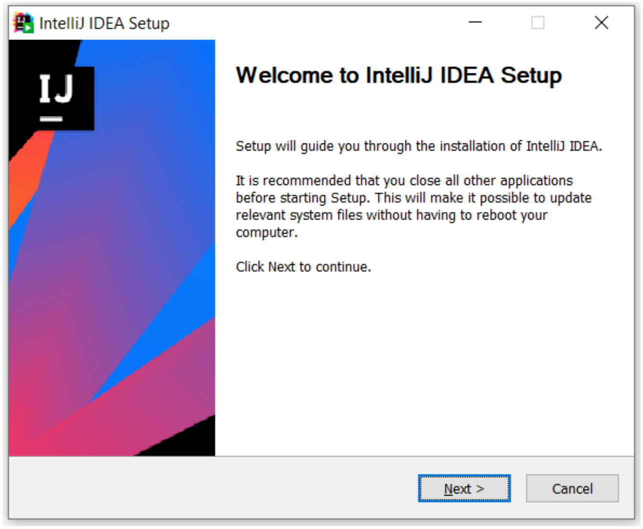
5) Click *OK*.

2. IntelliJ IDEA

We recommend using IntelliJ IDEA as the IDE for developing your applications during the laboratories. As a student, you can benefit of a free educational license for IntelliJ IDEA, by applying here https://www.jetbrains.com/community/education/#students -> Click on *Apply now* -> Fill in the requested information and click on *Apply for free products*. After obtaining the license, download the latest version of Intellij IDEA. Click on the downloaded file. You will be asked whether you allow the program to make changes to this computer -> Click Yes.

Then, follow the steps below:

1) Click next



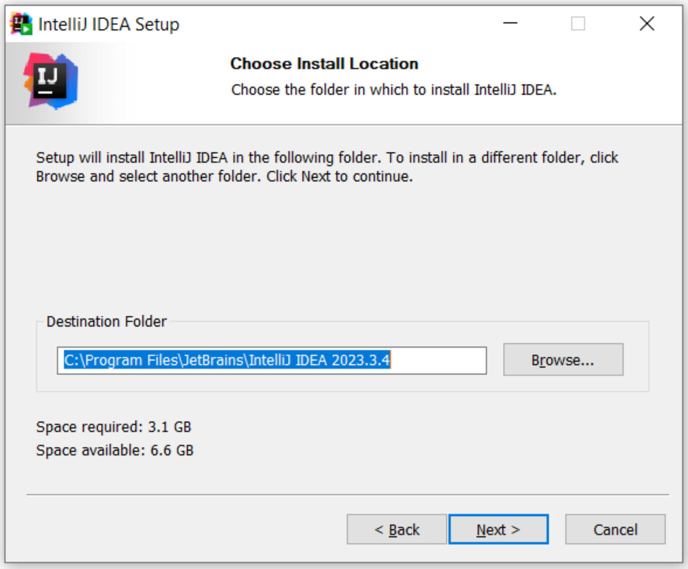
Welcome to IntelliJ IDEA Setup

Setup will guide you through the installation of IntelliJ IDEA.

It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

Click Next to continue.

2) You will be asked where you want to install Intellij IDEA. Use the default location and click Next.


IntelliJ IDEA Setup

Choose Install Location

Choose the folder in which to install IntelliJ IDEA.

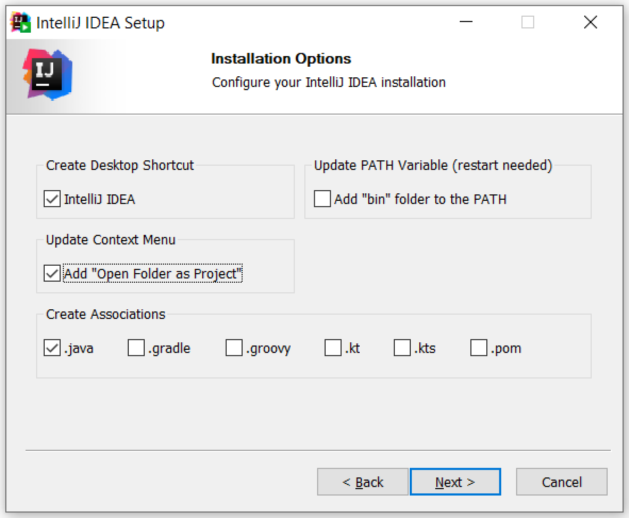
Setup will install IntelliJ IDEA in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.

Destination Folder

C:\Program Files\JetBrains\IntelliJ IDEA 2023.3.4

Space required: 3.1 GB
Space available: 6.6 GB

3) Configure the installation options (see an example in the image below) and click Next.


IntelliJ IDEA Setup

Installation Options

Configure your IntelliJ IDEA installation

Create Desktop Shortcut: [✓] IntelliJ IDEA

Update PATH Variable (restart needed): [ ] Add "bin" folder to the PATH

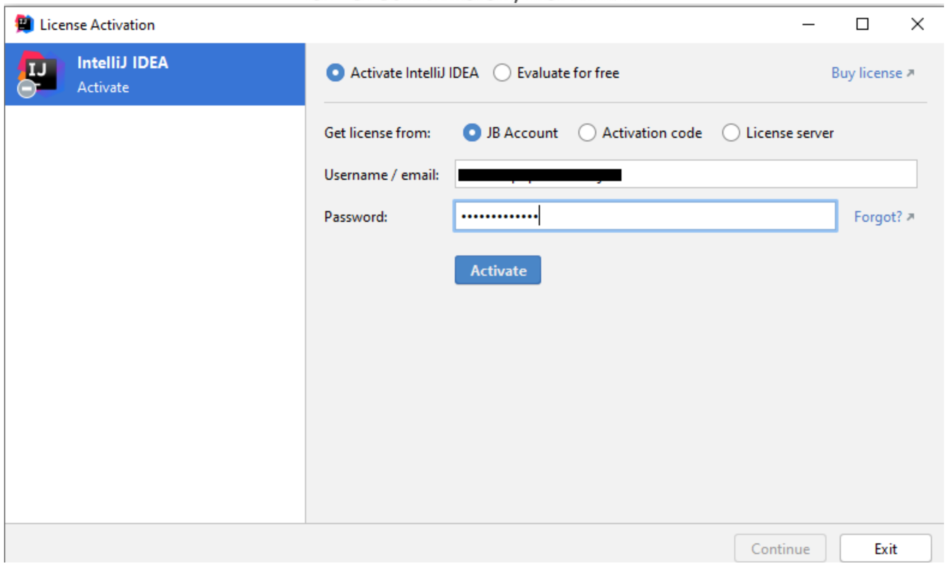
Update Context Menu: [✓] Add "Open Folder as Project"

Create Associations: [✓] .java [ ] .gradle [ ] .groovy [ ] .kt [ ] .kts [ ] .pom

4) Click Install.

5) After the installation completes click on *Finish.*

6) Run IntelliJ IDEA. When you first run IntelliJ IDEA after the installation completes you will have to activate it by inserting the username/email and password you used when creating the account for obtaining the students license. Click *Activate*.


License Activation

IntelliJ IDEA
Activate

[✓ Activate IntelliJ IDEA] Evaluate for free — Buy license ↗

Get license from: [✓ JB Account] Activation code License server

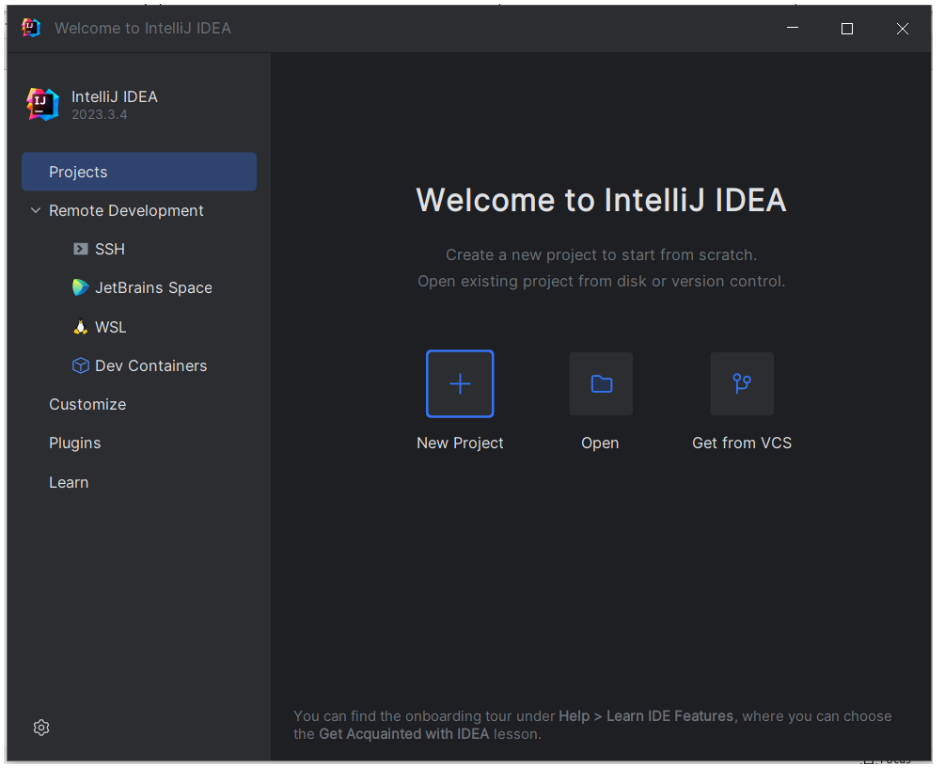
Username / email: [redacted]

Password: ••••••••••••• — Forgot? ↗

Activate

Continue Exit

7) After the activation completes, the IDE will open.


Welcome to IntelliJ IDEA

IntelliJ IDEA
2023.3.4

Projects
Remote Development
SSH
JetBrains Space
WSL
Dev Containers
Customize
Plugins
Learn

Welcome to IntelliJ IDEA

Create a new project to start from scratch.
Open existing project from disk or version control.

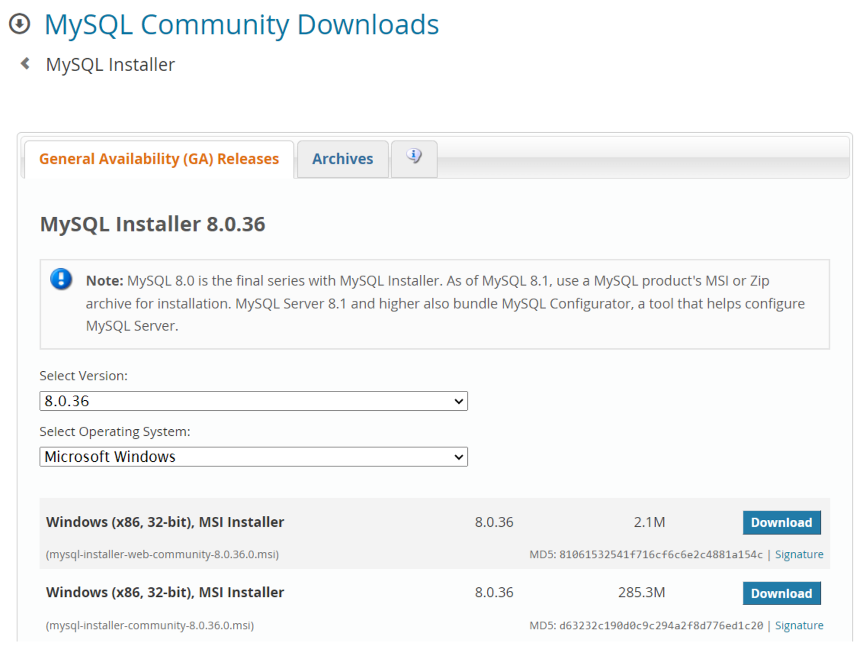
New Project Open Get from VCS

You can find the onboarding tour under **Help > Learn IDE Features**, where you can choose the **Get Acquainted with IDEA** lesson.

3. Database Server

A database server must be installed on the local machine. A MySQL server can be used to run locally the projects. For installing the MySQL server follow the next steps:

1) Click on the next link to download the latest version: https://dev.mysql.com/downloads/windows/installer/.



MySQL Community Downloads

MySQL Installer

MySQL Installer 8.0.36

Note: MySQL 8.0 is the final series with MySQL Installer. As of MySQL 8.1, use a MySQL product's MSI or Zip archive for installation. MySQL Server 8.1 and higher also bundle MySQL Configurator, a tool that helps configure MySQL Server.

Select Version: 8.0.36

Select Operating System: Microsoft Windows

Windows (x86, 32-bit), MSI Installer	8.0.36	2.1M	Download
(mysql-installer-web-community-8.0.36.0.msi)		MD5: 81061532541f716cf6c6e2c4881a154c | Signature	
Windows (x86, 32-bit), MSI Installer	8.0.36	285.3M	Download
(mysql-installer-community-8.0.36.0.msi)		MD5: d63232c190d0c9c294a2f8d776ed1c20 | Signature	

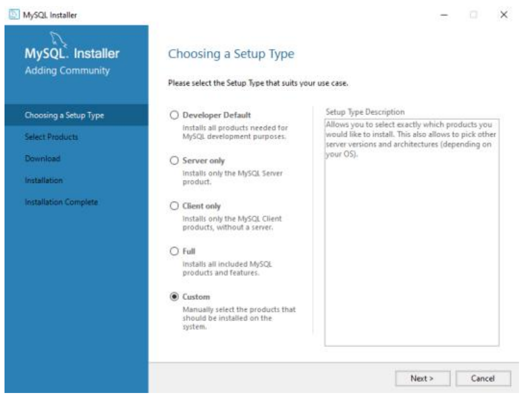
2) Click on the second Download button.

3) Click on *No thanks, just start my download*.

4) Click on the downloaded file.

5) You will be asked whether you allow the program to make changes to this computer -> Click *Yes*.

6) You will be asked to select the Setup Type that suits your use case. Select *Custom* and click *Next*.



MySQL Installer

MySQL. Installer
Adding Community

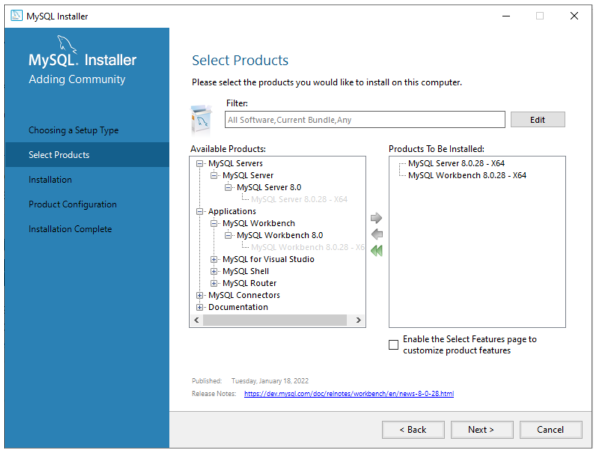
Choosing a Setup Type

Please select the Setup Type that suits your use case.

- ○ Developer Default — Installs all products needed for MySQL development purposes.
- ○ Server only — Installs only the MySQL Server product.
- ○ Client only — Installs only the MySQL Client products, without a server.
- ○ Full — Installs all included MySQL products and features.
- ● Custom — Manually select the products that should be installed on the system.

Setup Type Description: Allows you to select exactly which products you would like to install. This also allows to pick other server versions and architectures (depending on your OS).

7) You will be redirected to “Select Products and Features”. Select “MySQL Server *version*” and “MySQL Workbench *version*” and click *Next*.



MySQL Installer — Adding Community

Select Products

Please select the products you would like to install on this computer.

Filter: All Software,Current Bundle,Any

Available Products:
- MySQL Servers
  - MySQL Server
    - MySQL Server 8.0
      - MySQL Server 8.0.28 - X64
- Applications
  - MySQL Workbench
    - MySQL Workbench 8.0
      - MySQL Workbench 8.0.28 - X6
  - MySQL for Visual Studio
  - MySQL Shell
  - MySQL Router
- MySQL Connectors
- Documentation

Products To Be Installed:
- MySQL Server 8.0.28 - X64
- MySQL Workbench 8.0.28 - X64

☐ Enable the Select Features page to customize product features

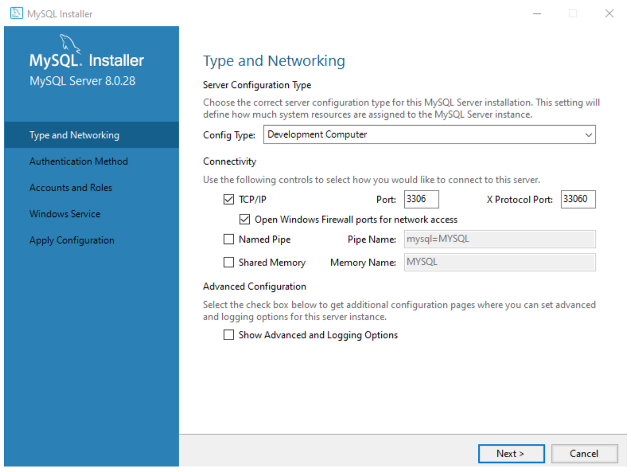
Published: Tuesday, January 18, 2022
Release Notes: https://dev.mysql.com/doc/relnotes/workbench/en/news-8-0-28.html

8) Click Next.

9) Click Execute.

10) Click Next.

11) Click Next and follow the steps for the configuration of the MySQL Server. At this stage pay attention to the port on which the MySQL server is running on (i.e., 3306 is the default port).



MySQL Installer — MySQL Server 8.0.28

Type and Networking

Server Configuration Type

Choose the correct server configuration type for this MySQL Server installation. This setting will define how much system resources are assigned to the MySQL Server instance.

Config Type: Development Computer

Connectivity

Use the following controls to select how you would like to connect to this server.

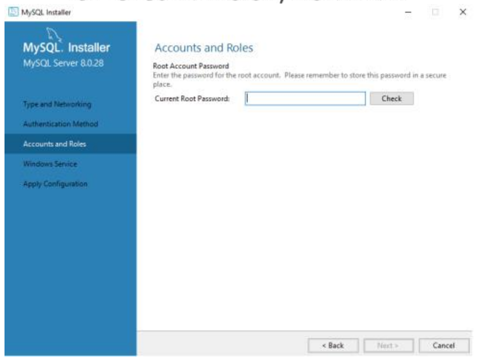
☑ TCP/IP Port: 3306 X Protocol Port: 33060
☑ Open Windows Firewall ports for network access
☐ Named Pipe Pipe Name: mysql=MYSQL
☐ Shared Memory Memory Name: MYSQL

Advanced Configuration

Select the check box below to get additional configuration pages where you can set advanced and logging options for this server instance.

☐ Show Advanced and Logging Options

- Click *Next*
- Select the Authentication Mode (select *Use Strong Password Encryption for Authentication*) and click *Next*.
- Set the password for the root account – this must be set by you and make sure you remember it as it will be used for further connections to the MySQL server. After inserting the password click on the *Check* button.


MySQL Installer

MySQL. Installer
MySQL Server 8.0.28

Type and Networking
Authentication Method
Accounts and Roles
Windows Service
Apply Configuration

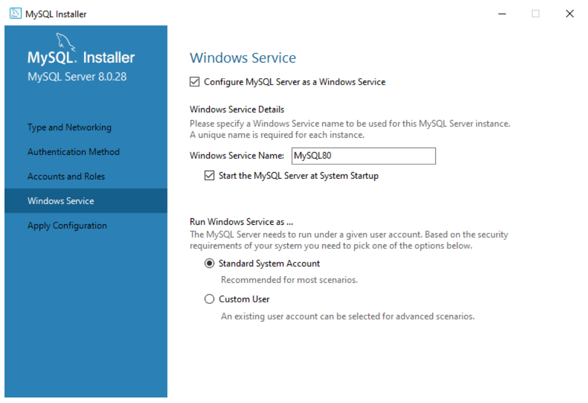
Accounts and Roles

Root Account Password
Enter the password for the root account. Please remember to store this password in a secure place.

Current Root Password: [ ] Check

< Back | Next > | Cancel

- Click *Next*


MySQL Installer

MySQL. Installer
MySQL Server 8.0.28

Type and Networking
Authentication Method
Accounts and Roles
Windows Service
Apply Configuration

Windows Service

[✓] Configure MySQL Server as a Windows Service

Windows Service Details
Please specify a Windows Service name to be used for this MySQL Server instance.
A unique name is required for each instance.

Windows Service Name: MySQL80

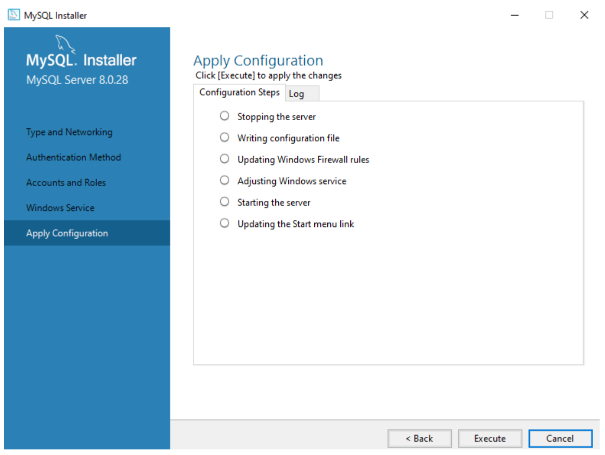
[✓] Start the MySQL Server at System Startup

Run Windows Service as ...
The MySQL Server needs to run under a given user account. Based on the security requirements of your system you need to pick one of the options below.

[✓ Standard System Account]
Recommended for most scenarios.

Custom User
An existing user account can be selected for advanced scenarios.

- Click *Next* and then *Execute*


MySQL Installer

MySQL. Installer
MySQL Server 8.0.28

Type and Networking
Authentication Method
Accounts and Roles
Windows Service
Apply Configuration

Apply Configuration
Click [Execute] to apply the changes

Configuration Steps | Log

- Stopping the server
- Writing configuration file
- Updating Windows Firewall rules
- Adjusting Windows service
- Starting the server
- Updating the Start menu link

< Back | Execute | Cancel

12) Click Finish.

Note: in case you uninstall MySQL, follow the steps presented here to completely uninstall it.

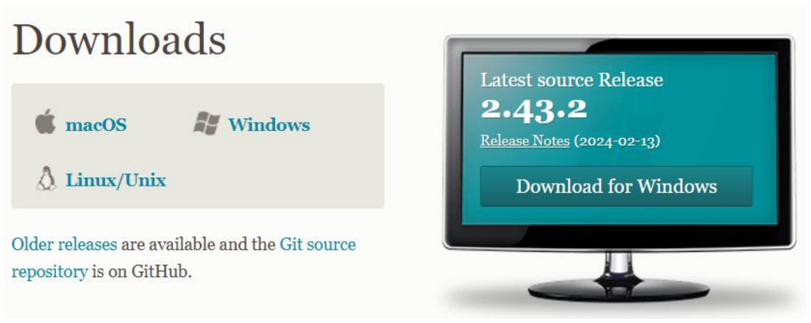
4 Git and Maven Projects

Modern software projects development requires effort from large teams of developers that have to collaborate in order to integrate their work and create the final product. Coordinating and tracking the changes between multiple source files is a difficult task, thus an automatic tool was developed in 2005, initially for the development of the Linux Kernel, and, since then, it has penetrated all levels of software development. The tool is a version control system (VCS), named GIT, which tracks changes of computer files and helps coordinating several people who work on those files.

Furthermore, large applications often encounter problems with the software build settings as well as with the dependency description. To address these problems, another automatic tool was created, MAVEN that defines conventions for the build procedure and uses an XML file to describe the software project, dependencies, external modules, components and plug-ins.

4.1 Git installation

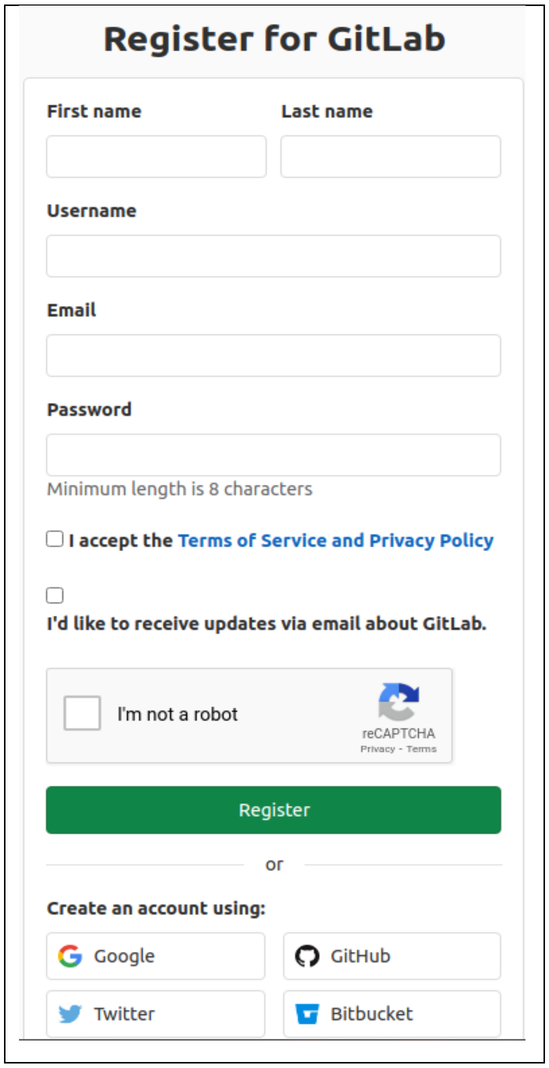
1) Click on https://git-scm.com/downloads.
2) Select your operating system.


Downloads
macOS
Windows
Linux/Unix
Older releases are available and the Git source repository is on GitHub.
Latest source Release
2.43.2
Release Notes (2024-02-13)
Download for Windows

3) If you select Windows, a file called *Git-version-64-bit.exe* should be downloaded. In the case you select another operating system or if your system is on 32 bits then a file with a similar name should be downloaded.
4) Click on this file and follow the default installation guidelines, except for the step where you are asked which terminal emulator you want to use. Select the second option.

4.2 Create Account on GitLab

1) Click on https://about.gitlab.com/.
2) In the right corner, click on Register. You will be asked to introduce your personal information. Or, if you already have an account, just *Sign In.*



3) In the next window, you can choose the role of “Software Developer”. In the next checkbox, you can choose the “Just me” option.
4) Following up, you will have to create a group. The name of the PRIVATE group must be of the format: *PT2024_GroupNumber_LastName_FirstName*

   (e.g. PT2024_30441_Popescu_Ioan)
5) For now, do not invite any other teammates to have access to the group, as we will do this later.

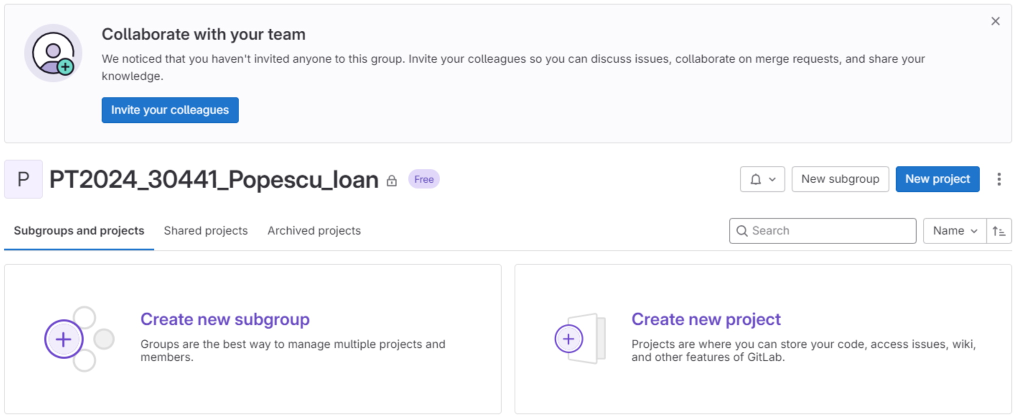
6) Click *Create group*.

If you already had an account and just signed in, you must create the group we have just talked about. Go to: *Groups ⟶ Your Groups ⟶ New Group* then create a new PRIVATE group with the format: *PT2024_GroupNumber_LastName_FirstName*

(e.g. PT2024_30441_Popescu_Ioan)

7) Now you must give access to your group, to the lab assistants. On your Group page, go to: *Manage ⟶Members ⟶ Invite Member ⟶* and offer **Maintainer** rights for the user: **utcn_dsrl** .

Inside the group, you can create your own projects for different applications of the PT lab. Click on *Create new project*.



Collaborate with your team

We noticed that you haven't invited anyone to this group. Invite your colleagues so you can discuss issues, collaborate on merge requests, and share your knowledge.

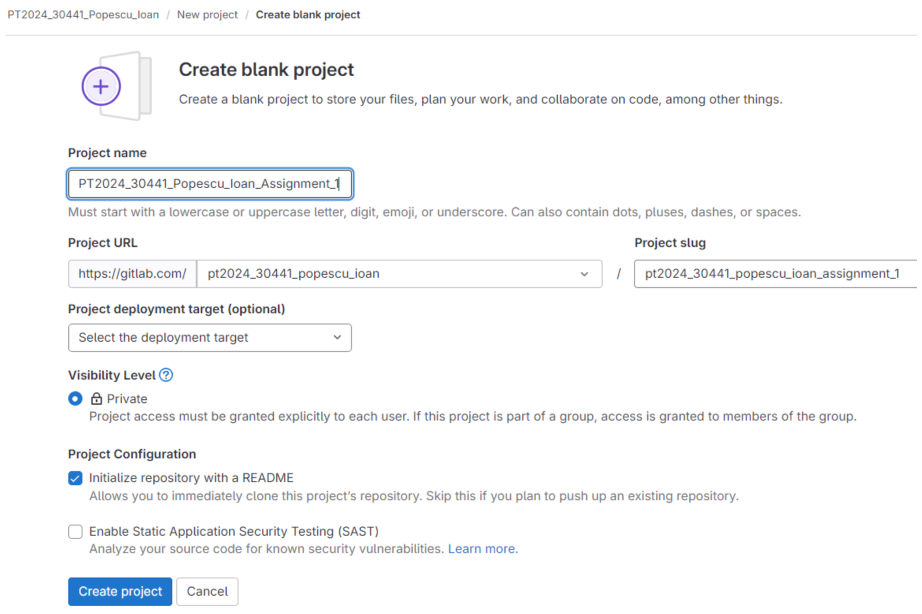
Create new subgroup

Groups are the best way to manage multiple projects and members.

Create new project

Projects are where you can store your code, access issues, wiki, and other features of GitLab.

Click on *Create blank project*. Remember to keep the same naming conventions for the projects.



PT2024_30441_Popescu_Ioan / New project / Create blank project

Create blank project

Create a blank project to store your files, plan your work, and collaborate on code, among other things.

Project name: PT2024_30441_Popescu_Ioan_Assignment_1

Must start with a lowercase or uppercase letter, digit, emoji, or underscore. Can also contain dots, pluses, dashes, or spaces.

Project URL: https://gitlab.com/ pt2024_30441_popescu_ioan / Project slug: pt2024_30441_popescu_ioan_assignment_1

Project deployment target (optional): Select the deployment target

Visibility Level

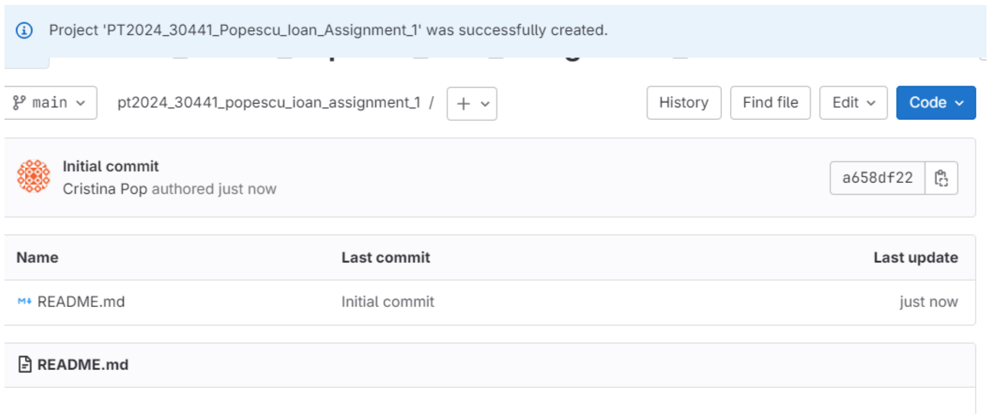
● Private
Project access must be granted explicitly to each user. If this project is part of a group, access is granted to members of the group.

Project Configuration

☑ Initialize repository with a README
Allows you to immediately clone this project's repository. Skip this if you plan to push up an existing repository.

☐ Enable Static Application Security Testing (SAST)
Analyze your source code for known security vulnerabilities. Learn more.

Click on *Create project*.



Project 'PT2024_30441_Popescu_Ioan_Assignment_1' was successfully created.

Initial commit
Cristina Pop authored just now
a658df22

Name	Last commit	Last update
README.md	Initial commit	just now

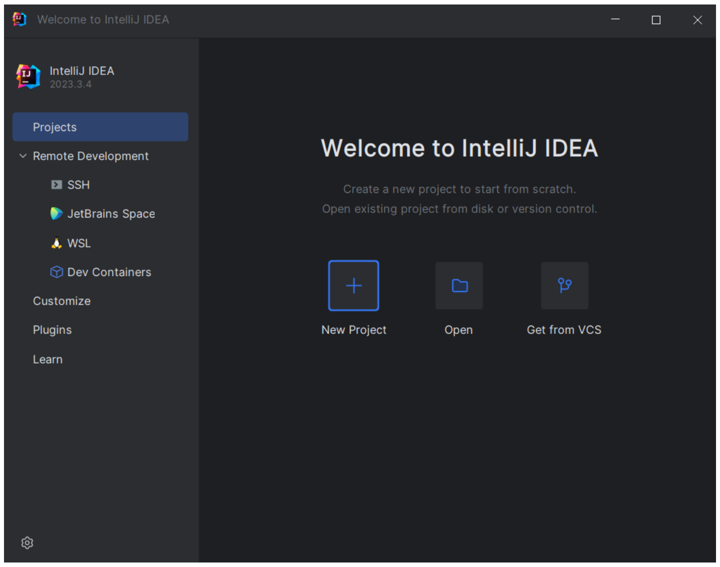
README.md

4.3 Basic Instructions

4.3.1 Create a project from scratch

1) Create the folder *PT2024_GroupNumber_LastName_FirstName* on *D:\*.

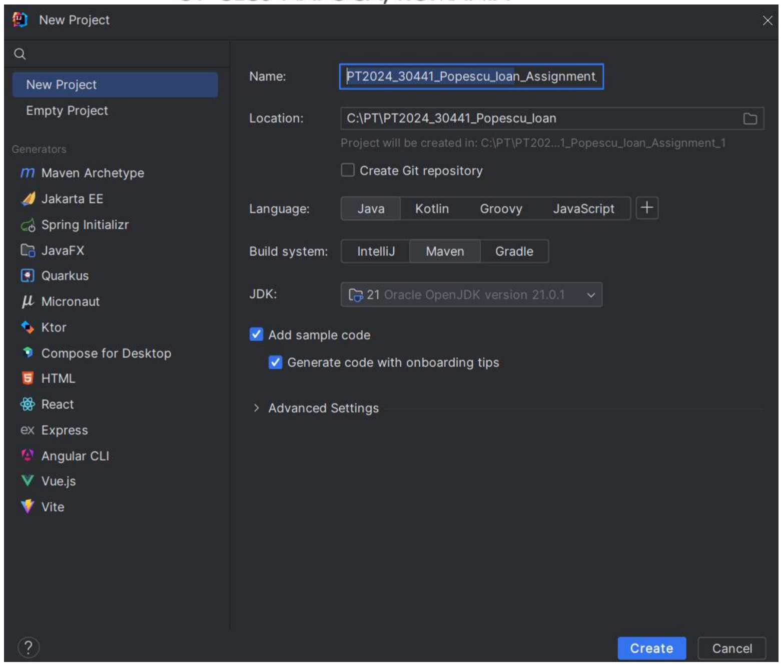
   Inside this folder create a new project. Open IntelliJ, click on *New Project.*



Then choose *Maven* from the list from the left, choose the appropriate Java SDK, then click on *Create*. Instead of using the default location suggested, use this one:

PT2024_GroupNumber_LastName_FirstName

Do not forget to also choose the appropriate name for your new Project, most likely using the format we requested (unless it is a test project, not for one of your assignments).


New Project

Name: PT2024_30441_Popescu_Ioan_Assignment

Location: C:\PT\PT2024_30441_Popescu_Ioan

Project will be created in: C:\PT\PT202...1_Popescu_Ioan_Assignment_1

Create Git repository

Language: Java | Kotlin | Groovy | JavaScript

Build system: IntelliJ | Maven | Gradle

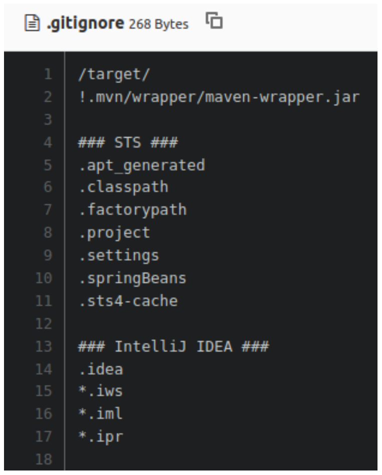
JDK: 21 Oracle OpenJDK version 21.0.1

[✓] Add sample code

[✓] Generate code with onboarding tips

2) Click *Create*.

3) Now, you must create a *.gitignore* file in the project if it is not already created, which tells git which files to ignore when committing and pushing to your remote projects. You don't want unnecessary files, such as IDE configuration files, to be pushed, because they are strictly relevant for your local system. Just create a file named "*.gitignore*" and write the following lines:


.gitignore 268 Bytes

```
/target/
!.mvn/wrapper/maven-wrapper.jar

### STS ###
.apt_generated
.classpath
.factorypath
.project
.settings
.springBeans
.sts4-cache

### IntelliJ IDEA ###
.idea
*.iws
*.iml
*.ipr

```

4) Right click on the folder *PT2024_GroupNumber_LastName_FirstName_Assignment_1* and click *Git Bash Here*.

5) Write the next commands:

a) git init

Note: the command will create a local repository with a default main branch. Notice that after executing this command, a hidden .git folder will be created inside the current folder representing the repository where git stores all necessary files. From now on, it will be possible to track all the changes that will be performed to the files from the original folder. The original folder is considered to be the working directory, while the .git folder is referred as the repository that tracks the made changes (for more details check this link).

***b)* git remote add origin** *https://gitlab.com/group_name/repository_name.git*

For example:
git remote add origin
https://gitlab.com/pt2024_30441_popescu_ioan/pt2024_30441_popescu_ioan_assignment_1.git

Note: this command will connect the local repository created using the "git init" command with the remote repository's origin (for more details check this link).

c) git add .

Note: the command will mark any changes that you have made to your project files (e.g. creating/modifying/deleting files) as staged so that they can be included in the next commit (for more details check this link).

d) git commit –a –m "initial commit"

Note: the command commits any files you have added with the git add command and commits any files you have changed since then – at this step the changes are saved only locally. An explanation message is given in order to document what has been added/changed (for more details check this link).

e) git branch -M main

f) git push –uf origin main

Note: the command sends the committed changes to your remote repository (for more details check this link).

4.3.2 Update the project

1) Create a new class named *App* in the same package as the class *Main*.
2) Navigate to folder *PT2023_GroupNumber_LastName_FirstName_Assignment_Number*, *right click* and select *Git Bash Here*
3) Insert the next commands:

a) git add .

b) git commit –a –m "add new class"

c) git pull origin main

Note: the command fetches and merges changes on the remote server to your working directory (for more details check this link).

d) git push –u origin main

4) You can always see the modification that were not committed yet by using:

a) git status

4.3.3. Create and work with a new branch

The real value of working with git, is the power of *branches*. They allow multiple developers to work simultaneously on the same project, on different features, and then to merge all the new changes in the main branch.

1) **Create a branch production**

The first step towards working with branches, is to create a new branch. When you create a new branch, it will automatically be initialized with the currently existent code. Then, while inside that branch, all changes will be added only on that branch.

To create a new branch:

- pull all the changes from the remote project, to be up to date:
  - *git pull*
- create the branch on your local machine and switch directly to that branch:
  - *git checkout -b <branch-name>*
- push the newly created branch to the remote repository:
  - *git push origin <branch_name>*

An example to create the branch production is the following:

```
#git pull

#git checkout -b production

#git push origin production
```

2) **switch between branches**

When working with multiple branches, it is important to keep track of all the available branches, and to always know on which branch you currently are.

- To bring locally meta-data information about existing branches:
  - *git fetch --all*
- In order to see all existent branches:

   - *git branch -a*
- In order to see on which branch you currently are:
   - *git status*
- *in order to switch from a branch to another, use the above-mentioned command:*
   - *git checkout <branch-name>*
      - *if the branch with that name is already existent, it will just switch to that one, instead of creating a new one*

You can try to create a new branch, make some small change, then switch back to the main branch, and see that change is not present in the main branch.

3) **commit changes to new branch**

When making changes on a new branch, you must always commit and push them to the remote branch, just like working on main.

- First, make sure you are on the right branch:
   - *git status*
- Repeat the same process as if you were working with main. However, pay attention to the names:
   - *git add .*
   - *git commit -m "commit message"*
   - *git push -u origin <branch-name>*
- And now, your remote branch <branch-name> contains all the changes you have pushed.

4) **merge branch with main**

An important step when working with branches, is to always keep the main branch up to date with the latest **working and functional** code from your other branches. Merging two branches, as the name suggests, is the process of merging the code from two branches. If the branches contain changes on different parts of the code, the merge process will work instantly. If both branches contain changes on the same parts of code, git will require you to solve the conflicts: from the two modifications, you must choose the one which you want to remain in the final version.

DO NOT FORGET: do not merge code which is not working properly, or which is not tested, into main. The main branch must always contain the latest functional version of your project.

In order to merge two branches:

- git merge <branch-with-new-changes> <branch-to-be-updated>

For a more in-depth explanation of branches and how they can be manipulated to serve your needs, we suggest checking the following tutorial:

https://git-scm.com/book/en/v2/Git-Branching-Basic-Branching-and-Merging

4.3.4. Getting git to work with a proxy server

1) In the UTCN laboratories you need to set the proxy server to use GIT bash
2) Open Git Bash
3) Insert the following commands:
   a) git config --global http.proxy http://proxy.utcluj.ro:3128
   b) git config –global --get http.proxy
4) In order to unset the proxy, use the following command:
   a) git config --global --unset http.proxy

4.3.5. Getting MAVEN to work with a proxy server

1) In the UTCN laboratories you need to set the proxy server in order to use MAVEN projects
2) Go to Windows Explorer-> Drive C-> Users -> *Your User* -> .m2
3) Create the folder **conf**
4) Go to conf folder and create the file **settings.xml** with the following content:

```
<settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0
                      http://maven.apache.org/xsd/settings-1.0.0.xsd">
  <localRepository/>
  <interactiveMode/>
  <usePluginRegistry/>
  <offline/>
  <pluginGroups/>
  <servers/>
  <mirrors/>
  <proxies>
    <proxy>
      <id>myproxy</id>
      <active>true</active>
      <protocol>http</protocol>
      <host>proxy.utcluj.ro</host>
      <port>3128</port>
      <username></username>
      <password></password>
    <nonProxyHosts>localhost,127.0.0.1</nonProxyHosts>
   </proxy>
  </proxies>
  <profiles/>
  <activeProfiles/>
</settings>
```

5) Go back to folder **.m2**
6) Delete the folder **repository**
7) For Eclipse
    a) Open **Eclipse**
    b) Go to **Window->|Preferences->|Maven->|User Settings**
    c) At the **User Settings** tab browse for the **settings.xml** file created at step 4
    d) Click **Apply** and **OK**
    e) Go on the project, right click and go to **Maven->Update Project**
8) For IntelliJ IDEA
    a) Open IntelliJ IDEA
    b) Go to **File->|Settings->|Build, Execution, Deployment->|Build Tools->|Maven**
    c) In the **User Settings file** field browse for the **settings.xml** file created at step 4.
    d) Click **Apply** and **OK**