

## Výhody spojené s používáním testovacích nástrojů

Testeři potřebují nástroje z několika důvodů, které usnadňují a zefektivňují proces testování softwaru. Zde jsou hlavní důvody, proč jsou nástroje pro testery důležité:

### Efektivita a úspora času

- **Automatizace:** Nástroje pro automatizaci testů umožňují rychlé provádění opakovaných testů, což šetří čas a úsilí testerů.
- **Zrychlení procesu:** S automatizovanými testy lze provádět stovky nebo tisíce testů během krátkého času, což urychluje cyklus vývoje.

### Přesnost a konzistence

- **Minimalizace lidských chyb:** Automatizované testy snižují riziko chyb způsobených lidským faktorem.
- **Konzistentní výsledky:** Nástroje zajišťují, že testy jsou prováděny stejným způsobem pokaždé, což zvyšuje spolehlivost výsledků.

### Pokročilé analýzy a reportování

- **Detailní reporty:** Nástroje často nabízejí pokročilé funkce pro generování reportů a analýzu výsledků testů, což umožňuje snadnější sledování pokroku a identifikaci problémů.
- **Vizualizace dat:** Umožňují zobrazovat výsledky testů graficky, což usnadňuje interpretaci a komunikaci s ostatními členy týmu.

### Sledování chyb a řízení kvality

- **Efektivní správa chyb:** Nástroje pro správu chyb umožňují testerům sledovat a dokumentovat nalezené problémy, což usnadňuje jejich řešení.
- **Kontrola kvality:** Pomáhají zajistit, aby software splňoval požadované standardy kvality.

### Podpora týmové spolupráce

- **Sdílení informací:** Nástroje umožňují testerům snadno sdílet výsledky, poznámky a dokumentaci s ostatními členy týmu, což podporuje spolupráci.
- **Integrace s dalšími nástroji:** Mnoho testovacích nástrojů se integruje s dalšími nástroji pro správu projektů, CI/CD a DevOps, což usnadňuje celý proces vývoje.

### Flexibilita a rozšiřitelnost

- **Podpora různých typů testů:** Nástroje mohou podporovat různé typy testování, včetně funkčního, výkonnostního, bezpečnostního a uživatelského testování.
- **Možnost přizpůsobení:** Mnohé nástroje umožňují testerům přizpůsobit testovací scénáře a skripty podle specifických potřeb projektu.

### Zvládání složitosti

- **Testování velkých systémů:** Moderní software je často složitý a zahrnuje mnoho komponent. Nástroje pomáhají testerům efektivně řídit testování těchto složitých systémů.
- **Podpora různých platforem:** Umožňují testovat aplikace na různých operačních systémech a zařízeních.

Tyto výhody ukazují, jak důležité jsou nástroje pro testy při zajišťování kvality softwaru, zvyšování efektivity a minimalizaci chyb. Umožňují testerům zaměřit se na důležitější aspekty testování, jako je analýza a optimalizace, místo aby se zabývali rutinními úkoly.

## Rizika související se zavedením testovacích nástrojů

### **Závislost na nástrojích**

- Přílišná závislost na automatizaci může vést k přehlížení manuálních testů, které mohou být stále důležité pro některé typy testování (např. uživatelské testování).

### **Vysoké počáteční náklady**

- Investice do pokročilých testovacích nástrojů a jejich integrace může být nákladná, což může být problém pro menší týmy nebo společnosti.

### **Složitost nástrojů**

- Některé nástroje mohou být složité na používání a vyžadují čas na školení a adaptaci, což může zpomalit proces testování v krátkodobém horizontu.

### **Údržba automatizovaných testů**

- Automatizované testy vyžadují pravidelnou údržbu a aktualizaci, aby odpovídaly změnám v aplikaci, což může být časově náročné.

### **Nedostatek flexibility**

- Některé testovací nástroje nemusí být dostatečně flexibilní, aby splnily specifické potřeby projektu, což může vést k dodatečným nákladům na přizpůsobení nebo výběr alternativních nástrojů.

### **Riziko falešně pozitivních/negativních výsledků**

- Automatizované testy mohou generovat falešně pozitivní nebo negativní výsledky, což může vést k chybným závěrům o kvalitě produktu.

### **Problémy s integrací**

- Někdy mohou nastat problémy při integraci testovacích nástrojů s jinými systémy a nástroji, což může zkomplikovat pracovní procesy.

# Nástroje pro řízení projektu

## Jira



Jira je nástroj od společnosti Atlassian, který je určen pro řízení projektů, sledování chyb (bug tracking) a správu úkolů. Původně byla vytvořena jako nástroj pro sledování chyb a požadavků v oblasti vývoje softwaru, ale díky své flexibilitě a rozšiřitelnosti je nyní široce využívána napříč různými typy projektů, včetně vývoje softwaru, IT a marketingu.

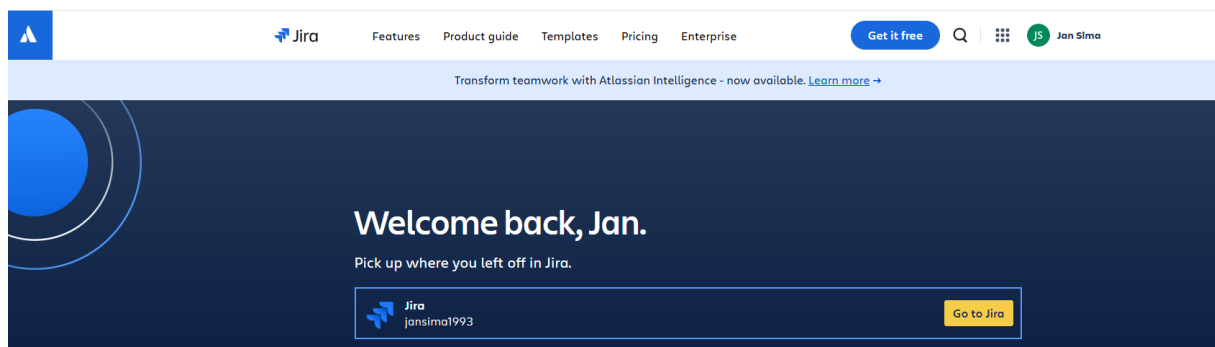
## GIT – Návod a instalace

### Jira - Registrace

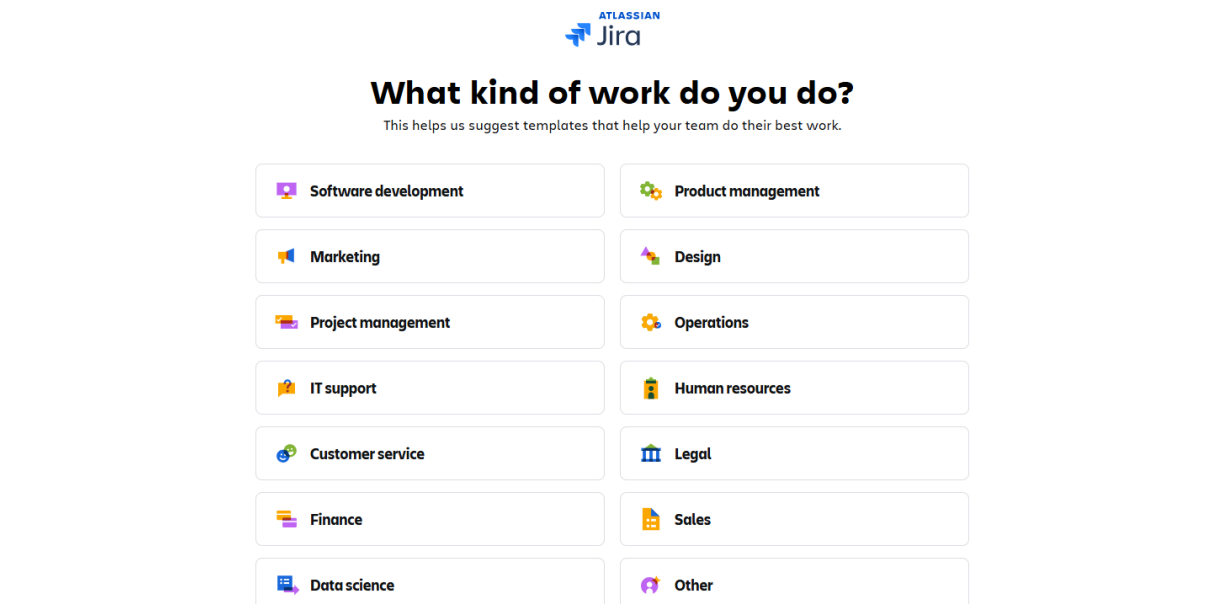
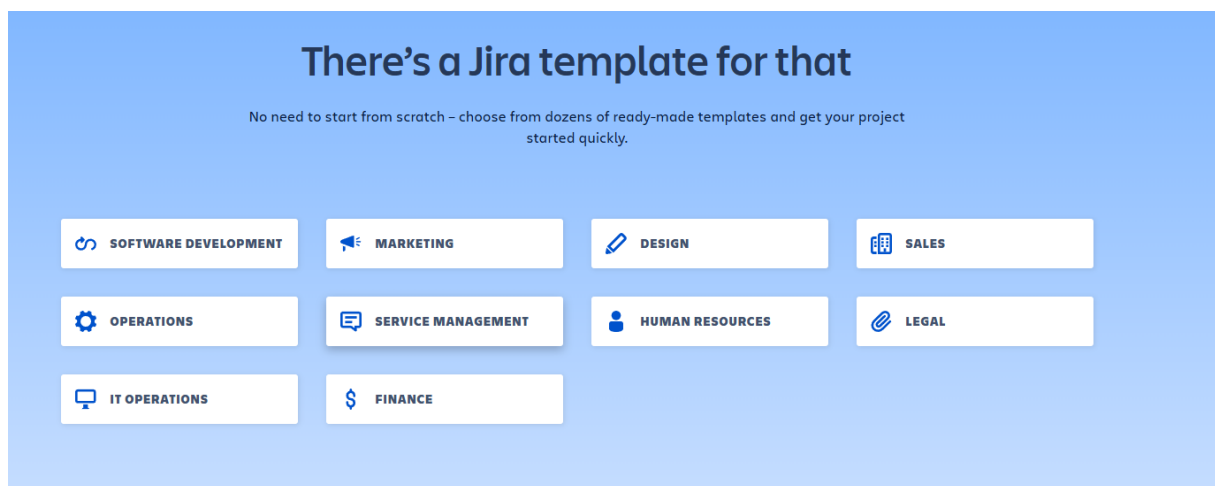
- Otevřete stránku <https://www.atlassian.com/software/jira>.
- Klikněte na Sign In. Vpravo nahoře.
- Ve vyskocovém okně klikněte na Vytvořit účet:

- Zadejte svůj email. Na ten vám poté přijde potvrzovací kód který zadáte. Poté vyberte jméno a heslo pro váš účet:

- Po registraci klikněte na záložku templates a vyberte záložku Templates:




- Vyberte template Software Development. Formát výběru se může lišit podle toho kde se v JIRA nacházíte. Proto přikládám dva screenshots:



- Na další záložce vyberte první možnost Scrum

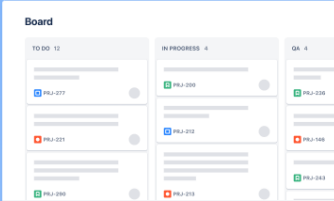
# Software development templates

From the first sprint to the retrospective, Jira's customizable templates make it easy to take software development projects from ideation to creation.




**Scrum**

Visualize, track, and manage your work easily from sprint to sprint.



**Kanban**

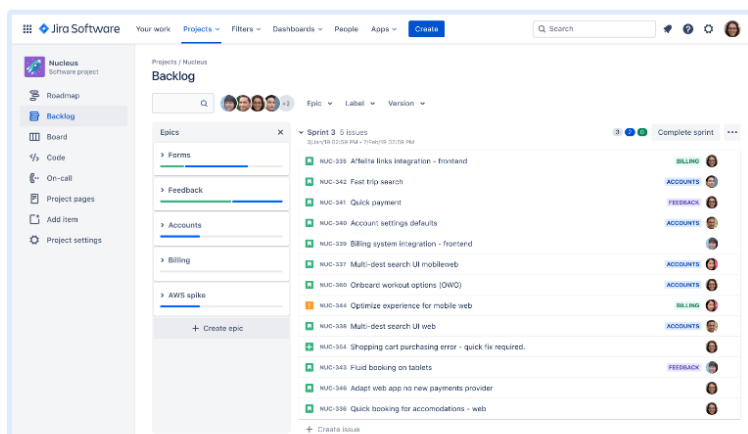
Manage a continuous delivery of work on a powerful board.



**Bug tracking**

Capture, track, and resolve bugs quickly.

- Klikněte na tlačítko Use free Scrum template:




## Scrum template


Easily plan, track, and manage work across sprints.

[Use free Scrum template](#)


BEST FOR

 Software development

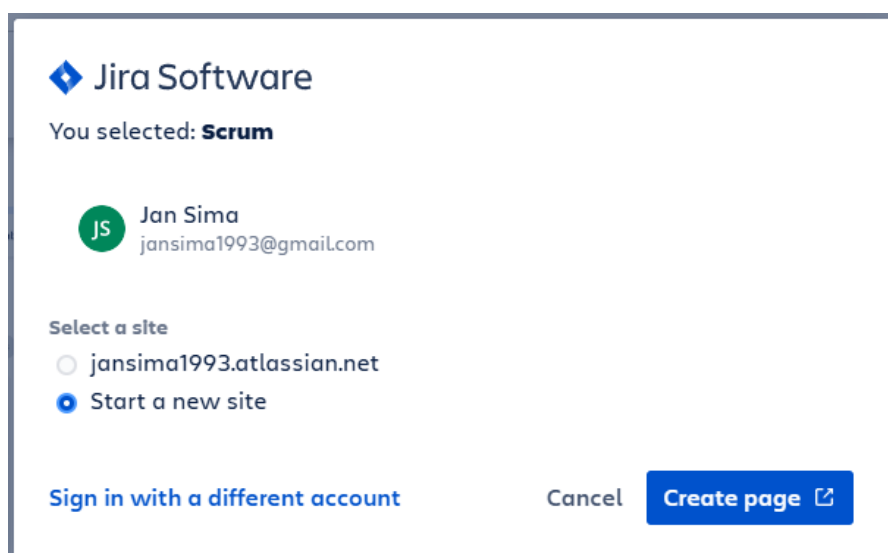
KEY FEATURES

 Sprint and task planning

 Progress tracking


 Sprint analytics tools

- Start a new site a poté klik na tlačítko Create page. Na další obrazovce vymyslete vlastní název stránky a klikněte na Agree and start now:



**Jira Software**

You selected: **Scrum**

 **Jan Sima**  
jansima1993@gmail.com

Select a site

☐ jansima1993.atlassian.net

☒ Start a new site

[Sign in with a different account](#) [Cancel](#) [Create page](#)

Work email

jansima1993@gmail.com

[Sign in with a different Atlassian account](#)

Your site


| .atlassian.net

I agree to the [Atlassian Customer Agreement](#), which incorporates by reference the [AI Product-Specific Terms](#), and acknowledge the [Privacy Policy](#).

Agree and start now

- Na dalším okně vyberte Select a team-managed project:

1 Project template




Scrum  
Jira

Change template

Sprint toward your project goals with a board, backlog, and timeline.

2 Choose a project type


 You'll need to create a new project if you decide to switch project types later.

Team-managed

Set up and maintained by your team.

For teams who want to control their own working processes and practices in a self-contained space. Mix and match agile features to support your team as you grow in size and complexity.

Simplified configuration




Select a team-managed project

Company-managed

Set up and maintained by your Jira admins.

For teams who want to work with other teams across many projects in a standard way. Encourage and promote organizational best practices and processes through a shared configuration.

Expert configuration



Select a company-managed project

- Vyberte jméno projektu a klikněte na Next:

## Add project details

Explore what's possible when you collaborate with your team.  
Edit project details anytime in project settings.

Required fields are marked with an asterisk \*

Name \*

JmenoProjektu

**Access** Anyone with access to jansima1993 can access and administer this project. [Upgrade your plan](#) to customize project permissions.

Key 

JMEN

Template

[Change template](#)



**Scrum**  
 **Jira**

Sprint toward your project goals with a board, backlog, and timeline.

Type

[Change type](#)



**Team-managed**

Control your own working processes and practices in a self-contained space.

[Cancel](#)

[Next](#)

- Další stránka umožňuje napojit JIRA na další nástroje. V tuto chvíli necháme nevypněno a klikneme na Continue:

## Connect your work

Activate some of Jira's best features by connecting your team's code repositories, documentation spaces, and more.

### Code



GitHub



Connect repository

### Project pages



Confluence



Connect space

### Security



GitHub Security



Connect security container

[Continue](#)

- Po těchto operacích se vytvoří nový projekt. Při prvním spuštění budete vyzváni k přidání spolupracovníků (nemusíte přidávat nikoho). Spolupracovníky lze pak přidat i dalšími způsoby například klikem na tlačítko na Board:

## JMEN board

 Search


### Add people to this project

To help you create work, invite these admins from your other projects. They'll be given Administrator roles.

D

KK

JD

MP

Add other people

Add 4 people

TO DO

IN PROGRESS

**Get started in the backlog**

Plan and start a sprint to see issues here.

Go to Backlog

## Hlavní funkce a vlastnosti Jira

- **Správa projektů a úkolů:** Umožňuje vytváření projektů a v rámci nich i různých úkolů, které se mohou týkat funkcionalit, chyb, požadavků na změny nebo jakékoliv jiného typu úkolu. Každý úkol má vlastní životní cyklus (workflow), což umožňuje řídit stav od vzniku až po dokončení nebo vyřešení.
- **Sledování chyb (bug tracking):** Jira byla původně navržena pro sledování chyb ve vývoji softwaru, což z ní dělá silný nástroj pro tento účel. Umožňuje sledovat a spravovat všechny chyby, jejich stav a prioritu, což je klíčové pro tým vývojářů při zajištění kvality.
- **Workflow a procesy:** Podporuje nastavení vlastních workflow pro různé typy úkolů. Každý workflow může obsahovat různé kroky (stavy) a přechody, které odpovídají potřebám projektu a týmu. Workflows lze přizpůsobit pro různé procesy, ať už jde o agilní vývoj, tradiční waterfall přístupy nebo hybridní modely.
- **Agilní řízení:** Jira podporuje agilní metody řízení projektů, jako jsou Scrum a Kanban. Nabízí nástroje jako sprinty, backlogy, Kanbanové tabule, které týmy používají k organizaci práce. Vizualizace pomocí Kanban nebo Scrum boardů umožňuje snadno sledovat průběh práce a plánování iterací.
- **Podrobné reporty a přehledy:** Nabízí analytické nástroje a reporty, jako jsou burndown chart, velocity chart, sprint report, a další, což týmu umožňuje sledovat pokrok, identifikovat problémy a optimalizovat procesy. Umožňuje sledovat výkon týmu, časy jednotlivých úkolů, nebo identifikovat úzká místa v procesech.
- **Integrace s dalšími nástroji:** Jira se integruje s mnoha dalšími nástroji, jako jsou Confluence (pro správu dokumentace), Bitbucket nebo GitHub (pro správu verzí a zdrojového kódu), Slack, Microsoft Teams a další. Nabízí širokou podporu rozšíření, což umožňuje přidání dalších funkcí, jako jsou automatizace procesů nebo integrace s CI/CD nástroji.
- **Automatizace a rozšiřitelnost:** Umožňuje nastavení pravidel pro automatizaci běžných úkonů, jako jsou změny statusů, upozornění, nebo aktualizace polí na základě podmínek. Díky API je možné ji rozšiřovat a přizpůsobovat, což je výhodné pro týmy s unikátními požadavky.

## Typické scénáře použití

- **Správa chyb a incidentů:** Umožňuje efektivní sledování chyb a správu jejich opravy.



- **Agilní řízení projektů:** Týmy využívající Scrum nebo Kanban mohou s pomocí Jiry plánovat sprinty, sledovat průběh práce a zlepšovat procesy pomocí retrospektivních dat.
- **Správa požadavků a funkcionalit:** Jira umožňuje organizovat nové funkcionality, sledovat změny a ověřovat požadavky od zákazníků nebo uživatelů.

## Výhody a nevýhody Jira

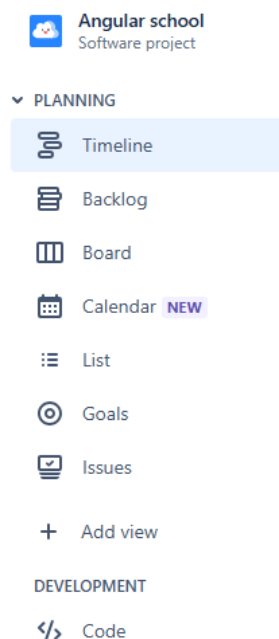
### Výhody:

- Flexibilní a přizpůsobitelná podle procesů a požadavků týmu.
- Široká integrace s dalšími nástroji.
- Podpora agilních metodik, vizuální Kanban a Scrum tabule.
- Robustní sledování chyb a historie změn v rámci úkolů.

### Nevýhody:

- Komplexita může být pro malé týmy nebo jednoduché projekty nadbytečná.
- Může být náročná na zavedení a školení.
- Cena může být vysoká pro větší týmy, pokud není využívána veškerá funkcionalita.

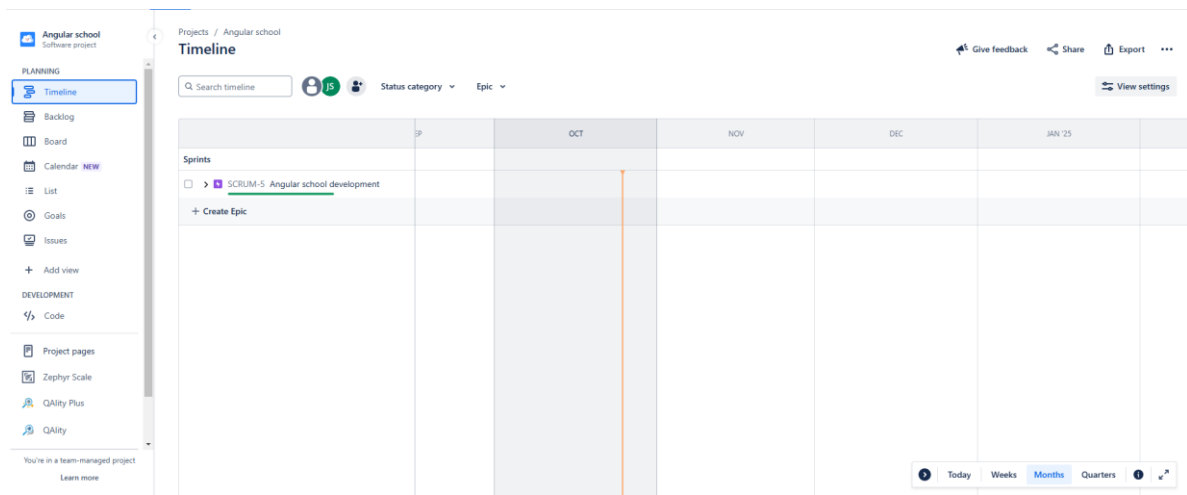
Jira je tedy silný nástroj pro organizaci práce, který usnadňuje komunikaci v týmech a umožňuje efektivní řízení projektů a úkolů.



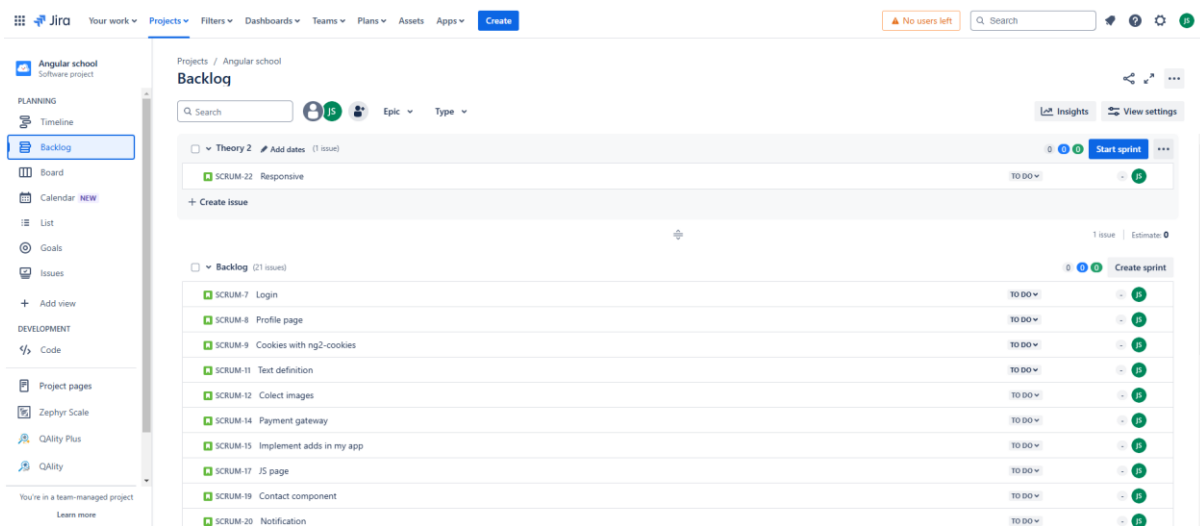
- **Timeline:** Umožňuje sledovat projekty a úkoly v čase. Na časové ose vidíš termíny, závislosti a postup jednotlivých úkolů, což pomáhá s plánováním a koordinací.
- **Backlog:** Seznam všech úkolů, které je potřeba dokončit, ale ještě nejsou přiřazeny ke sprintu nebo aktuální práci. Backlog slouží k organizaci a prioritizaci úkolů, než se přesunou na Board.
- **Board:** Vizualizace aktuální práce ve formě sloupců, obvykle pro zobrazení stavů úkolů (např. To Do, In Progress, Done). Pomáhá s přehledem o tom, kdo na čem pracuje a v jaké fázi úkoly jsou.

- **Calendar:** Zobrazuje úkoly a termíny v kalendářním formátu, což umožňuje lepší přehled o termínech a plánovaných aktivitách.
- **List:** Jednoduchý seznam všech úkolů v projektu, který poskytuje přehledný výpis všech položek, často s možností rychlých filtrů nebo úprav.
- **Goals:** Slouží k nastavování cílů nebo milníků v rámci projektu. Pomáhá týmu zůstat zaměřený na hlavní priority a sledovat postup vůči dlouhodobým cílům.
- **Issues:** Jde o základní jednotky práce v Jiře, reprezentující konkrétní úkoly, chyby nebo požadavky. Issues mohou mít různá pole (např. popis, priorita, termín dokončení) a jsou organizovány a sledovány v rámci projektu.

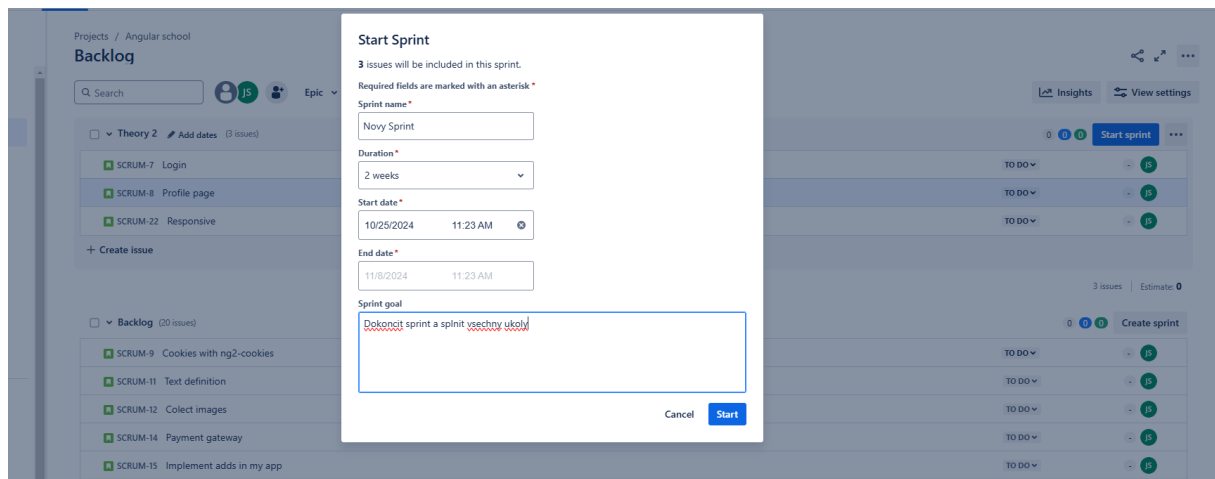
Příklad záložky Timeline. Zde se provedou časové odhady projektu a sleduje se časový postup na projektu:



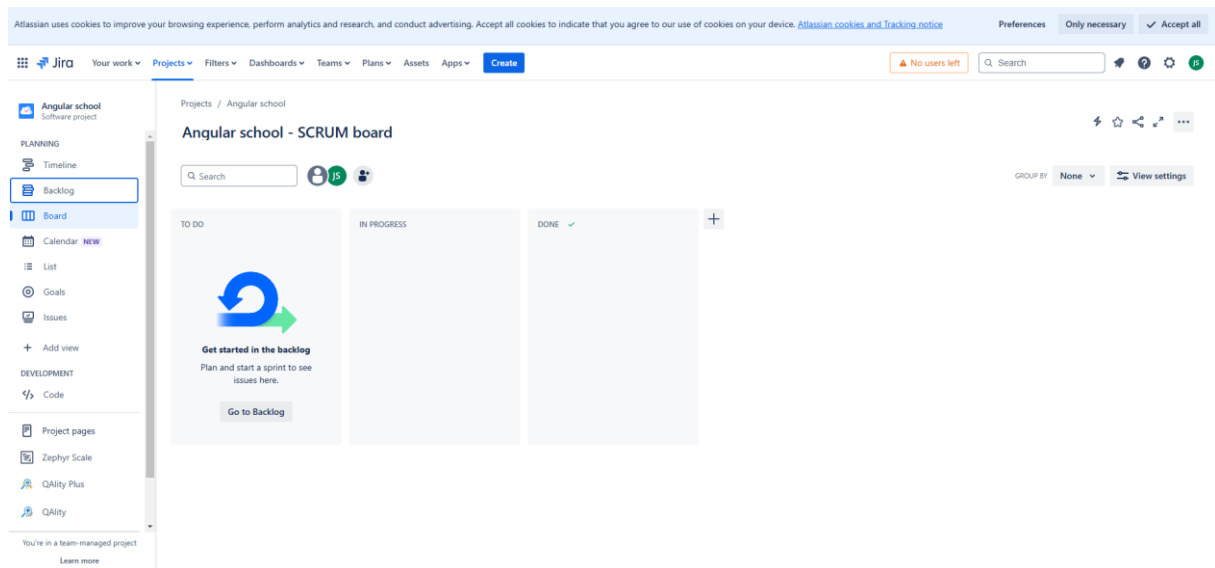
Příklad záložky Backlog. Jedna z nejdůležitějších komponent JIRA (společně s Board). Zde se nejlépe tvoří nové úkoly a plánují a zahajují Sprints (tlačítko Start sprint):



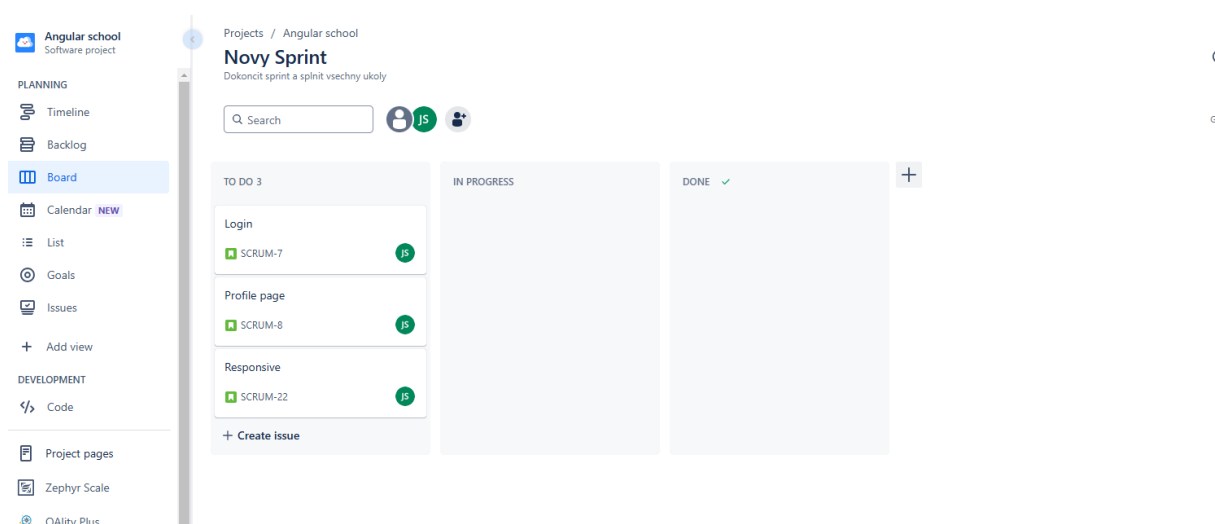
Po stisknutí tlačítka Start sprint se objeví toto vyskakovací okno. Lze upravit název, délku a cíl Sprintu:



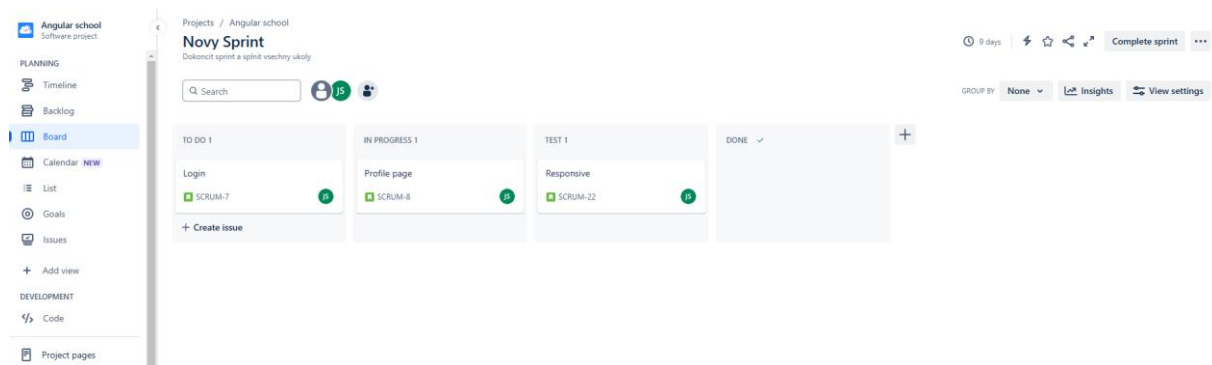
Příklad záložky Board. Jedna z nejdůležitějších komponent JIRA (společně s Backlog). Zde najdete stav úkolů pro současný Sprint:



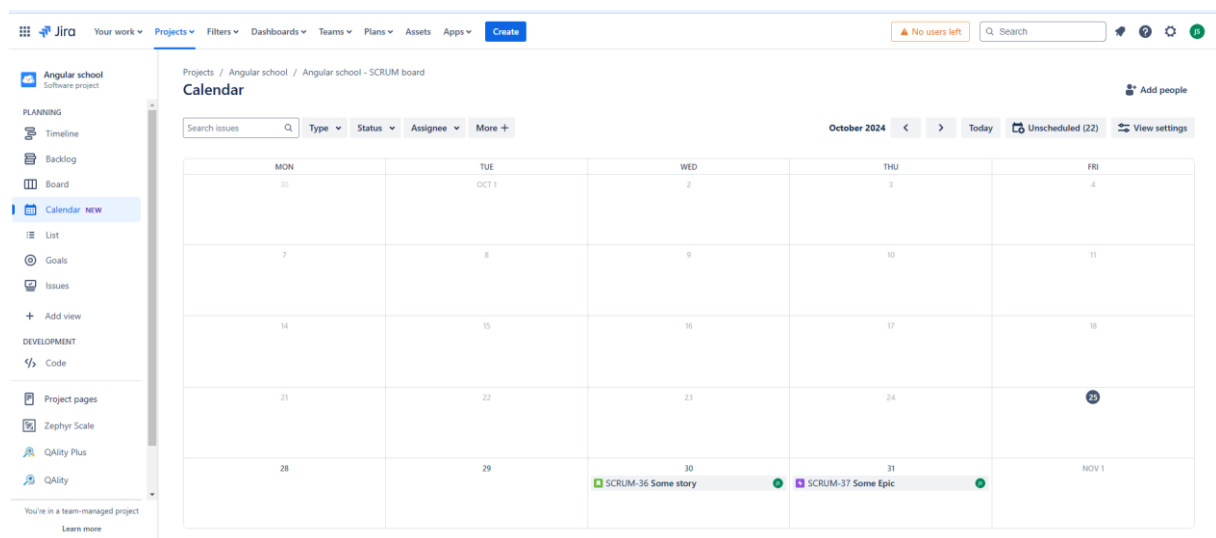
Po spuštění Sprintu (záložka Backlog). Se na Board objeví úkoly:



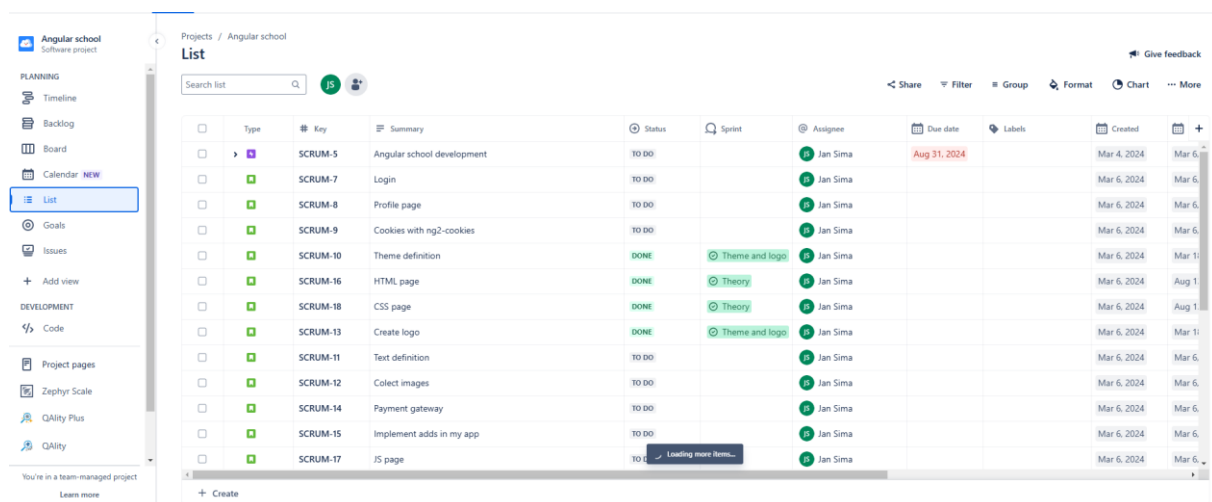
Tyto úkoly lze přesouvat mezi sloupci a měnit tak jejich stav. Defaultně jsou to sloupce To Do, In Progress a Done. Lze však přidat a odebírat další sloupce. V našem případě sloupec Test:



Příklad záložky Calendar. Zde najdete plánované aktivity a úkoly pro projekt:



Příklad záložky List. Zde seznam všech úkolů:



Příklad záložky Issues. Podobně jako List zde najdete seznam všech úkolů. Issues však umožňují jednodušší editaci těchto úkolů:

Angular school  
Software project

PLANNING

- Timeline
- Backlog
- Board
- Calendar **NEW**
- List
- Goals
- Issues**
- + Add view

DEVELOPMENT

- Code
- Project pages
- Zephyr Scale
- QAity Plus
- QAity

You're in a team-managed project  
Learn more

Projects / Angular school

Issues

Share Export Go to all issues LIST VIEW DETAIL VIEW

Search issues Project Angular school Type Status Assignee More + Save filter

Created 17

Some Epic

SCRUM-37

Some story

SCRUM-36

PokusBug

SCRUM-35

Test test case

SCRUM-34

Test - TC001

SCRUM-33

Create forum page

SCRUM-30

Component for coding in browser

SCRUM-29

SEO

SCRUM-28

1-33 of 33

Add epic / SCRUM-34

+ Add

Apps

Description

Add a description...

Linked issues

tests

SCRUM-33 Test - TC001

TO DO

Activity

Show: All Comments History

Newest first 17

15

Add a comment...

Looks good! Need help? This is blocked... Can you clarify...? This is on track

Pro tips press M to comment

4 of 33

To Do

Actions

Details

Assignee

Jan Sima

Labels

None

Parent

None

Team

None

Story point estimate

None

Sprint

None

Development



**Azure DevOps** je sada nástrojů od společnosti Microsoft, která slouží pro kompletní řízení životního cyklu vývoje softwaru a správu DevOps procesů. Umožňuje spravovat projekty, organizovat práci týmu, a provádět integraci a nasazování aplikací (CI/CD). Azure DevOps je silně integrovaný s dalšími nástroji od Microsoftu, ale je také velmi flexibilní a otevřený pro integrace s jinými systémy.

### Hlavní funkce a vlastnosti Azure DevOps

#### Boards (Tabule):

- Umožňuje spravovat a sledovat úkoly a pracovní procesy pomocí **Scrum** nebo **Kanbanových** boards, backlogů a plánování sprintů.
- Obsahuje nástroje pro řízení práce, jako jsou uživatelské příběhy (user stories), úkoly, chyby a epiky.
- Umožňuje přiřazení úkolů, sledování jejich stavu a řízení workflow podobně jako Jira.

#### Repos (Repozitáře):

- Nabízí hostování Git repozitářů, kde týmy mohou spravovat zdrojový kód, sledovat změny, vytvářet pull requesty a provádět code reviews.
- Podpora pro **branchování** a **správu verzí** usnadňuje spolupráci v týmu, zejména v kombinaci s CI/CD.
- Je integrovaný s **GitHubem** a **GitHub Actions**, což umožňuje sdílení kódu mezi různými platformami.

#### Pipelines (Pipeliny):

- Automatizované CI/CD pipelines umožňují kontinuální integraci (Continuous Integration) a nasazení (Continuous Deployment) aplikací.
- Podporuje vytváření a provozování build a release pipeline pro různé platformy, včetně Docker, Kubernetes, Azure Cloud a dalších cloudových služeb.
- Pipelines mohou být nakonfigurovány pomocí YAML souborů nebo vizuálního editoru, což poskytuje flexibilitu pro různé týmy a projekty.

#### Test Plans (Testovací plány):

- Umožňuje vytvářet, spravovat a spouštět manuální i automatizované testovací plány.
- Podporuje psaní testovacích případů a scénářů, zaznamenávání výsledků a vytváření reportů z testů.
- Pomáhá zajistit kvalitu kódu před jeho nasazením do produkčního prostředí.

#### Artifacts (Artefakty):

- Služba pro správu balíčků, která umožňuje ukládání a distribuci balíčků, jako jsou NuGet, npm, Maven nebo Python balíčky.

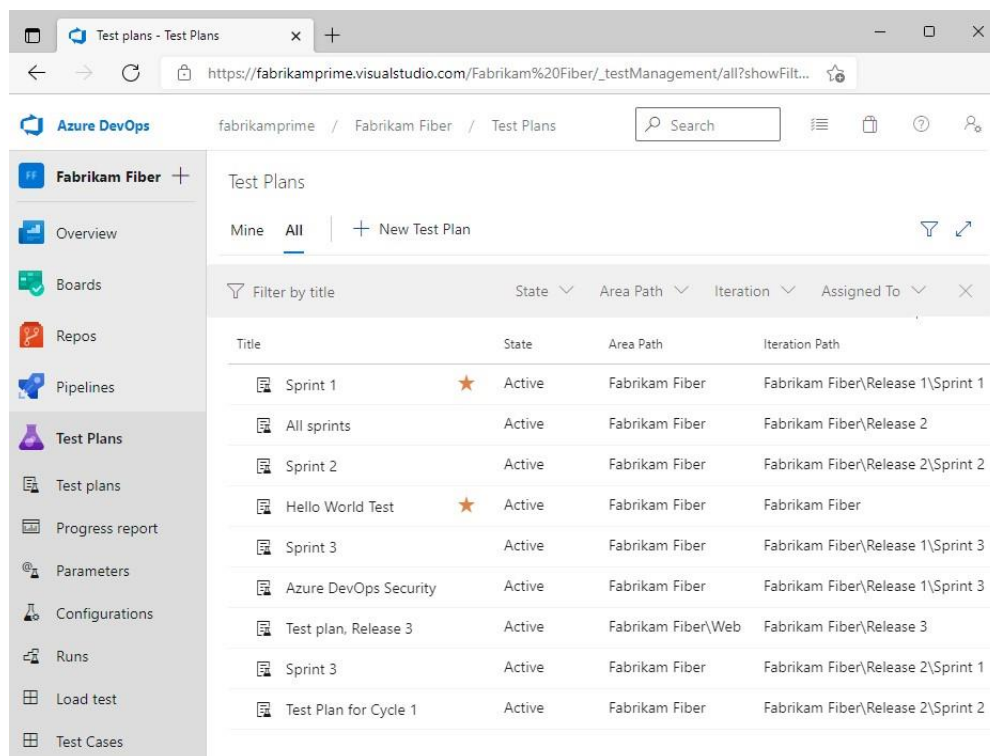
- Integruje se s CI/CD pipelines, takže artefakty z buildů mohou být automaticky přístupné a nasazeny v dalších procesech.
- Pomáhá týmu efektivně spravovat knihovny a závislosti přímo v rámci DevOps prostředí.

#### Sledování stavu a reporty:

- Azure DevOps nabízí analytické a reportovací nástroje pro sledování pokroku a měření výkonu týmu.
- **Analytics Service** umožňuje vytvářet detailní přehledy o pracovním postupu, efektivitě CI/CD procesů, a testování.
- Pomocí nástrojů jako Power BI je možné vytvářet vlastní interaktivní reporty.

#### Typické scénáře použití

- **Správa vývoje a DevOps procesů:** Azure DevOps umožňuje týmům spravovat celý proces vývoje, od plánování přes psaní kódu až po testování a nasazení.
- **Automatizované CI/CD:** Týmy mohou vytvářet a spravovat automatizované build a release procesy, což pomáhá zrychlit a stabilizovat nasazování aplikací.
- **Plánování a sledování úkolů:** Azure Boards poskytují výkonné nástroje pro plánování sprintů, správu backlogů a sledování práce, což pomáhá týmům dodržet termíny a zlepšuje transparentnost.
- **Správa balíčků:** Azure Artifacts umožňuje spravovat závislosti a knihovny v rámci DevOps procesů, čímž usnadňuje spolupráci na velkých projektech.



The screenshot shows the 'Test Plans' section in Azure DevOps for the 'Fabrikam Fiber' project. The interface includes a sidebar with navigation options like Overview, Boards, Repos, Pipelines, Test Plans, Test plans, Progress report, Parameters, Configurations, Runs, Load test, and Test Cases. The main area displays a table of test plans with columns for Title, State, Area Path, and Iteration Path. The table lists several test plans, including 'Sprint 1', 'All sprints', 'Sprint 2', 'Hello World Test', 'Sprint 3', 'Azure DevOps Security', 'Test plan, Release 3', 'Sprint 3', and 'Test Plan for Cycle 1'. All listed test plans are in an 'Active' state.

Title	State	Area Path	Iteration Path
Sprint 1	Active	Fabrikam Fiber	Fabrikam Fiber\Release 1\Sprint 1
All sprints	Active	Fabrikam Fiber	Fabrikam Fiber\Release 2
Sprint 2	Active	Fabrikam Fiber	Fabrikam Fiber\Release 2\Sprint 2
Hello World Test	Active	Fabrikam Fiber	Fabrikam Fiber
Sprint 3	Active	Fabrikam Fiber	Fabrikam Fiber\Release 1\Sprint 3
Azure DevOps Security	Active	Fabrikam Fiber	Fabrikam Fiber\Release 1\Sprint 3
Test plan, Release 3	Active	Fabrikam Fiber\Web	Fabrikam Fiber\Release 3
Sprint 3	Active	Fabrikam Fiber	Fabrikam Fiber\Release 2\Sprint 1
Test Plan for Cycle 1	Active	Fabrikam Fiber	Fabrikam Fiber\Release 2\Sprint 2

#### Výhody a nevýhody Azure DevOps

##### Výhody:

- **Kompletní DevOps řešení:** Nabízí vše potřebné pro řízení projektů, správu kódu, automatizaci buildů a nasazení i testování v jednom balíčku.
- **Integrace s Azure Cloud:** Skvěle spolupracuje s dalšími produkty Microsoftu, zejména s Azure Cloud, což je výhodné pro týmy, které již Azure používají.

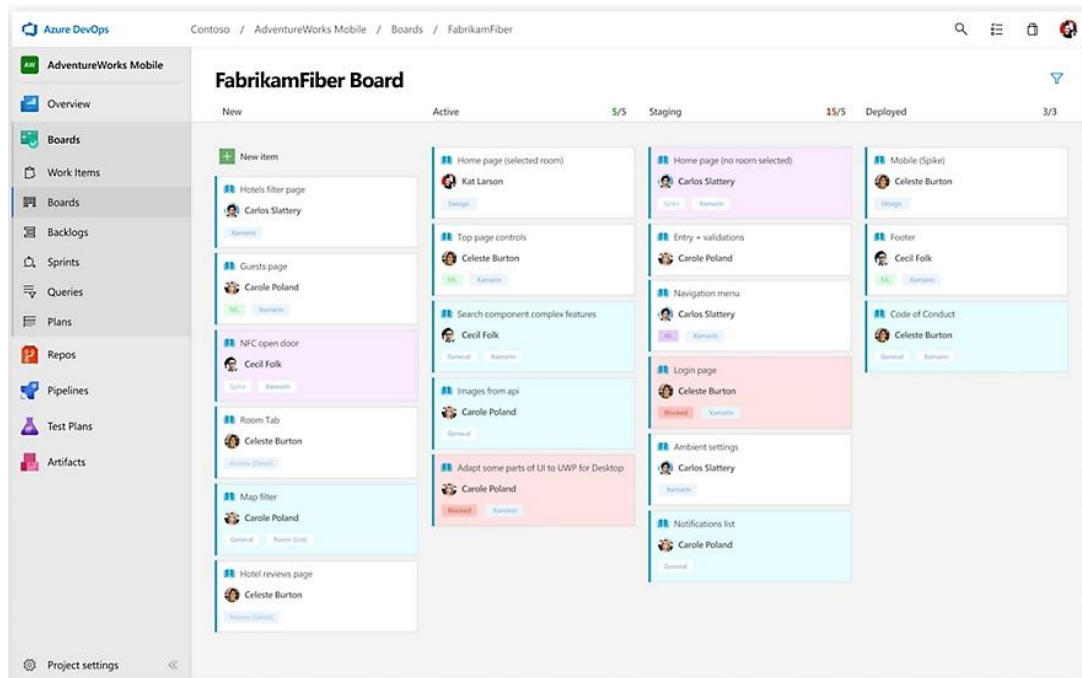
- **Flexibilita a otevřenost:** Podporuje integrace s externími nástroji a platformami, jako je GitHub, Jenkins nebo Docker.
- **Škálovatelnost:** Dobře funguje pro malé i velké týmy, nabízí robustní možnosti řízení přístupu a oprávnění.

### Nevýhody:

- **Komplexita:** Celé prostředí může být složité pro týmy, které s DevOps teprve začínají.
- **Závislost na Microsoftu:** I když je možné integrovat s jinými platformami, některé funkce mohou být optimalizované spíše pro uživatele Microsoft produktů.
- **Náklady:** Při růstu projektu nebo týmu mohou náklady na využívání Azure DevOps narůstat, zejména při využití většího množství funkcí a zdrojů v cloudu.

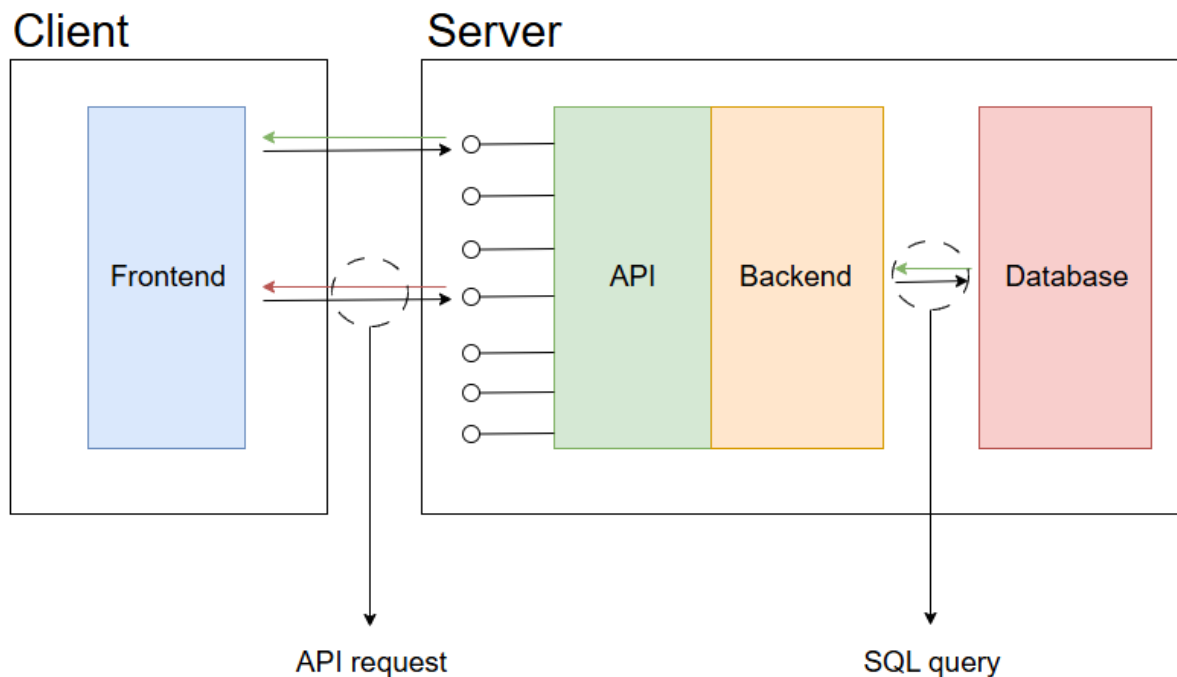
Azure DevOps je silný nástroj pro řízení projektů a DevOps procesů, který poskytuje komplexní řešení pro vývojářské týmy a usnadňuje automatizaci procesů a správu DevOps cyklu.

Příklad boardu v Azure DevOps:





## Frontend, Backend, API, databáze. Jak to spolu funguje?



- **Frontend:** Představuje část aplikace, kterou uživatel přímo vidí a používá. Je to vizuální rozhraní vytvořené pomocí technologií jako HTML, CSS a JavaScript (např. Angular nebo React). Frontend zobrazuje data, která získává z backendu, a umožňuje uživatelům interakci.
- **Backend:** Je část aplikace, která běží na serveru a je zodpovědná za logiku aplikace, zpracování požadavků a správu dat. Backend zpracovává dotazy z frontendu, komunikuje s databází a vrací zpět potřebná data. Typické backendové technologie zahrnují Node.js, Java, Python (Django), PHP nebo Ruby.
- **API (Application Programming Interface):** API slouží jako rozhraní, které umožňuje frontendu a backendu komunikovat. API definuje způsob, jakým frontend posílá požadavky na backend a jak backend odpovídá. Často se používají REST API pro předávání dat mezi frontendem a backendem.
- **Databáze:** Je úložiště, kde jsou data aplikace uložena. Backend komunikuje s databází, aby ukládal, načítal nebo upravoval data podle požadavků z frontendu. Mezi populární databáze patří MySQL, PostgreSQL, MongoDB nebo SQLite.

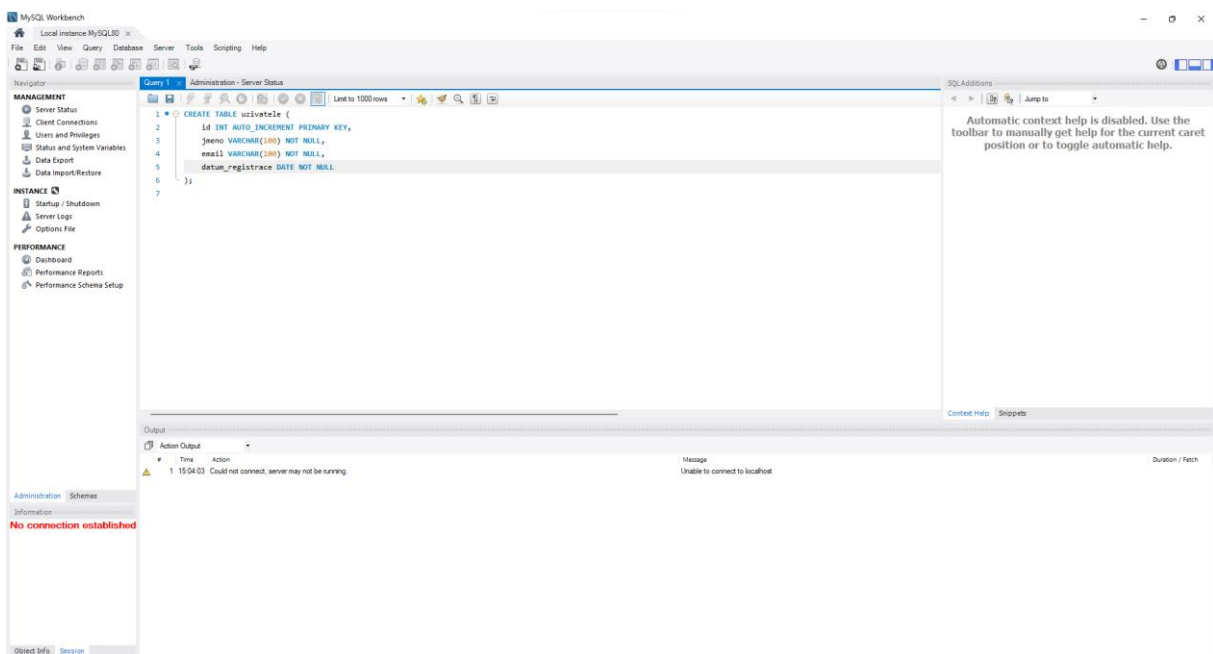
## Databázové nástroje

Databázové nástroje, jsou určeny k tomu, aby usnadnily správu a manipulaci s databázemi. Více bude probráno v modulu „Základy SQL“.

### MySQL Workbench



- **Popis:** MySQL Workbench je oficiální grafický nástroj pro správu a návrh databází MySQL.
- **Hlavní funkce:**
  - Návrh databáze: Umožňuje vizuální návrh databázových struktur a generování SQL skriptů.
  - Správa databází: Umožňuje spravovat schémata, tabulky, pohledy a další objekty.
  - Dotazování: Nabízí SQL editor pro psaní a provádění dotazů.
  - Monitoring výkonu: Umožňuje sledovat a optimalizovat výkon databází.
- **Využití:** Používá se pro správu a návrh databází MySQL, ideální pro vývojáře a administrátory databází.



### PhpMyAdmin



- **Popis:** phpMyAdmin je open-source webová aplikace pro správu MySQL databází.
- **Hlavní funkce:**
  - Správa databází: Umožňuje snadno vytvářet, upravovat a mazat databáze, tabulky a data.
  - Export a import dat: Podporuje import a export databází ve formátech jako SQL, CSV, a další.
  - Vykonávání SQL dotazů: Nabízí jednoduchý interface pro psaní a provádění SQL dotazů.
  - Uživatelské rozhraní: Intuitivní webové uživatelské rozhraní, které je snadné na ovládání.
- **Využití:** Ideální pro webové aplikace, kde je potřeba spravovat MySQL databáze pomocí webového rozhraní.



## SQL Server Management Studio

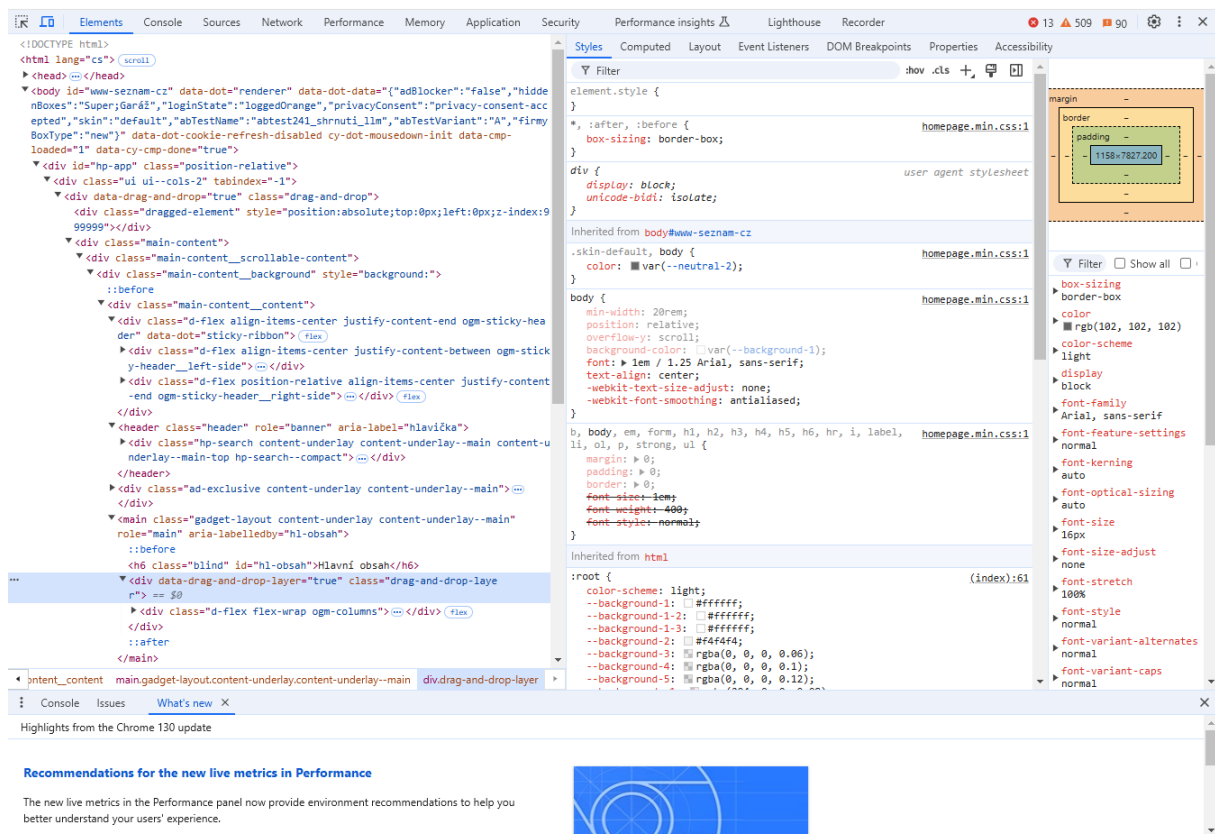


- **Popis:** SQL Server Management Studio je integrované prostředí pro správu a administraci Microsoft SQL Serveru.
- **Hlavní funkce:**
  - Správa databází: Umožňuje spravovat všechny aspekty SQL Serveru, včetně databází, schémat, tabulek a dalších objektů.
  - Tvorba a provádění dotazů: Poskytuje SQL editor pro psaní, provádění a ladění dotazů.
  - Návrh databáze: Umožňuje vizuálně navrhovat databázové struktury a generovat skripty.
  - Monitoring a ladění výkonu: Obsahuje nástroje pro monitorování výkonu a analýzu dotazů.
- **Využití:** Používá se především pro správu a administraci Microsoft SQL Serveru ve firemním prostředí.

## Vývojářské nástroje ve webovém prohlížeči

Vývojářské nástroje ve webovém prohlížeči (známé jako "DevTools") jsou integrované nástroje, které vývojářům pomáhají analyzovat, ladit a optimalizovat webové aplikace a stránky. Lze je otevřít zkratkou F12 nebo klikem pravým tlačítkem myši do stránky a vybrání možnosti „Prozkoumat“. Další funkce a možnosti vývojářských nástrojů budou probrány v modulu „Základy frontendu“.

Zpět	Alt+Klávesa šipka vlevo
Vpřed	Alt+Klávesa šipka vpravo
Načíst znovu	Ctrl+R
Uložit jako...	Ctrl+S
Tisk...	Ctrl+P
Odeslat...	
 Vyhledat pomocí Google Lens	
Otevřít v režimu čtení	
 Vytvořit QR kód pro tuto stránku	
Přeložit do jazyka čeština	
Zobrazit zdrojový kód stránky	Ctrl+U
Prozkoumat	



The screenshot displays the Google Chrome DevTools interface. The top bar shows the 'Elements' panel selected. The 'Elements' panel on the left shows the DOM tree with a selected element. The 'Styles' panel on the right shows the computed styles for the selected element. The 'Console' panel at the bottom shows the current log messages.

**Elements Panel:**

```

<!DOCTYPE html>
<html lang="cs">
  <head>
    <body id="www-seznam-cz" data-dot="renderer" data-dot-data="{"adBlocker":false,"hideBoxes":true,"loginState":true,"loggedOn":true,"privacyConsent":true,"accepted":true,"skin":"default","abTestName":"abtest241_shrnuti_11m","abTestVariant":"A","firmBoxType":"new"}" data-dot-cookie-refresh-disabled="true" data-dot-mousedown-init="true" data-cmp-loaded="1" data-cmp-done="true">
      <div id="hp-app" class="position-relative">
        <div class="ui ui--cols-2" tab-index="1">
          <div data-drag-and-drop="true" class="drag-and-drop">
            <div class="dragged-element" style="position:absolute;top:0px;left:0px;z-index:99999"></div>
          <div class="main-content">
            <div class="main-content__scrollable-content">
              <div class="main-content__background" style="background:>
                ::before
              <div class="main-content__content">
                <div class="d-flex align-items-center justify-content-end ogm-sticky-header" data-dot="sticky-ribbon">
                  <div class="d-flex align-items-center justify-content-between ogm-sticky-header__left-side"></div>
                  <div class="d-flex position-relative align-items-center justify-content-end ogm-sticky-header__right-side"></div>
                </div>
                <div class="header" role="banner" aria-label="hlavička">
                  <div class="hp-search content-underlay content-underlay--main content-underlay--main-top hp-search--compact"></div>
                </div>
                <div class="ad-exclusive content-underlay content-underlay--main"></div>
                <div class="main class=gadget-layout content-underlay content-underlay--main" role="main" aria-labelledby="hl-obsah">
                  ::before
                  <div class="blind" id="hl-obsah">Hlavní obsah</div>
                  <div data-drag-and-drop-layer="true" class="drag-and-drop-layer">
                    <div class="d-flex flex-wrap ogm-columns"></div>
                  </div>
                </div>
              </div>
            </div>
          </div>
        </div>
      </body>
    </html>
  
```

**Styles Panel:**

```

element.style {
  min-width: 20rem;
  position: relative;
  overflow: scroll;
  background-color: var(--background-1);
  font: 1em / 1.25 Arial, sans-serif;
  text-align: center;
  -webkit-text-size-adjust: none;
  -webkit-font-smoothing: antialiased;
}
body {
  min-width: 20rem;
  position: relative;
  overflow: scroll;
  background-color: var(--background-1);
  font: 1em / 1.25 Arial, sans-serif;
  text-align: center;
  -webkit-text-size-adjust: none;
  -webkit-font-smoothing: antialiased;
}
body, em, form, h1, h2, h3, h4, h5, h6, hr, i, label, li, ol, p, strong, ul {
  margin: 0;
  padding: 0;
  border: 0;
  font-size: 1em;
  font-weight: 400;
  font-style: normal;
}
:root {
  color-scheme: light;
  --background-1: #ffffff;
  --background-1-2: #ffffff;
  --background-1-3: #ffffff;
  --background-2: #f4f4f4;
  --background-3: rgba(0, 0, 0, 0.06);
  --background-4: rgba(0, 0, 0, 0.1);
  --background-5: rgba(0, 0, 0, 0.12);
}

```

**Console Panel:**

Highlights from the Chrome 130 update

**Recommendations for the new live metrics in Performance**

The new live metrics in the Performance panel now provide environment recommendations to help you better understand your users' experience.

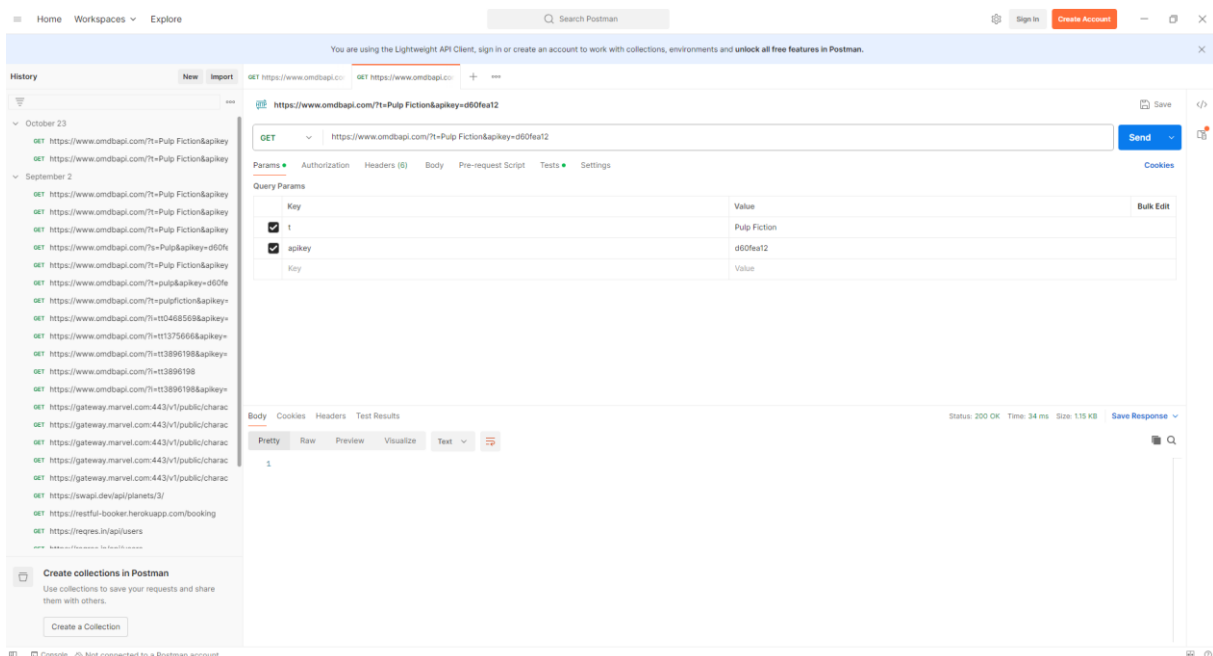
## Nástroje pro testování API

Testování API je klíčovou součástí vývoje softwaru, která zajišťuje, že API fungují podle očekávání a splňují požadavky. Více bude probráno v modulu „API testování – Postman“.

### Postman



- **Popis:** Postman je populární nástroj pro testování API, který umožňuje snadné vytváření, testování a dokumentaci API.
- **Hlavní funkce:**
  - Vytváření a odesílání různých typů HTTP požadavků (GET, POST, PUT, DELETE).
  - Podpora pro testování a skriptování pomocí JavaScriptu.
  - Možnost organizovat požadavky do kolekcí a sdílet je s ostatními.
  - Generování dokumentace pro API.




### Swagger



- **Popis:** Swagger je framework pro návrh a dokumentaci API. Umožňuje definovat API v formátu OpenAPI Specification (OAS).
- **Hlavní funkce:**
  - Generování interaktivní dokumentace pro API, kterou mohou vývojáři snadno procházet.
  - Možnost testovat API přímo z dokumentace.
  - Nástroje jako Swagger UI a Swagger Editor pro vizualizaci a editaci API specifikací.

Příklad Swaggeru: <https://petstore.swagger.io/>

 Swagger  
OpenAPI

Explore

## Swagger Petstore

1.0.7 OAS 2.0

[ Base URL: [petstore.swagger.io/v2](https://petstore.swagger.io/v2) ]  
<https://petstore.swagger.io/v2/swagger.json>

This is a sample server Petstore server. You can find out more about Swagger at <http://swagger.io> or on <https://twitter.com/strongloop>. For this sample, you can use the api key `special-key` to test the authorization filters.

[Terms of service](#)  
[Contact the developer](#)  
[Apache 2.0](#)  
[Find out more about Swagger](#)

Schemes

HTTPS

Authorize

**pet** Everything about your Pets

Find out more

POST

/pet/{petId}/uploadImage

uploads an image

POST

/pet

Add a new pet to the store

PUT

/pet

Update an existing pet

GET

/pet/findByStatus

Finds Pets by status

GET

/pet/findByTags

Finds Pets by tags

GET

/pet/{petId}

Find pet by ID

POST

/pet/{petId}

Updates a pet in the store with form data

# Nástroje pro verzování projektu

## GIT



Odkaz: <https://git-scm.com/downloads>

Git je distribuovaný systém pro správu verzí, který se široce používá při vývoji softwaru. Umožňuje programátorům a týmům efektivně sledovat změny v kódu, spravovat různé verze projektu a spolupracovat na vývoji. Zde je podrobnější popis Gitu:

## GIT – Návod a instalace

### Stažení a instalace Gitu

- Otevři stránky Git <https://git-scm.com/downloads>.
- Klikni na tlačítko **Download for Windows** (případně vyber svůj operační systém) a stáhni instalační soubor.
- Spust' stažený soubor a postupuj podle instalačního průvodce. Doporučuji ponechat výchozí nastavení. Obvykle je lepší nechat možnosti, které jsou přednastavené, zvláště pokud Git instaluješ poprvé.
- Po dokončení instalace můžeš Git spustit přes **Git Bash** nebo příkazový řádek **Command Prompt**.

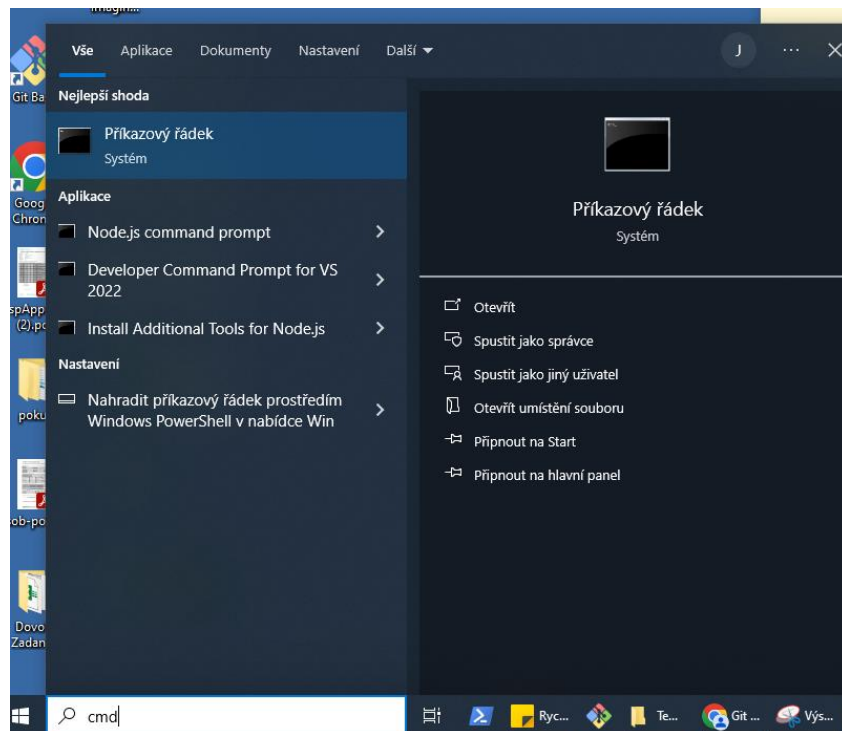


### Ověření instalace

- Otevři **Git Bash** nebo **Command Prompt** (zadej cmd do Windows vyhledávání).
- Zadej příkaz:

```
git --version
```

- Pokud vidíš verzi Gitu, instalace proběhla úspěšně.



```
Microsoft Windows [Version 10.0.19045.5011]
(c) Microsoft Corporation. Všechna práva vyhrazena.

C:\Users\simaja>git --version
git version 2.42.0.windows.2

C:\Users\simaja>
```

## Nastavení uživatelského jména a emailu

Pro spojení Gitu s GitHubem je potřeba nastavit jméno a email, které budou spojené s tvými commity:

- V Git Bash napiš následující příkazy (s tvými údaji):

```
git config --global user.name "TvojeJmeno"
git config --global user.email "tvuj@email.com"
```

```
Microsoft Windows [Version 10.0.19045.5011]
(c) Microsoft Corporation. Všechna práva vyhrazena.

C:\Users\simaja>git --version
git version 2.42.0.windows.2

C:\Users\simaja>git config --global user.name "TvojeJmeno"
```

```
Microsoft Windows [Version 10.0.19045.5011]
(c) Microsoft Corporation. Všechna práva vyhrazena.

C:\Users\simaja>git --version
git version 2.42.0.windows.2

C:\Users\simaja>git config --global user.email "tvuj@email.com"
```



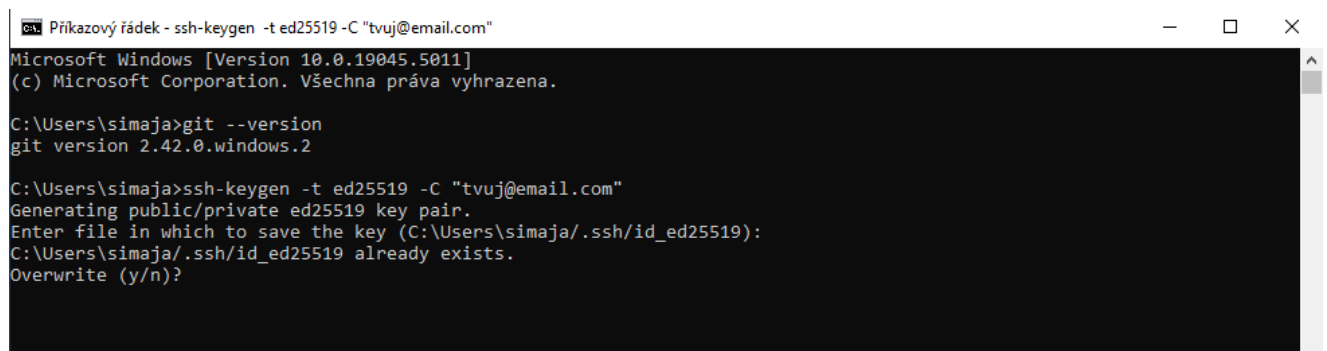
## Vytvoření SSH klíče

SSH klíč ti umožní bezpečné propojení s GitHubem bez nutnosti pokaždé zadávat heslo.

- V Git Bash zadej:

```
ssh-keygen -t ed25519 -C "tvuj@email.com"
```

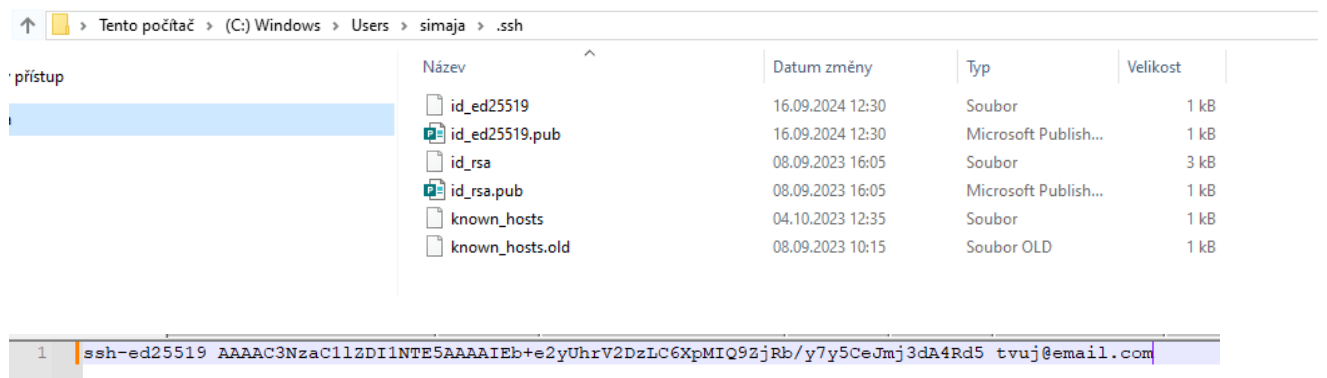
- Po zobrazení výzvy ke specifikaci cesty pro uložení stiskni **Enter** (pro výchozí cestu).
- Po zobrazení výzvy k zadání hesla můžeš stisknout **Enter** (nebo můžeš zadat heslo pro větší bezpečnost).
- SSH klíč najdeš v cestě C:\Users\<uzivatelskejmeno>\.ssh\id\_ed25519.pub. Tento klíč otevři v textovém editoru a zkopíruj jeho obsah.



```
Příkazový řádek - ssh-keygen -t ed25519 -C "tvuj@email.com"
Microsoft Windows [Version 10.0.19045.5011]
(c) Microsoft Corporation. Všechna práva vyhrazena.

C:\Users\simaja>git --version
git version 2.42.0.windows.2

C:\Users\simaja>ssh-keygen -t ed25519 -C "tvuj@email.com"
Generating public/private ed25519 key pair.
Enter file in which to save the key (C:\Users\simaja\.ssh\id_ed25519):
C:\Users\simaja\.ssh\id_ed25519 already exists.
Overwrite (y/n)?
```

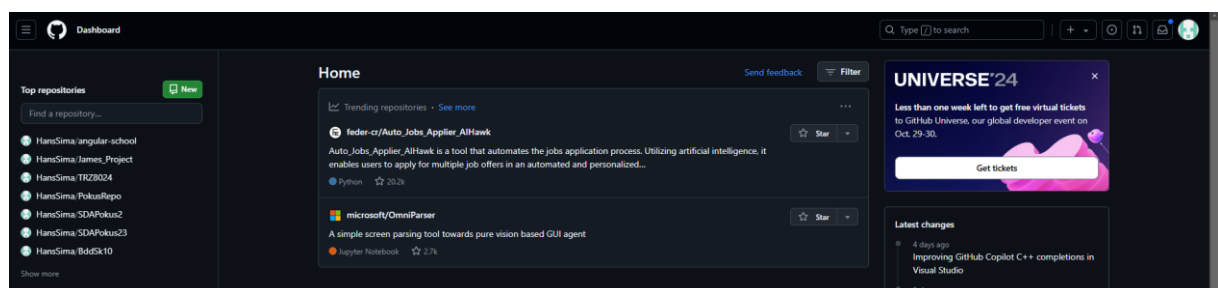


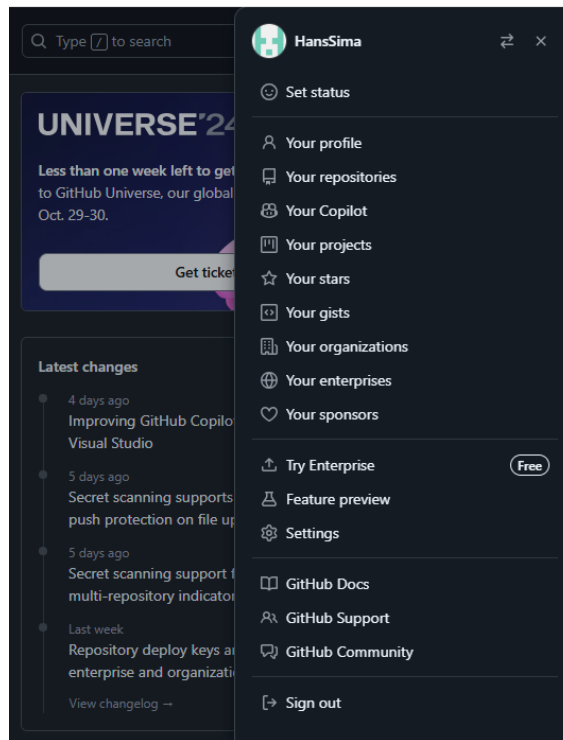
Název	Datum změny	Typ	Velikost
id_ed25519	16.09.2024 12:30	Soubor	1 kB
id_ed25519.pub	16.09.2024 12:30	Microsoft Publish...	1 kB
id_rsa	08.09.2023 16:05	Soubor	3 kB
id_rsa.pub	08.09.2023 16:05	Microsoft Publish...	1 kB
known_hosts	04.10.2023 12:35	Soubor	1 kB
known_hosts.old	08.09.2023 10:15	Soubor OLD	1 kB

```
1 ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIEb+e2yUhrV2DzLC6XpMIQ9ZjRb/y7y5CeJmj3dA4Rd5 tvuj@email.com
```

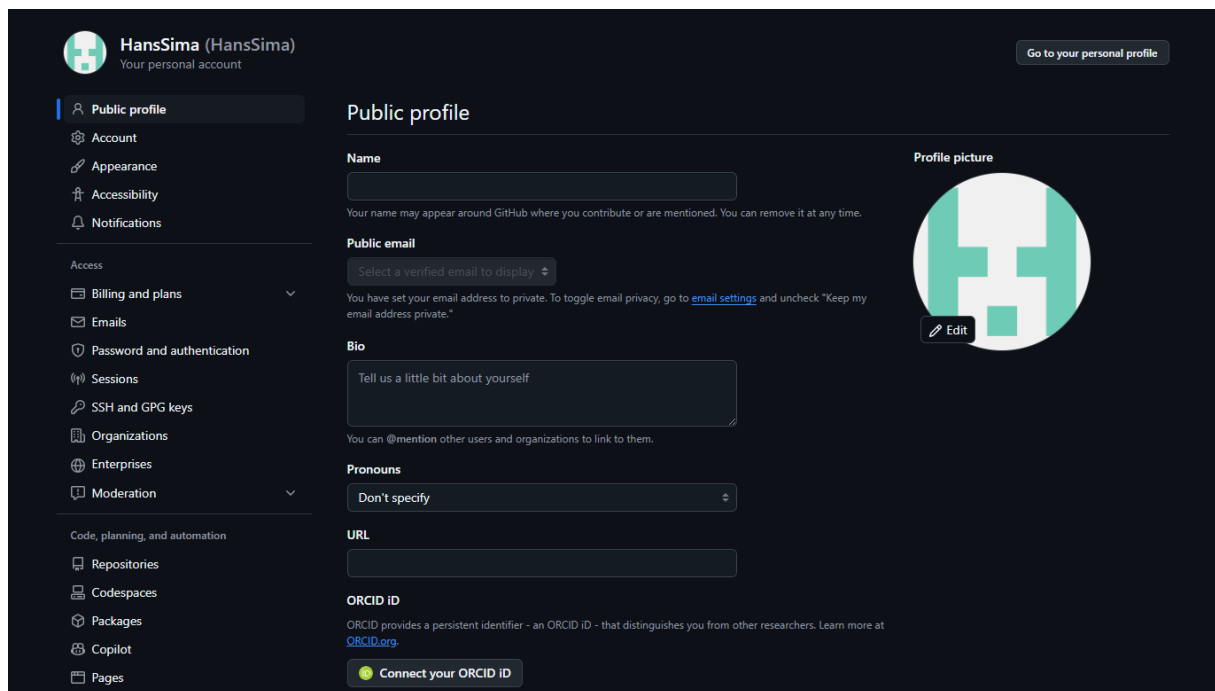
## Přidání SSH klíče na GitHub

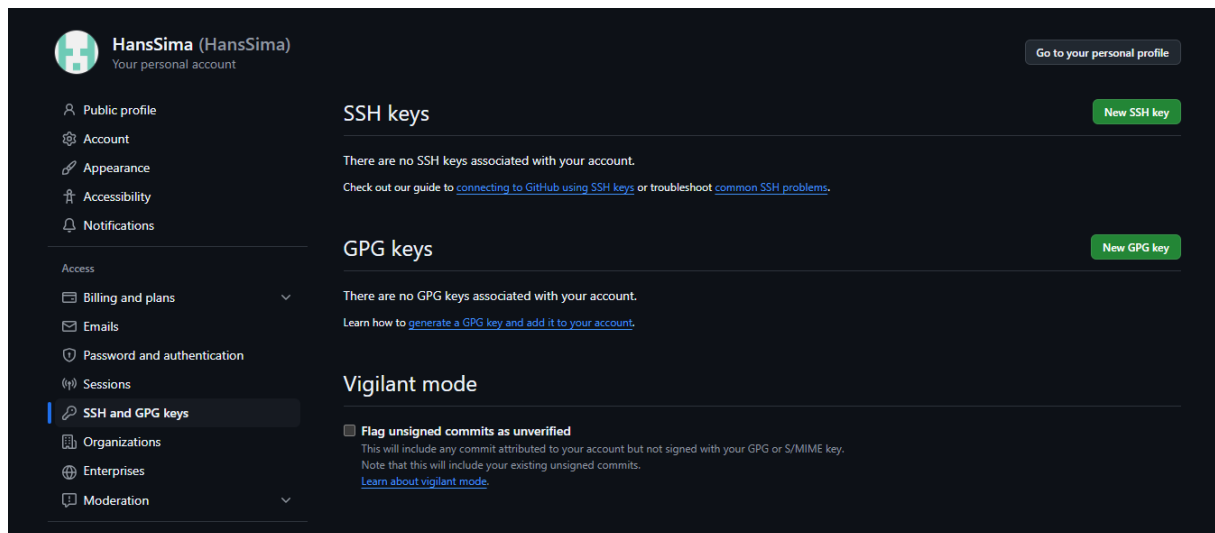
- Na GitHubu přejdi do nastavení (Settings) a vyber **SSH and GPG keys**.



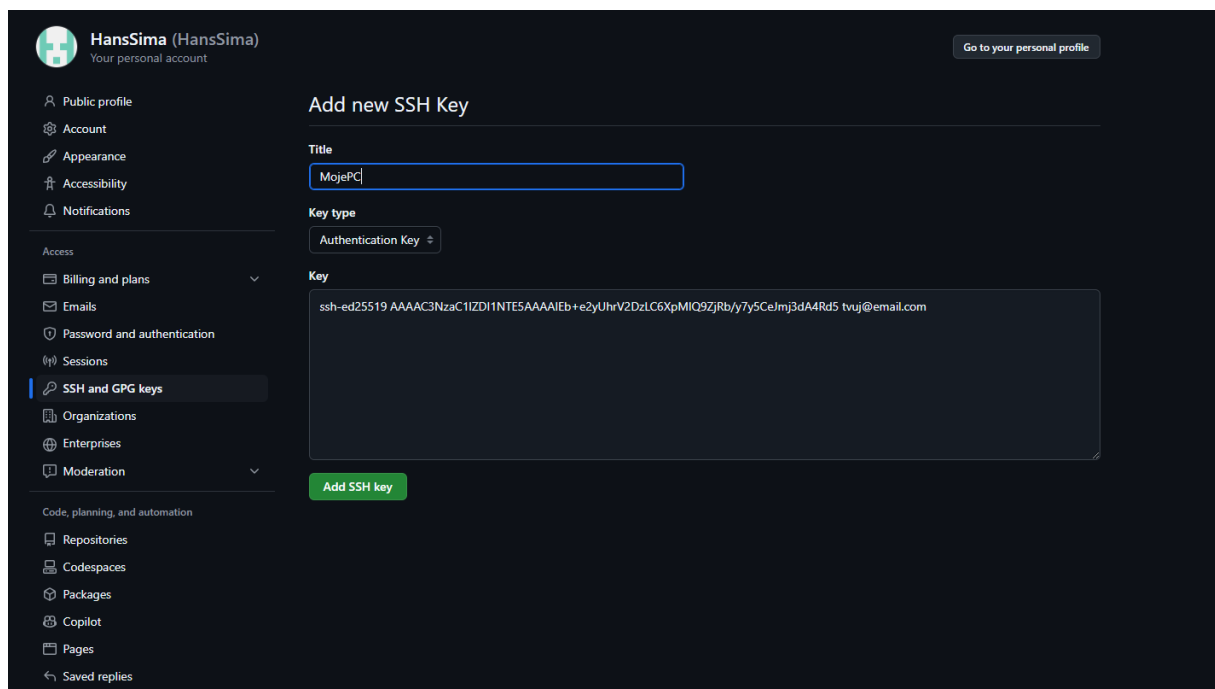


- Klikni na tlačítko **New SSH key**.





- Pojmenuj klíč (např. “MojePC”) a do pole **Key** vlož zkopírovaný obsah SSH klíče.
- Klikni na **Add SSH key**.



## Test propojení s GitHubem

- V Git Bash napiš následující příkaz:

```
ssh -T git@github.com
```

- Pokud je vše v pořádku, zobrazí se zpráva podobná tomuto:

```
Hi username! You've successfully authenticated, but GitHub does not
provide shell access.
```

Tímto je propojení Git a GitHub účtu na Windows hotové!

## Základní vlastnosti Gitu

- **Distribučný systém:** Každý uživatel má plnou kopii celého repozitáře, což znamená, že může pracovat offline a všechny změny se synchronizují s centrálním repozitářem když je to potřeba.
- **Sledování verzí:** Git umožňuje sledovat změny v kódu v čase, což umožňuje návrat k předchozím verzím, revizi historie a sledování změn, které byly provedeny.
- **Rychlost:** Operace v Gitu, jako je přepínání větví, jsou velmi rychlé, což zvyšuje efektivitu práce.
- **Podpora větvení a sloučení:** Git usnadňuje práci s větvemi, což umožňuje vývojářům vyvíjet nové funkce nebo opravy v izolovaných větvích, které mohou být později sloučeny zpět do hlavní větve (např. main nebo master).
- **Bezpečnost:** Git používá kryptografické hashování pro zajištění integrity dat, což znamená, že jakékoli změny v historii jsou jasně sledovatelné.
- **Spolupráce:** Git usnadňuje spolupráci mezi více vývojáři tím, že umožňuje snadné sloučení změn a řešení konfliktů, které mohou nastat při práci na stejných souborech.

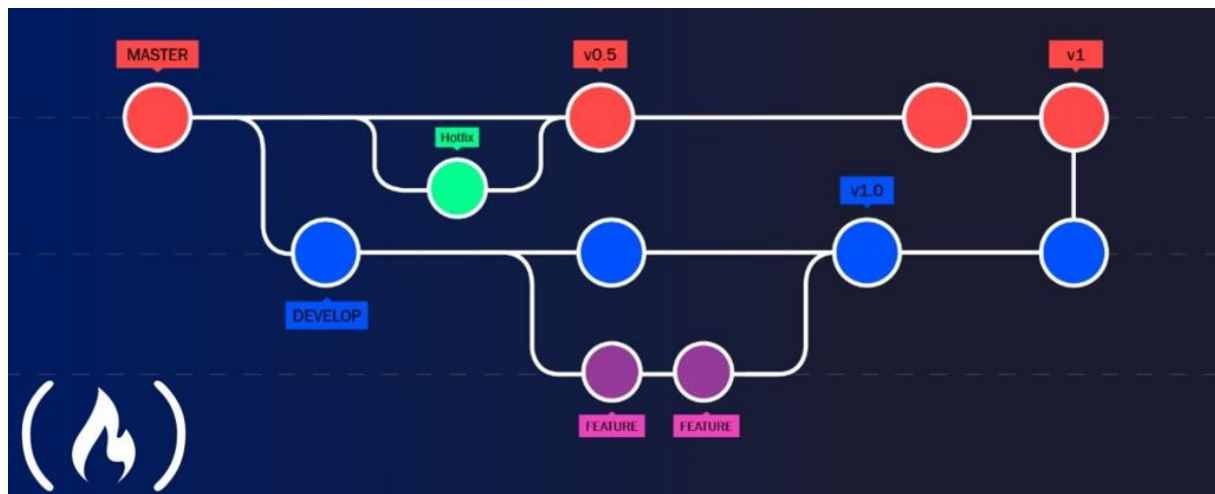
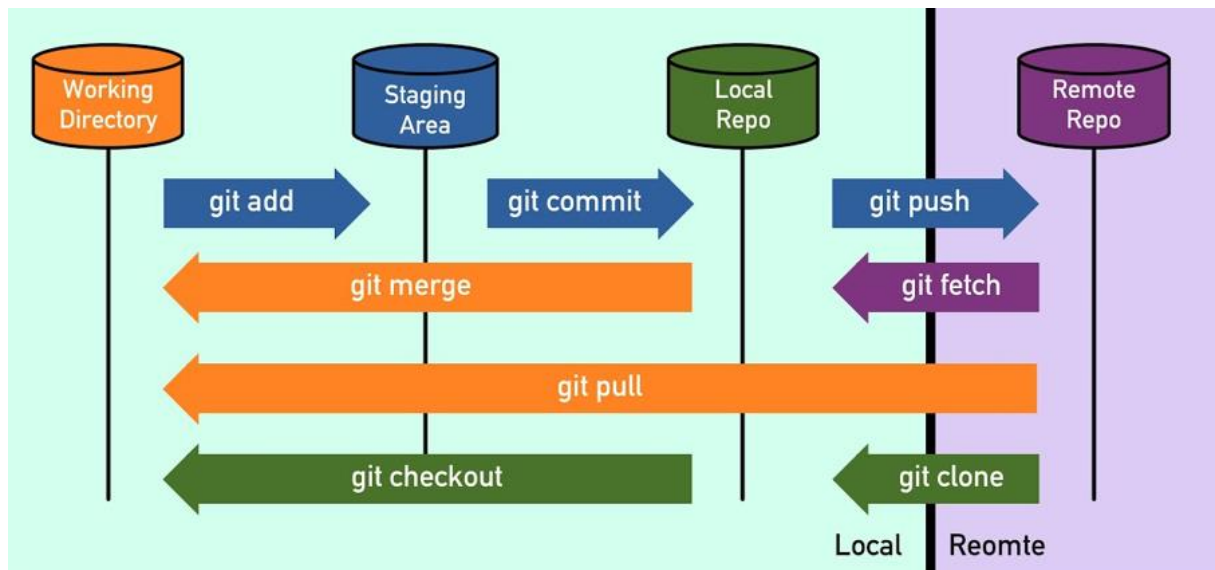
## Klíčové pojmy v Gitu

- **Repozitář (repository):** Místo, kde jsou uloženy všechny související soubory a historie verzí projektu.
- **Commit:** Záznam o změnách provedených v kódu. Každý commit má unikátní identifikátor a obsahuje zprávu, která popisuje změny.
- **Větev (branch):** Izolovaná linie vývoje, která umožňuje pracovat na různých funkcích nebo opravách bez ovlivnění hlavní větve.
- **Merge:** Proces sloučení změn z jedné větve do druhé, který integruje změny a aktualizuje historii.
- **Pull request:** Žádost o sloučení změn z jedné větve do druhé, obvykle s možností revize a diskuse mezi vývojáři.

## Použití Gitu

- **Instalace:** Git je dostupný pro různé operační systémy a lze ho snadno nainstalovat.
- **Základní příkazy:**
  - `git init`: Inicializace nového repozitáře.
  - `git clone <repo-url>`: Klonování existujícího repozitáře.
  - `git add <soubor>`: Přidání změn do staging area.
  - `git commit -m "popis změn"`: Uložení změn s popisnou zprávou.
  - `git push`: Odeslání změn do vzdáleného repozitáře.
  - `git pull`: Stáhnutí a sloučení změn z vzdáleného repozitáře.

Git je mocný nástroj pro správu verzí, který hraje klíčovou roli v moderním vývoji softwaru. Jeho flexibilita, rychlost a podpora pro spolupráci činí z Gitu jeden z nejpoužívanějších a nejpoužívanějších nástrojů v oblasti vývoje. Dnes je běžně integrován s dalšími platformami, jako jsou GitHub, GitLab nebo Bitbucket, které usnadňují sdílení a spolupráci na projektech.

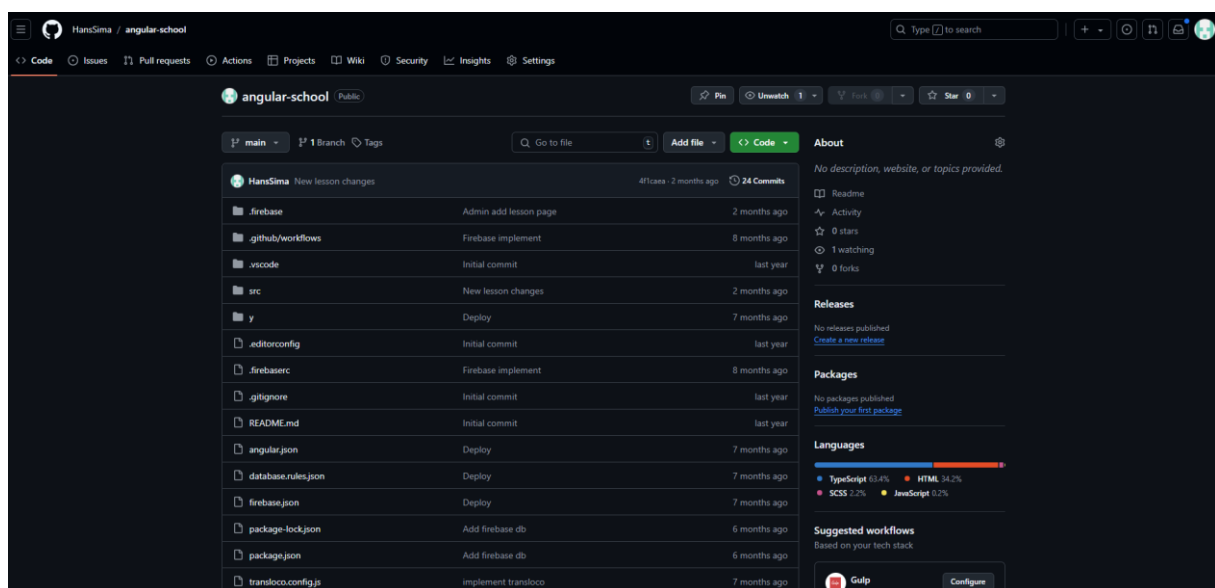
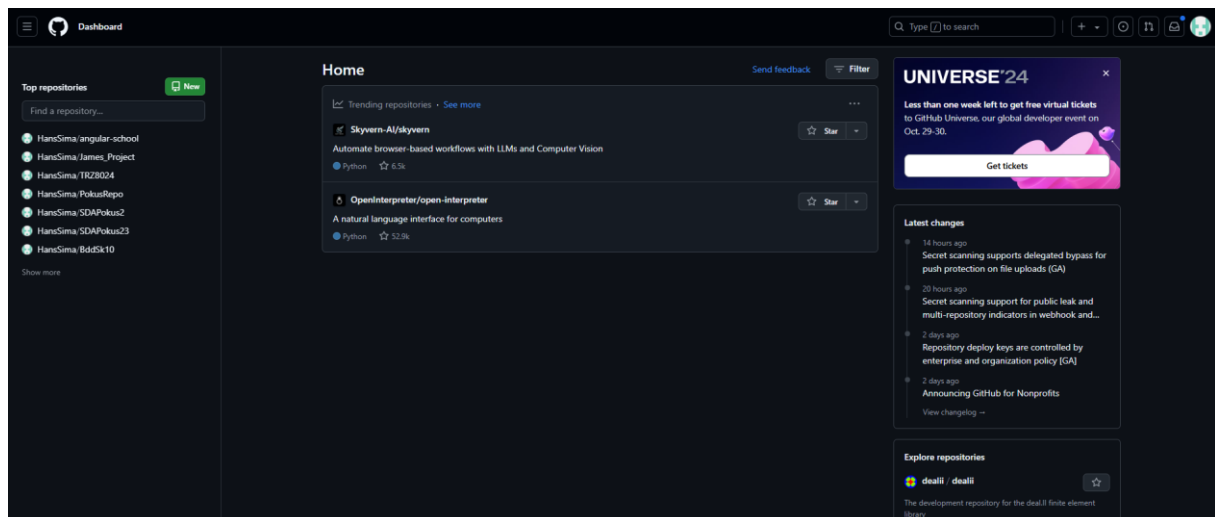


GitHub



GitHub je jednou z největších a nejpopulárnějších platforem pro hostování Git repozitářů. Byla založena v roce 2008 a rychle se stala standardem pro open-source projekty.

Odkaz: [www.github.com](https://www.github.com)

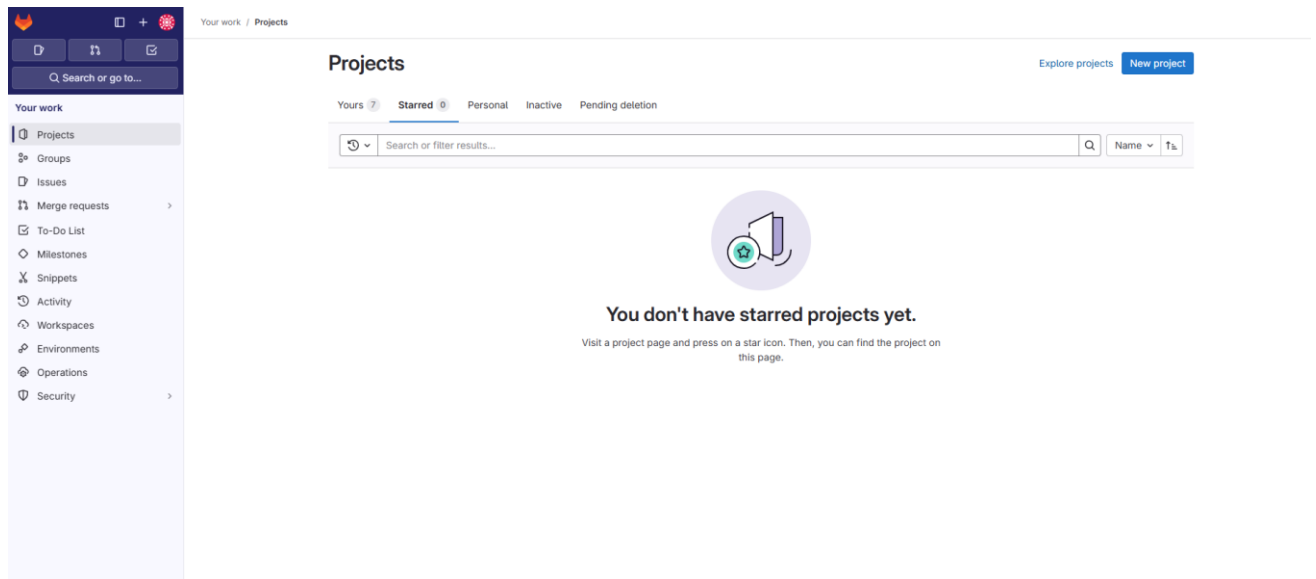


## GitLab



GitLab je další platforma pro hostování Git repozitářů, která byla založena v roce 2011. Nabízí robustní sadu funkcí pro správu životního cyklu vývoje softwaru. Konkurence GitHub.

Odkaz: [www.gitlab.com](https://www.gitlab.com)



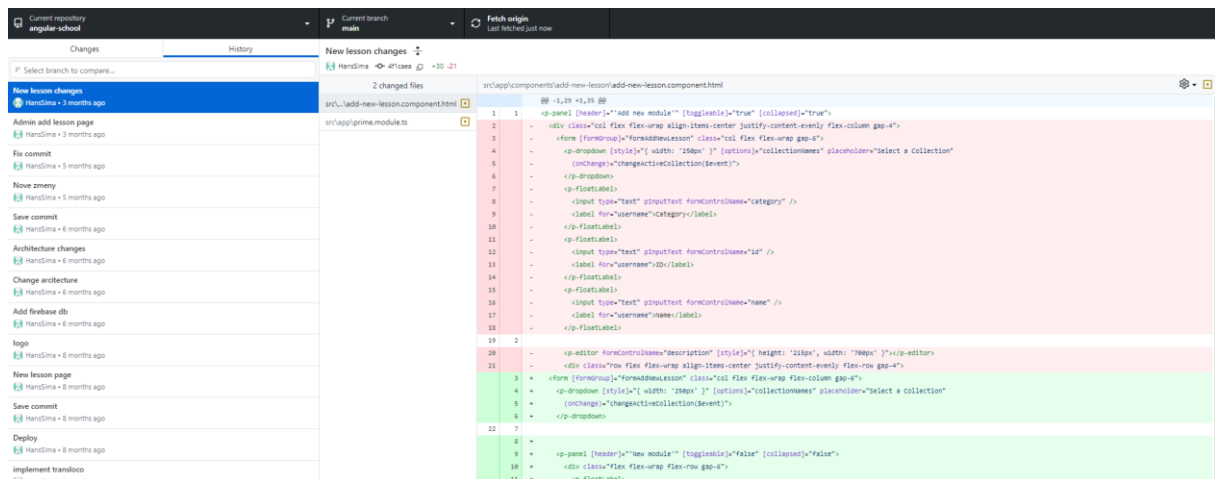
## GIT – GUI

GIT na lokálním PC lze ovládat přes CLI (Command Line Interface) ve Windows lépe známé jako Command prompt (CMD) neboli příkazový řádek. Alternativou k příkazovému řádku ve Windows může být například Powershell. V operačních systémech Linux a MacOS je pak na výběr například Terminál nebo Bash. Mnoho tzv. IDE však poskytuje podporu pro GIT a usnadňuje tak práci. Další alternativou je pak použít jeden z GIT-GUI. Což je software specializovaný přímo na správu GIT.

## GitHub Desktop



- **Popis:** Aplikace vyvinutá společností GitHub, která usnadňuje správu Git repozitářů na platformě GitHub.
- **Funkce:**
  - Snadné klonování, správa a synchronizace repozitářů.
  - Vizualizace změn a historie.
  - Umožňuje vytváření pull requestů a sledování problémů.
  - Podpora pro více účtů.

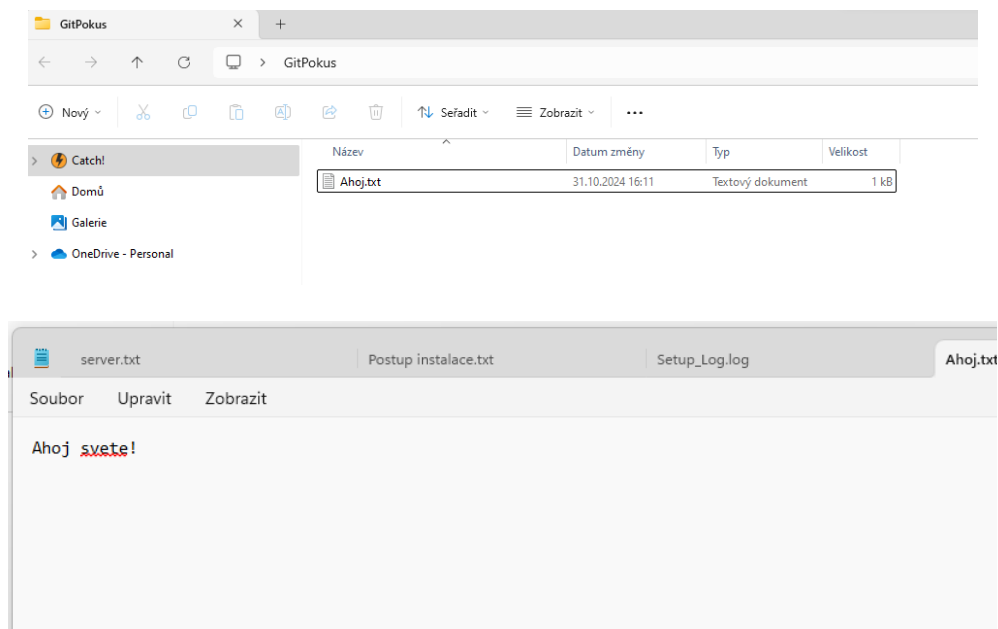


## GitKraken

- **Popis:** Moderní Git klient s intuitivním uživatelským rozhraním, který je k dispozici na více platformách.
- **Funkce:**
  - Podpora pro GIT-flow, což usnadňuje práci s větvemi.
  - Grafické zobrazení commitů a větví.
  - Vytváření a správa pull requestů.
  - Integrace s různými platformami, jako jsou GitHub, Bitbucket a GitLab.

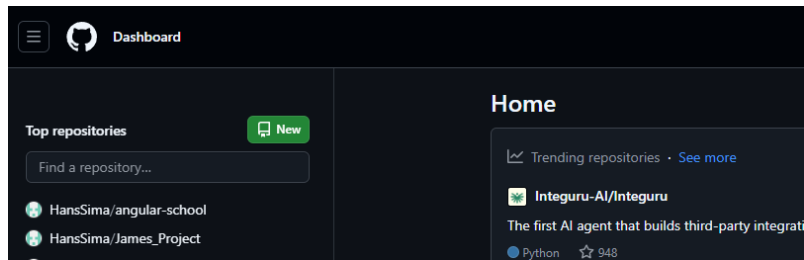
## Praktická zkouška GIT

- Vytvořím složku GitPokus a do ní textový soubor. Do textového souboru napíšu nějaký text:



- Přepnu se na GitHub a založím nový repositář klikem na tlačítko New:





- V dalším okně pojmenuju projekt a nastavím modifikátor přístupu (private nebo public). V tuto chvíli je lepší vybrat private. A klikneme na Create repository:

**Create a new repository**

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

*Required fields are marked with an asterisk (\*).*

Owner \* HansSima / Repository name \* NewRepository  
 NewRepository is available.

Great repository names are short and memorable. Need inspiration? How about [silver-lamp](#) ?

Description (optional)

☐ **Public**  
 Anyone on the internet can see this repository. You choose who can commit.

☒ **Private**  
 You choose who can see and commit to this repository.

Initialize this repository with:

☐ **Add a README file**  
 This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore  
 .gitignore template: None

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

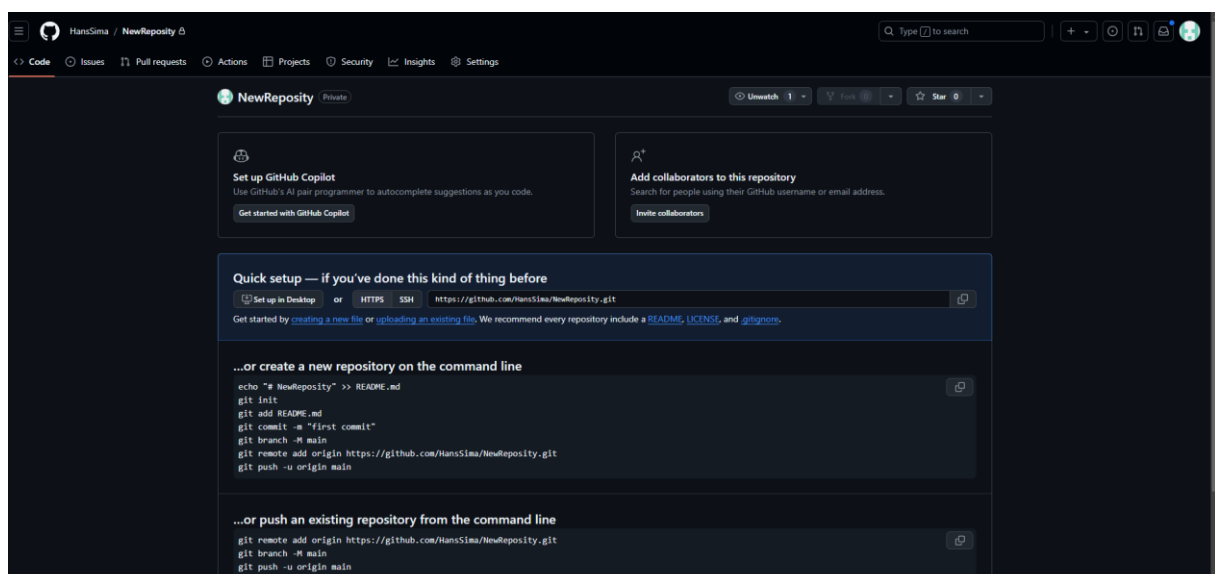
Choose a license  
 License: None

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

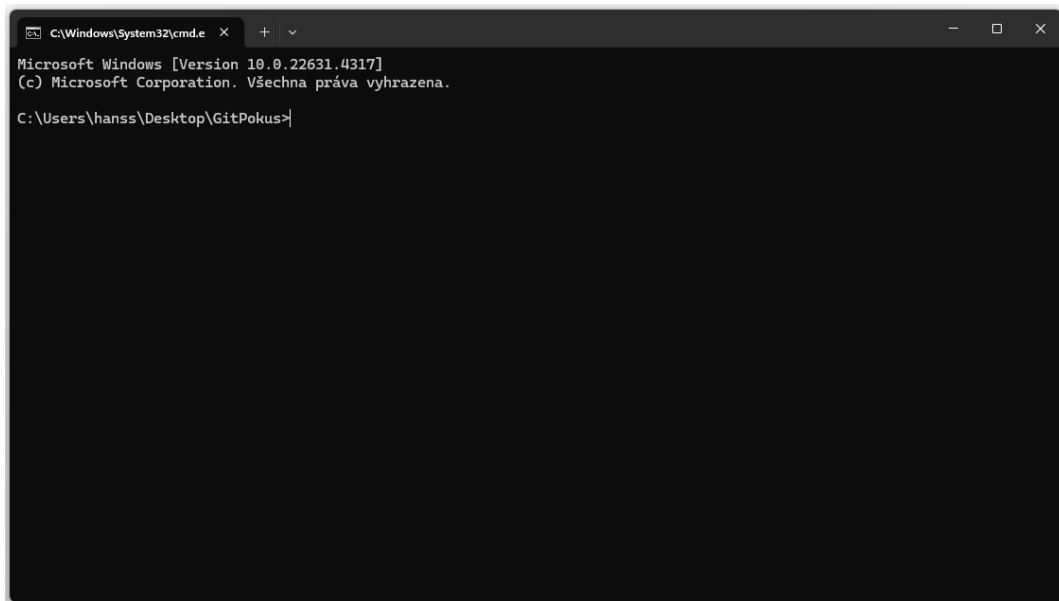
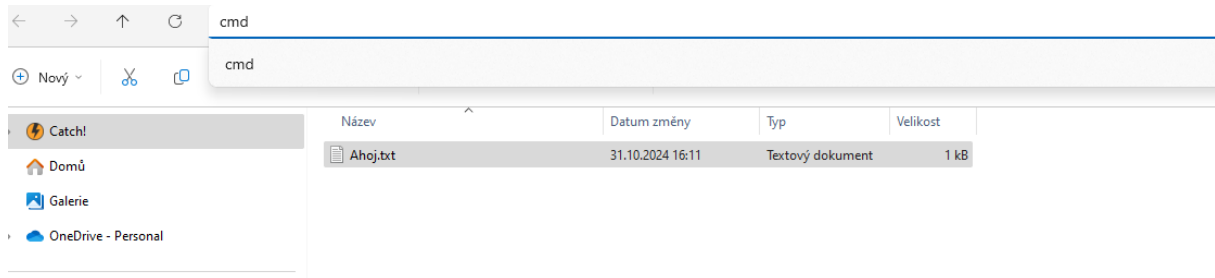
ⓘ You are creating a private repository in your personal account.

**Create repository**

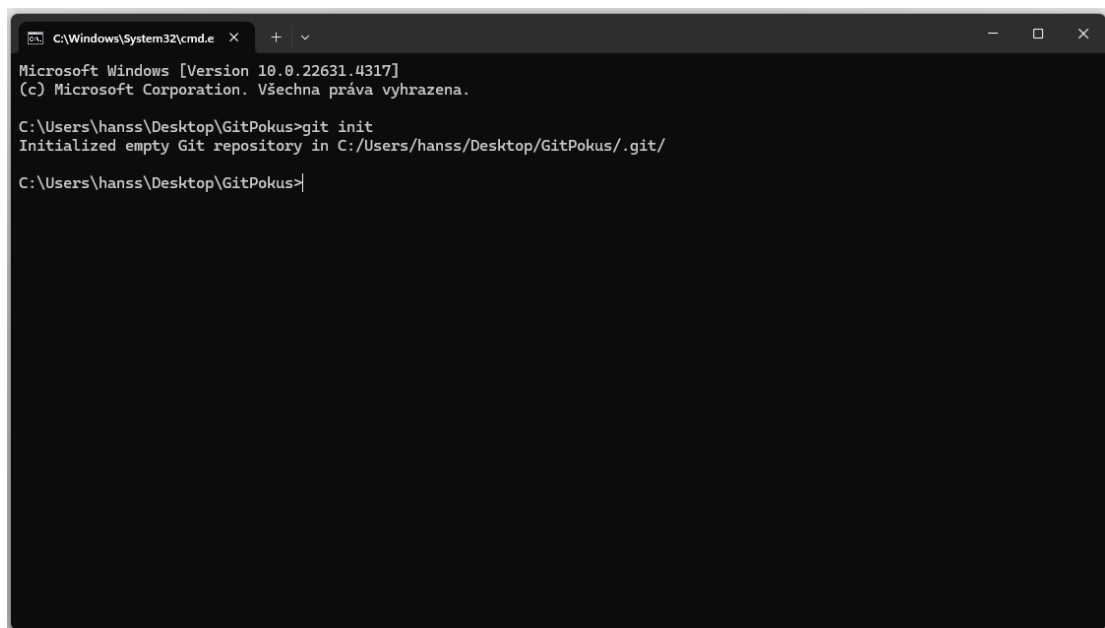
- To vytvoří nový repositář. Pro nahrání kódu lze pak využít i přiložený návod **...or create a new repository on the command line**:



- Překliknu se zpět do složky s textovým souborem a místo cesty napíšu příkaz cmd. To otevře Příkazový řádek přímo na složce:



- Zadám příkaz git init. To vytvoří novou git složku (ta je skrytá) a iniciuje GIT na složce:



- Zadám příkaz git add . . To nastaví GITu, že má verzovat všechny soubory ve složce (pozor za add je ještě tečka):

```
C:\Windows\System32\cmd.e X + v
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. Všechna práva vyhrazena.

C:\Users\hanss\Desktop\GitPokus>git init
Initialized empty Git repository in C:/Users/hanss/Desktop/GitPokus/.git/

C:\Users\hanss\Desktop\GitPokus>git add .

C:\Users\hanss\Desktop\GitPokus>|
```

- Zadám příkaz `git commit -m "prvni commit"`. To nastaví GITu, že má uložit změny do lokálního repozitáře:

```
C:\Windows\System32\cmd.e X + v
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. Všechna práva vyhrazena.

C:\Users\hanss\Desktop\GitPokus>git init
Initialized empty Git repository in C:/Users/hanss/Desktop/GitPokus/.git/

C:\Users\hanss\Desktop\GitPokus>git add .

C:\Users\hanss\Desktop\GitPokus>git commit -m "prvni commit"
[master (root-commit) 5c7a320] prvni commit
1 file changed, 1 insertion(+)
create mode 100644 Ahoj.txt

C:\Users\hanss\Desktop\GitPokus>|
```

Zadám příkaz `git remote add origin https://github.com/HansSima/NewRepository.git`. Propojí vzdálený repozitář na GitHubu s lokálním repozitářem. Odkaz na váš vzdálený repozitář najdete na vašem GitHubu v konkrétním repozitáři:

```
C:\Windows\System32\cmd.e X + v
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. Všechna práva vyhrazena.

C:\Users\hanss\Desktop\GitPokus>git init
Initialized empty Git repository in C:/Users/hanss/Desktop/GitPokus/.git/

C:\Users\hanss\Desktop\GitPokus>git add .

C:\Users\hanss\Desktop\GitPokus>git commit -m "prvni commit"
[master (root-commit) 5c7a320] prvni commit
1 file changed, 1 insertion(+)
create mode 100644 Ahoj.txt

C:\Users\hanss\Desktop\GitPokus>git remote add origin https://github.com/HansSima/NewRepository.git

C:\Users\hanss\Desktop\GitPokus>|
```

- Zadám příkaz git branch -M main. Díky tomu vytvoří GIT branch se jménem main:

```
C:\Users\hanss\Desktop\GitPokus>git branch -M main
```

- Zadám příkaz git push -u origin main. Tímto příkazem nahrajeme branch main na vzdálený repositář GitHubu:

```
C:\Windows\System32\cmd.e X + v
(c) Microsoft Corporation. Všechna práva vyhrazena.

C:\Users\hanss\Desktop\GitPokus>git init
Initialized empty Git repository in C:/Users/hanss/Desktop/GitPokus/.git/

C:\Users\hanss\Desktop\GitPokus>git add .

C:\Users\hanss\Desktop\GitPokus>git commit -m "prvni commit"
[master (root-commit) 5c7a320] prvni commit
1 file changed, 1 insertion(+)
create mode 100644 Ahoj.txt

C:\Users\hanss\Desktop\GitPokus>git remote add origin https://github.com/HansSima/NewRepository.git

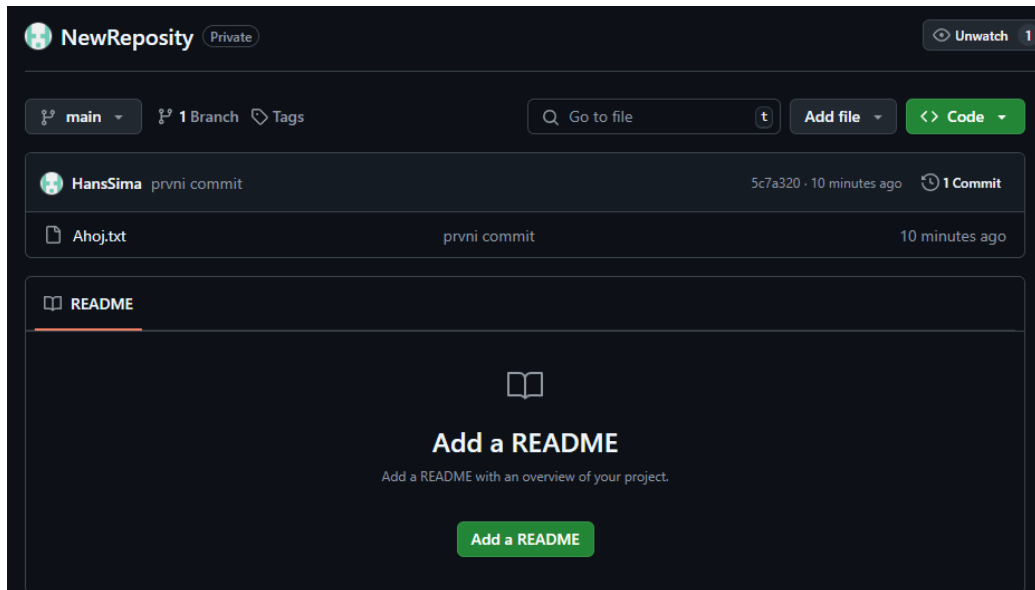
C:\Users\hanss\Desktop\GitPokus>git push -u origin main
error: src refspec main does not match any
error: failed to push some refs to 'https://github.com/HansSima/NewRepository.git'

C:\Users\hanss\Desktop\GitPokus>git branch -M main

C:\Users\hanss\Desktop\GitPokus>git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 221 bytes | 221.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/HansSima/NewRepository.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.

C:\Users\hanss\Desktop\GitPokus>|
```

- Nahraný soubor lze nalézt na repositáři v GitHub:



- Pokud něco změníte a chcete nahrát novou verzi na vzdálený repositář použijte tyto příkazy
  - `git add <nazevSouboru>` (použijte tečku místo názvu souborů pokud chcete nahrát vše z adresáře)
  - `git commit -m "<zpravaCommitu>"`
  - `git push -u origin <jmenoBranche>`

```
C:\Users\hanss\Desktop\GitPokus>git add Ahoj.txt

C:\Users\hanss\Desktop\GitPokus>git commit -m "druhy commit"
[main 673ea45] druhy commit
 1 file changed, 1 insertion(+), 1 deletion(-)

C:\Users\hanss\Desktop\GitPokus>git push -u origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Writing objects: 100% (3/3), 256 bytes | 256.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/HansSima/NewRepository.git
 5c7a320..673ea45  main -> main
branch 'main' set up to track 'origin/main'.
```

## Nejpoužívanější jazyky

Nejpoužívanější programovací jazyky se mohou lišit v závislosti na konkrétních oborech, ale podle globálních statistik pro rok 2024 jsou nejpoužívanější jazyky:

- **JavaScript** – Jeden z nejrozšířenějších jazyků, především pro vývoj webových aplikací (frontend i backend díky Node.js).



```
console.log("Hello, World!");
```

- **TypeScript** – Rozšíření JavaScriptu, přidává typovou bezpečnost, velmi populární mezi vývojáři velkých projektů.



```
console.log("Hello, World!");
```

- **Python** – Oblíbený pro vědecké výpočty, datovou analýzu, umělou inteligenci a automatizaci.



```
print("Hello, World!")
```

- **Java** – Stále velmi používaný pro velké enterprise systémy a mobilní aplikace (zejména Android).



```
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello, World!");  
    }  
}
```

- **C#** – Populární zejména ve vývoji her díky Unity a pro Windows aplikace.



```
using System;

class HelloWorld {
    static void Main() {
        Console.WriteLine("Hello, World!");
    }
}
```

- **C/C++** – Používané v oblasti systémového a embedded vývoje, herní engine a výkonnostně náročné aplikace.



C++ příklad:

```
#include <iostream>

int main() {
    std::cout << "Hello, World!" << std::endl;
    return 0;
}
```

- **PHP** – Především v oblasti webového vývoje, zejména pro backend aplikací.



```
<?php
echo "Hello, World!";
?>
```

- **Go** – Stále populárnější jazyk, zaměřený na výkon a jednoduchost, používán například pro vývoj serverů a distribuovaných systémů.



```
package main

import "fmt"

func main() {
    fmt.Println("Hello, World!")
}
```

- **Swift** – Oficiální jazyk pro vývoj iOS a macOS aplikací.



```
print("Hello, World!")
```

- **Rust** – Roste v popularitě pro svou bezpečnost paměti a výkonnost, používán v systémech s vysokými nároky na výkon a bezpečnost.



```
fn main() {
    println!("Hello, World!");
}
```

Používání se však může měnit v závislosti na trendu, projektech a konkrétních požadavcích vývojářské komunity.

## Frameworky

Framework je sada nástrojů, knihoven a pravidel, která usnadňují vývoj softwaru. Poskytuje strukturu a základní funkce, takže se vývojář může soustředit na specifické části aplikace, aniž by musel začínat od nuly.

JavaScript (Typescript)	Python	Java	C#	C++	PHP	Go	Swift	Rust
React	Django	Spring	.NET (ASP.NET Core)	Qt	Laravel	Gin	SwiftUI	Rocket
Angular	Flask	Hibernate	Entity Framework	Boost	Symfony	Echo	UIKit	Actix
Vue.js			Unity					
Node.js								

### 1. JavaScript (Typescript)

- **Angular** – Kompletní framework od Googlu s rozsáhlými možnostmi pro vývoj komplexních aplikací.
- **React** – Knihovna pro tvorbu uživatelských rozhraní, populární zejména v oblasti frontendu.
- **Vue.js** – Flexibilní a snadno použitelný framework pro budování uživatelských rozhraní.



- **Node.js** – Serverová platforma, která umožňuje běh JavaScriptu na backendu.

## 2. Python

- **Django** – Framework pro rychlý vývoj webových aplikací s vysokou bezpečností.
- **Flask** – Lehký webový framework, který je vhodný pro menší aplikace a mikroslužby.

## 3. Java

- **Spring (Spring Boot)** – Komplexní framework pro vývoj enterprise aplikací, zejména pro backend a webové služby.
- **Hibernate** – ORM (Object-Relational Mapping) framework pro práci s databázemi.

## 4. C#

- **.NET (ASP.NET Core)** – Hlavní framework pro vývoj webových aplikací a API na platformě .NET.
- **Entity Framework** – ORM pro práci s databázemi, který umožňuje efektivní mapování mezi objekty a databázemi.
- **Unity** – Framework a engine pro tvorbu her, využívaný hlavně pro vývoj 2D a 3D her.

## 5. C++

- **Qt** – Populární framework pro vývoj multiplatformních GUI aplikací.
- **Boost** – Kolekce knihoven pro různé oblasti (např. datové struktury, síťové operace).

## 6. PHP

- **Laravel** – Moderní a oblíbený framework pro vývoj webových aplikací.
- **Symfony** – Flexibilní framework, široce používaný zejména ve velkých projektech.

## 7. Go

- **Gin** – Lehký a rychlý framework pro tvorbu REST API.
- **Echo** – Minimalistický a výkonný framework pro tvorbu webových aplikací a API.

## 8. Swift

- **SwiftUI** – Moderní framework od Applu pro tvorbu uživatelských rozhraní na iOS a macOS.
- **UIKit** – Tradiční framework pro vývoj aplikací pro iOS.

## 9. Rust

- **Rocket** – Výkonný framework pro vývoj webových aplikací a API.
- **Actix** – Asynchronní webový framework s vysokým výkonem.

## IDE

IDE vývojářům usnadňují práci při psaní, testování a ladění kódu. Každé IDE má své specifické vlastnosti a podporu pro různé programovací jazyky:

- **IntelliJ IDEA** – Silné a oblíbené IDE pro vývoj v Javě, ale také podporuje další jazyky jako Kotlin, Scala, Groovy, a Python. Je ceněné pro své chytré nástroje a výkonné refactoringy.



- **Visual Studio Code (VS Code)** – Lehké, ale výkonné prostředí od Microsoftu, populární mezi vývojáři různých programovacích jazyků díky rozsáhlé podpoře rozšíření a pluginů. Často používané pro JavaScript, Python, Go, a mnoho dalších jazyků.



- **Visual Studio** – Plnohodnotné IDE od Microsoftu zaměřené na vývoj ve Windows, C#, .NET, ale také podporuje mnoho dalších jazyků, včetně C++, Pythonu a F#. Používá se zejména v enterprise a herním vývoji.



- **PyCharm** – IDE zaměřené speciálně na Python, vyvinuté společností JetBrains (stejná firma jako IntelliJ IDEA). Nabízí podporu pro datovou vědu, analýzu kódu, a vývoj webových aplikací v Pythonu.



- **Eclipse** – Tradiční IDE často používané pro vývoj v Javě, ale s podporou i dalších jazyků. Mnoho velkých projektů a enterprise aplikací stále využívá Eclipse.



- **Xcode** – Oficiální IDE od Apple pro vývoj aplikací pro iOS, macOS, watchOS a tvOS, především ve Swift a Objective-C.



- **Android Studio** – Hlavní IDE pro vývoj Android aplikací, založené na IntelliJ IDEA a podporující jazyky jako Java a Kotlin.



- **NetBeans** – Open-source IDE často používané pro Javu, ale také podporuje PHP, HTML5, a další jazyky. Využíváno zejména ve vzdělávacím prostředí.



- **Sublime Text** – I když nejde o plnohodnotné IDE, mnoho vývojářů používá Sublime Text jako lehký textový editor s množstvím pluginů a rozšíření pro různé jazyky a frameworky.



## Nástroje pro automatizaci testů

Nástroje pro automatizaci testů jsou nezbytné pro efektivní testování softwaru. Tyto nástroje umožňují automatizaci opakovaných testovacích případů a zajišťují, že aplikace funguje správně při každé změně kódu.

### Selenium



- **Popis:** Selenium je populární open-source nástroj pro automatizaci testování webových aplikací.
- **Hlavní funkce:**
  - Podpora pro různé prohlížeče (Chrome, Firefox, Safari, atd.).
  - Možnost psaní testů v několika programovacích jazycích (Java, C#, Python, Ruby).
  - Podpora pro paralelní testování.
- Bude dále probíráno v modulu „Selenium“

### Cypress



- **Popis:** Cypress je moderní nástroj pro end-to-end testování webových aplikací, který se zaměřuje na vývojáře.
- **Hlavní funkce:**
  - Snadné nastavení a rychlé provádění testů.
  - Možnost sledovat testy v reálném čase.
  - Podpora pro testování jak REST, tak GraphQL API.

### Robot Framework



- **Popis:** Robot Framework je open-source nástroj pro automatizaci testování s využitím klíčových slov.
- **Hlavní funkce:**
  - Podpora pro různé testovací knihovny (Selenium, Appium, API testing).
  - Možnost psaní testů ve snadno čitelném formátu.
  - Generování reportů a logů.

## Další nástroje

### Nástroje pro správu dokumentace

Nástroje pro správu dokumentace usnadňují týmům tvorbu, organizaci, údržbu a sdílení dokumentace k projektům, což je klíčové pro efektivní spolupráci a sdílení znalostí. Zde jsou některé z nejpoužívanějších nástrojů pro správu dokumentace:

#### Confluence



- **Popis:** Nástroj od Atlassianu zaměřený na týmovou spolupráci a správu znalostí. Nabízí možnost vytváření a sdílení dokumentace, zápisů ze schůzek, projektových plánů a dalších typů dokumentů.
- **Použití:** Dokumentace projektů, zápisy schůzek, příručky, interní wiki.
- **Výhody:** Integrace s Jira, přizpůsobitelné šablony, organizační struktura pomocí stromů stránek a flexibilita formátování.

#### Microsoft Word



- **Popis:** Součást Microsoft Office, tradiční nástroj pro vytváření dokumentů s rozsáhlými možnostmi formátování a úprav.
- **Použití:** Dokumentace projektů, příručky, formální zprávy a smlouvy, technické specifikace.
- **Výhody:** Široké možnosti formátování, revize a komentáře pro spolupráci a integraci s dalšími nástroji Microsoft Office a sdílení přes OneDrive.

#### Microsoft Excel



- **Popis:** Nástroj pro správu a analýzu dat, který nabízí rozsáhlé možnosti práce s tabulkami, výpočty a grafy. Excel je často používán i pro organizaci informací a základní dokumentaci.
- **Použití:** Technické reporty, tabulky s metrikami, analytické přehledy, plány projektů.
- **Výhody:** Pokročilé funkce pro analýzu dat (včetně grafů, funkcí a kontingenčních tabulek), snadná organizace číselných dat, integrace s Power BI a dalšími nástroji.

### Nástroje pro výkonové a zatěžkávací testy

Výkonové a zatěžkávací testy (performance a load testing) jsou klíčové pro zajištění, že aplikace bude fungovat spolehlivě i při vysokém zatížení. Tyto testy simulují provozní podmínky a měří, jak aplikace reaguje na různé úrovně zátěže, což pomáhá identifikovat potenciální problémy s výkonem a škálovatelností. Níže jsou uvedeny některé z nejpoužívanějších nástrojů pro tyto typy testů:

## Apache JMeter



- **Popis:** Open-source nástroj pro zatěžkávací a výkonové testování webových aplikací. Podporuje různé protokoly jako HTTP, FTP, JDBC a SOAP.
- **Použití:** Zatěžkávací testování webových aplikací, simulace různých úrovní zátěže a testování API.
- **Výhody:** Flexibilita, široká škála funkcí, možnost spouštět testy na více vláknech pro simulaci většího množství uživatelů a bohatá komunita.

## LoadRunner



- **Popis:** Výkonný nástroj pro výkonové testování, který podporuje široké spektrum aplikací a protokolů. Poskytuje pokročilé analýzy a monitorování.
- **Použití:** Výkonové testování aplikací s vysokou zátěží, testování chování aplikací v reálných scénářích.
- **Výhody:** Robustní monitorovací možnosti, podpora různých typů aplikací, bohatá analytická rozhraní pro podrobnou diagnostiku.

## Nástroje pro testování bezpečnosti

Nástroje pro testování bezpečnosti aplikací a systémů jsou nezbytné pro identifikaci zranitelností a zabezpečení softwarových produktů.

### OWASP ZAP



- **Popis:** Open-source nástroj pro testování webových aplikací, který pomáhá identifikovat zranitelnosti.
- **Použití:** Aktivní skenování a pasivní analýza webových aplikací, testování na zranitelnosti jako XSS, SQL Injection, CSRF.
- **Výhody:** Uživatel-friendly, podporuje automatizaci, rozsáhlá dokumentace a aktivní komunita.

## CI/CD nástroje

Nástroje pro Continuous Integration (CI) a Continuous Deployment (CD) hrají klíčovou roli v moderním vývoji softwaru. Umožňují automatizaci procesu vývoje, testování a nasazení aplikací, což zajišťuje rychlejší a spolehlivější dodání softwaru.

### Jenkins



- **Popis:** Open-source automatizační server, který umožňuje automatizaci různých aspektů vývoje softwaru, včetně CI/CD.
- **Použití:** Spouštění buildů, automatizace testování, nasazení aplikací.
- **Výhody:** Velká komunita, široká nabídka pluginů pro integraci s různými nástroji a technologiemi, flexibilita.

## Gitlab

- **Popis:** Integrovaný CI/CD systém, který je součástí GitLab. Umožňuje správu repozitářů a automatizaci buildů a nasazení.
- **Použití:** Automatizace workflow od kódu po nasazení, správa repozitářů.
- **Výhody:** Snadná integrace s GitLabem, uživatelsky přívětivé rozhraní, podpora pro různá prostředí a kontejnery.

## Nástroje pro mobile testing

Nástroje pro testování mobilních aplikací jsou nezbytné pro zajištění kvality a výkonu aplikací na různých mobilních platformách.

### Android Studio

- **Popis:** Oficiální IDE pro vývoj Android aplikací, které obsahuje vestavěné nástroje pro testování a ladění.
- **Použití:** Vývoj, testování a ladění Android aplikací, včetně podpory pro unit testy a instrumentační testy.
- **Výhody:** Integrované nástroje pro testování, podpora pro Espresso a UI Automator, snadná konfigurace testovacích prostředí.

## Podpůrné nástroje

Zoom, Slack, Microsoft Teams, Webex



Nástroje pro komunikaci ve firmě. Výběr nástroje závisí na rozhodnutí vedení nebo týmu.

### Snipping Tool, Print screen



Nástroje pro získání screenshotu obrazovky. Lépe identifikuje chyby, výsledný screenshot lze přiložit k hlášení o chybě

### OBS Studio, ScreenFlow



Nástroje pro nahrávání obrazovky jsou používány pro identifikaci složitějších chyb. Výslednou nahrávku lze přiložit k hlášení o chybě. Nutno podotknout, že v praxi není tak často využíváno. Screenshot bývá dostačující.