Martins Bērziņš 12.A

Cēsu Valsts ģimnāzija

**NASA API Datu Vizualizācijas un Meklēšanas Rīks**

**Ievads**

Astronomijas un kosmosa izpētes dati ir plaši pieejami, tomēr plašai sabiedrībai un skolēniem tie bieži vien nav viegli pieejami vai saprotami. NASA piedāvā vairākas publiskas API (lietojumprogrammu saskarnes), kas nodrošina piekļuvi milzīgam datu apjomam, ieskaitot Mars Rover fotoattēlus, astronomiskos attēlus (APOD - Astronomy Picture of the Day) un citus zinātniskos datus. Problēma ir tāda, ka šie dati nav viegli pieejami, analizējami un vizualizējami bez speciālām programmēšanas zināšanām.

Mūsdienās astronomijas izglītība un kosmosa izpēte ir būtiska jauniešu intereses veicināšanai par STEM (zinātne, tehnoloģija, inženierzinātnes un matemātika) nozarēm. Šis projekts risina nepieciešamību pēc vienkāršas un izglītojošas sistēmas, kas ļauj lietotājiem piekļūt un analizēt kosmosa datus, tādējādi veicinot interesi par astronomiju un programmēšanu.

**Darba aktualitāte:** NASA piedāvā atvērtus datu API, bet daudziem lietotājiem trūkst programmēšanas iemaņu vai ērtas piekļuves šiem datiem. Skolēni, pasniedzēji un astronomijas entuziasti bieži meklē attēlus vai informāciju par konkrētiem notikumiem (piem., meteoru lietus, Zemes attēli no satelītiem), bet nav vienkārša rīka, kas to nodrošinātu.

**Darba mērķis:** Izstrādāt Python balstītu lietotni, kas pieslēdzas NASA API, sniedz lietotājam iespēju meklēt un skatīt NASA attēlus, reāllaika informāciju un saglabāt iecienītākos rezultātus datubāzē.

**Darba uzdevumi:**

* Izstrādāt lietotājam draudzīgu programmatūru, kas izmanto NASA API, lai iegūtu, analizētu un vizualizētu astronomiskos datus.
* Implementēt datubāzi datu glabāšanai un analīzei.
* Implementēt objektorientētās programmēšanas principus, izmantojot klašu hierarhiju.
* Izmantot vismaz vienu specializētu datu struktūru datu efektīvai apstrādei.
* Īstenot datu vizualizācijas iespējas, izmantojot atbilstošas Python bibliotēkas.

**Saturs**

[**1.** **Programmatūras prototipa plānošana** 2](#_Toc197380060)

[**1.1. Datu ieguve par problēmsituāciju un mērķauditoriju** 2](#_Toc197380061)

[**1.2. Programmatūras prasību specifikācija** 2](#_Toc197380062)

[**1.3. Tehnoloģiju izpētes rezultātā izvēlēto resursu apraksts** 3](#_Toc197380063)

[**1.4. Datu aizsardzības pasākumi** 3](#_Toc197380064)

[**2. Detalizēts prototipa apraksts** 4](#_Toc197380065)

[**2.1. Datu uzglabāšana** 4](#_Toc197380066)

[**2.2. API Apraksts** 4](#_Toc197380067)

[**2.3. Testēšanas plāns** 5](#_Toc197380068)

[**2.4. Programmatūras izstrādes un ieviešanas plāns** 5](#_Toc197380069)

[**3. Izstrādes un testēšanas dokumentēšana** 6](#_Toc197380070)

[**3.1. Algoritmu, datu struktūru un bibliotēku izvēles pamatojums** 6](#_Toc197380071)

[**3.2. Testēšanas pārskats** 7](#_Toc197380072)

[**Licence** 8](#_Toc197380073)

[**Lietotāja ceļvedis** 8](#_Toc197380074)

[**Secinājumi** 9](#_Toc197380075)

[**Izmantotie avoti** 10](#_Toc197380076)

[**Pielikumi** 11](#_Toc197380077)

# **Programmatūras prototipa plānošana**

## **1.1. Datu ieguve par problēmsituāciju un mērķauditoriju**

Šobrīd vairums NASA datu vizualizācijas rīku ir paredzēti zinātniekiem vai lietotājiem ar specifiskām tehniskām zināšanām. Mērķis ir izveidot programmatūru, kas būtu piemērota vidusskolas un augstskolas studentiem, kas interesējas par astronomiju, astronomijas skolotājiem, kas vēlas izmantot NASA datus mācību procesā, kā arī astronomijas entuziastiem bez specifiskām tehniskām zināšanām.

Veiktajā izpētē secināts, ka mērķauditorijai ir vairākas būtiskas vajadzības, tostarp vienkārša piekļuve NASA API datiem bez nepieciešamības pēc dziļām programmēšanas zināšanām, iespēja saglabāt interesējošos datus turpmākai analīzei, datu vizualizācijas iespējas, kā arī personalizēta pieredze, kas ļauj katram lietotājam veidot savu datu kolekciju atbilstoši savām interesēm un vajadzībām astronomijas izpētē.

## **1.2. Programmatūras prasību specifikācija**

Programmatūras ievaddati sevī ietver lietotāja pieprasījumus konkrētiem NASA API (APOD, Mars Rover, NEO (Near Earth Objects)), dažādus meklēšanas parametrus (datums, kategorija, misijas nosaukums utt.), lietotāja autentifikācijas datus, kā arī lietotāja saglabāto meklējumu un iecienīto objektu datus, kas tiks izmantoti personalizētai pieredzei. Savukārt izvaddati ietver astronomisko attēlu un video vizualizācijas, Marsa virsmas attēlus no dažādām misijām, statistiskos datus par tuvumā esošajiem asteroīdiem, dažādas datu vizualizācijas (grafikus, diagrammas), kā arī eksportējamus datus CSV un JSON formātos, kas ļaus lietotājiem veikt tālāku datu analīzi citās programmās.

Projektā tiks apstrādāti dažādi datu veidi, tostarp attēli (JPG, PNG formātos), video (MP4 formātā), tekstuālie dati (JSON, CSV formātos), kā arī lietotāju dati (teksts, šifrētas paroles). Galvenie datu avoti būs NASA APOD API (Astronomy Picture of the Day), NASA Mars Rover Photos API, NASA NeoWs API (Near Earth Object Web Service), kā arī lokāla SQL datubāze, kas tiks izmantota lietotāju datiem un saglabātajiem meklējumiem, nodrošinot personalizētu pieredzi un meklējumu vēstures saglabāšanu.Tehnoloģiju izpētes rezultātā izvēlēto resursu apraksts[[1]](#footnote-1)

## **1.3. Tehnoloģiju izpētes rezultātā izvēlēto resursu apraksts**

Projekta izstrādei izvēlēta Python programmēšanas valoda, ņemot vērā tās plašo bibliotēku klāstu un piemērotību datu analīzei. No Python bibliotēkām un API tiks izmantotas vairākas specializētas bibliotēkas - requests HTTP pieprasījumu veikšanai NASA API, DBBrowser kā ORM (Object-Relational Mapping) rīks datubāzes darbiem, Flask web interfeisa izveidei, Pillow (PIL) attēlu apstrādei, cryptography datu šifrēšanai, kā arī pytest automatizētai testēšanai. Attiecībā uz datubāzes risinājumiem, izstrādes procesā tiks izmantota SQLite datubāze, bet produkcijas vidē - PostgreSQL, kas nodrošinās labāku veiktspēju lielāka datu apjoma gadījumā. Projekts balstīsies uz NASA Open API (https://api.nasa.gov/), kuras izmantošanai nepieciešama API atslēga, ko var iegūt ar bezmaksas reģistrāciju NASA API portālā.

## **1.4. Datu aizsardzības pasākumi**

Lai gan šī projekta lietotne nesatur sensitīvus personas datus, tika ievēroti vairāki datu aizsardzības pamatprincipi, kas atbilst labās prakses piemēriem programmēšanā. Projekta galvenie dati, piemēram, favorītu attēlu nosaukumi, apraksti, attēlu saites un uzņemšanas datumi, tiek saglabāti lokāli SQLite datubāzē. Šie dati nav publiski pieejami un tiek uzglabāti tikai lietotāja ierīcē, tādējādi novēršot nepilnvarotu piekļuvi no ārējiem avotiem.

Datu integritātei un drošībai tika ieviesti pasākumi, lai aizsargātu datubāzi no dublikātiem un kļūdainas ievades. Piemēram, saglabājot favorītus, tiek pārbaudīts, vai konkrētais attēls jau eksistē datubāzē, lai novērstu lieku datu uzkrāšanos un datu dublēšanos. Tas uzlabo datu kvalitāti un nodrošina labāku pārvaldību.

Tā kā lietotne darbojas tikai lokāli, nav nepieciešama datu šifrēšana pārsūtīšanas laikā, taču kodā ir iespējams viegli papildināt funkcionalitāti, kas šifrētu lokāli uzglabājamos datus, ja tas būtu nepieciešams. Lietotāja datu aizsardzību nodrošina arī tas, ka lietotne neievāc nekādu personisku informāciju, nepieslēdzas ārējiem serveriem lietotāja vārdā un neglabā nekādu kontu informāciju.

Visi pieprasījumi uz NASA API ir tikai lasāmas darbības, un tajos netiek izmantota nekādu lietotāju autentifikācija vai tokeni, kas saistīti ar lietotāju identitāti. Tas padara lietotni drošu arī no privātuma viedokļa.[[2]](#footnote-2)

# **2. Detalizēts prototipa apraksts**

## **2.1. Datu uzglabāšana**

Projektā izmantojamai datubāzei ir izstrādāta detalizēta relāciju shēma *(skatīt 1. pielikumu*), kas sastāv no vairākām savstarpēji saistītām tabulām. Lietotājs vispirms veic meklēšanas darbību, ievadot atslēgvārdu grafiskajā lietotāja saskarnē. Šis ievadītais vaicājums tiek nosūtīts uz NASA publisko attēlu API, kur tiek veikta attēlu atlase, pamatojoties uz meklēšanas frāzi. Kad NASA API nosūta atpakaļ rezultātus, tie tiek apstrādāti un lietotājam parādīti pārskatāmā veidā – ar nosaukumu, īsu aprakstu un saiti uz attēlu. Ja lietotājs ir reģistrējies un piesakās sistēmā, viņam tiek dota iespēja izvēlētos attēlus saglabāt favorītos, kas tiek uzglabāti lokālā datubāzē. Papildus tam tiek veikta autentifikācija un parole tiek šifrēta, lai nodrošinātu lietotāja datu aizsardzību.

Projekts izmanto SQLite datubāzi ar divām savstarpēji saistītām tabulām – users un favorites. Tabula users satur lietotājvārdus, šifrētas paroles un lietotāju lomas (piemēram, "user" vai "admin"). Šī struktūra ļauj ne tikai autentificēt lietotāju, bet arī potenciāli diferencēt piekļuves tiesības nākotnē. Tabula favorites ir paredzēta lietotāja izvēlēto NASA attēlu saglabāšanai, un tā satur informāciju par attēla nosaukumu, URL, kā arī lietotāja ID, ar kuru šis attēls ir saistīts.

Lietotāja saskarne ir izveidota ar tkinter, kas nodrošina vienkāršu, bet funkcionālu grafisko vidi. Lietotājam tiek nodrošināti vairāki logi: sākotnējais reģistrācijas/pieteikšanās logs, meklēšanas logs, attēlu rezultātu logs un favorītu pārvaldības logs. Saskarne ir veidota tā, lai tā būtu ērta lietošanai arī lietotājiem bez tehniskām zināšanām, ar skaidrām pogām, ievadlaukiem un vizuāli atdalītām funkcijām.[[3]](#footnote-3)

## **2.2. API Apraksts**

Šī projekta ietvaros tiek izmantots NASA atvērtais attēlu un video bibliotēkas API – NASA Image and Video Library API, kas ļauj iegūt attēlus no NASA plašā digitālā arhīva. API ir publiski pieejams un neprasa autentifikāciju ar API atslēgu, kas padara to vienkārši integrējamu mācību projektos.

Lai meklētu attēlus, lietotne sūta HTTP GET pieprasījumu uz šādu adresi: https://images-api.nasa.gov/search. Pieprasījumam tiek pievienots parametru komplekts, kur vissvarīgākais ir vārds, kas apzīmē meklēšanas atslēgvārdu (piemēram, “mars” vai “apollo”). Šis parametrs ļauj lietotājam atrast attēlus, kas atbilst viņa interesēm. Papildu tam tiek norādīts parametrs media\_type=image, lai atlasītu tikai attēlus, nevis videoklipus vai audio materiālus.

Kad pieprasījums tiek nosūtīts, API atgriež datus JSON formātā. Galvenie dati, kurus lietotne izmanto, ir title (attēla virsraksts), description (apraksts), date\_created (datums, kad attēls uzņemts vai publicēts) un links.href (attēla saite). Šī informācija tiek izmantota, lai veidotu lietotāja saskarnē meklēšanas rezultātu sarakstu, kā arī sniegtu iespēju attēlu priekšskatīt, lejupielādēt vai pievienot favorītiem.

Programmā šī API funkcionalitāte ir integrēta vairākās vietās. Meklēšanas lodziņš ļauj lietotājam ievadīt atslēgvārdu, pēc kura tiek nosūtīts pieprasījums API serverim. Pēc atbildes saņemšanas, dati tiek apstrādāti un attēloti kā rezultātu saraksts ar attēlu virsrakstiem, datumu un saiti uz attēlu. Attēlus var atvērt pārlūkprogrammā, lejupielādēt uz datora vai saglabāt vietējā datubāzē kā favorītus, lai vēlāk tos apskatītu atsevišķā logā.

## **2.3. Testēšanas plāns**

Projekta testēšanas stratēģija ir visaptveroša un paredz vairākus testēšanas līmeņus, lai nodrošinātu programmatūras kvalitāti un atbilstību prasībām. Vienību testēšanas līmenī tiks testēta katra API pieprasījumu funkcija, pārbaudot dažādus ievaddatu scenārijus un robežgadījumus, lai pārliecinātos par to pareizu darbību. Papildus tam tiks testētas datu apstrādes un transformācijas funkcijas, lai nodrošinātu datu integritāti un pareizu formātu konvertāciju, kā arī autentifikācijas un drošības funkcijas, lai verificētu lietotāju autentifikācijas un autorizācijas procesus, paroles šifrēšanu un citus drošības aspektus.

Lai nodrošinātu uzticamu darbību, prototips tika rūpīgi testēts. Testēšana ietvēra pareizu meklēšanas darbību ar derīgiem un nederīgiem atslēgvārdiem, lietotāju reģistrāciju un autentifikāciju, kļūdu apstrādi, kā arī attēlu saglabāšanu favorītos. Katra funkcionalitāte tika pārbaudīta vairākos scenārijos, lai pārliecinātos par stabilitāti un datu apstrādes pareizību. Turklāt tika pārbaudīts, ka sistēma spēj apstrādāt potenciāli neparedzētus ievades gadījumus, piemēram, tukšus laukus vai nepareizas autentifikācijas mēģinājumus.[[4]](#footnote-4)

## **2.4. Programmatūras izstrādes un ieviešanas plāns**

Programmatūras izstrādes un ieviešanas plāns tika veidots, balstoties uz klasisko programmatūras dzīves ciklu, kas ietver problēmas definēšanu, prasību specifikāciju, dizainu, izstrādi, testēšanu un ieviešanu. Projekta pirmajā posmā tika veikta problēmsituācijas izpēte un noteikta mērķauditorija – skolēni, studenti vai citi interesenti, kuri vēlas piekļūt NASA attēliem vienkāršā un strukturētā veidā. Balstoties uz šo, tika definēti lietotāja vajadzībām atbilstoši uzdevumi – meklēt attēlus, apskatīt tos, lejupielādēt vai saglabāt favorītos.

Otrajā posmā tika izstrādāta programmatūras prasību specifikācija, iekļaujot detalizētu informāciju par ievaddatiem (lietotāja meklēšanas atslēgvārds), izvaddatiem (attēlu nosaukumi, apraksti, datumi, saites), izmantotajām tehnoloģijām (NASA API, SQLite datubāze, Tkinter) un datu aizsardzības aspektiem. Tika izvēlēts izmantot objektorientētu programmēšanas pieeju, izveidojot NASASearch klasi, kas nodrošina API pieprasījumu un datu apstrādi, tādējādi strukturējot programmas kodu viegli uzturamā veidā.

Trešajā posmā tika veikta lietotāja interfeisa izstrāde, izmantojot Tkinter bibliotēku. Tā ietvēra ievades lauku, rezultātu attēlošanas zonu, pogas attēla lejupielādei un favorītu saglabāšanai, kā arī atsevišķu logu favorītu pārskatīšanai. Šajā posmā tika arī izstrādāta datubāzes struktūra – izveidota viena tabula ar laukiem nosaukumam, aprakstam, attēla URL un uzņemšanas datumam. Paralēli norisinājās funkciju implementācija datu iegūšanai, glabāšanai un attēlošanai.[[5]](#footnote-5)

Noslēgumā tika veikta lietotnes ieviešana – tai tika pievienoti skaidrojoši komentāri, lietotāja ceļvedis un dokumentācija. Lietotne ir pilnībā funkcionāla un gatava demonstrācijai. Ieviešanas posmā lietotne tika arī sagatavota tālākai iespējai to uzlabot vai papildināt ar jaunu funkcionalitāti, piemēram, meklēšanas filtriem pēc datuma vai misijas, autentifikācijas sistēmu vai papildus API integrāciju.

# **3. Izstrādes un testēšanas dokumentēšana**

## **3.1. Algoritmu, datu struktūru un bibliotēku izvēles pamatojums**

Šajā projektā izmantoto algoritmu izvēle balstījās uz nepieciešamību efektīvi izgūt un attēlot NASA publiskos datus, nodrošinot lietotājam iespēju meklēt un skatīt attēlus ar saprotamu un ātri reaģējošu saskarni. Galvenais algoritmiskais kodols ietver datu izgūšanu no NASA API, datu filtrēšanu un attēlu ielādi ar asinhronu apstrādi (izmantojot requests bibliotēku). Tas ļauj lietotnei vienkāršā veidā nosūtīt HTTP pieprasījumus, kas minēts rakstā no *Pedro Pregueiro*, un analizēt saņemto JSON atbildi.[[6]](#footnote-6)

Datubāzes pārvaldībai tika izmantota SQLite tehnoloģija, kas ir piemērota nelieliem un lokāliem projektiem, jo tā neprasa atsevišķu serveri un tiek glabāta kā vienkāršs fails. Tā ir integrēta Python valodā un tika viegli sasaistīta ar lietotni. Favorītu attēlu glabāšanai tika izveidota tabula ar loģisku struktūru – laukiem nosaukumam, aprakstam, attēla URL un uzņemšanas datumam, kas ļauj saglabāt un pārvaldīt attēlus efektīvā un pārskatāmā veidā.

No bibliotēkām tika izmantota tkinter, kas ir standarta Python bibliotēka lietotāja saskarņu izstrādei. Tā ļauj izveidot grafisko interfeisu bez papildu atkarībām, kas ir ideāli piemērots skolas vai mācību projektiem. Lietotāja saskarne tika papildināta ar vairākām pogām, attēlu rāmi un ievades laukiem, kas nodrošina funkcionalitāti un lietojamību. Attēlu apstrādei tika izmantota arī PIL (Pillow), kas atvieglo attēlu izmēru pielāgošanu, lai tie pareizi parādītos lietotnes logā.2

Visbeidzot, kods strukturēts, izmantojot objektorientēto programmēšanu (OOP). Galvenā klase NASASearch organizē datu izgūšanu un nodrošina metodes datu apstrādei. Šāda pieeja ļauj kodu strukturēt skaidrāk, uzlabot tā pārskatāmību un padara to vieglāk uzturamu un paplašināmu nākotnē. Koda sadalīšana klasēs un funkcijās ļauj arī [[7]](#footnote-7)vienkāršāk testēt katru daļu atsevišķi.

## **3.2. Testēšanas pārskats**

**Atkļūdošana:**

* Problēma: Meklēšanas rezultāti netika parādīti logā.

Risinājums: Tika pārbaudīta API atbilde – izrādījās, ka meklējamais vārds atgrieza tukšu items sarakstu. Pievienota kļūdu apstrāde, lai informētu lietotāju, ja rezultātu nav.

* Problēma: Favorītu pievienošanas laikā programma avarēja, ja netika atrasts attēla URL.

Risinājums: Tika ieviests nosacījums, kas pārbauda, vai attēla URL eksistē pirms datubāzes ievades.

* Problēma: Tkinter logs bija norganizēts, un pogas pārklājās.

Risinājums: Sakārtots grid() un pack() izkārtojums katram rāmim, nodrošinot vienotu un pārskatāmu GUI.

**Akcepttestēšana:**

* Apraksts: Meklēšanas funkcija ar atslēgvārdu “Mars”.

Rezultāts: Atrada un attēloja rezultātus.

* Apraksts: Favorīta pievienošana ar klikšķi uz pogas.

Rezultāts: Ieraksts veiksmīgi pievienots datubāzē.

* Apraksts: Atvērts “Skatīt favorītus” logs.

Rezultāts: Parādīja visus saglabātos attēlus

* Apraksts: Attēla pievienošana un pēc tam atkārtota pievienošana.

Rezultāts: Netiek pievienots dublikāts.

# **Licence**

Programmatūra tiek izplatīta saskaņā ar MIT licenci. Tas nozīmē, ka:

* Programmatūru var brīvi izmantot, modificēt un izplatīt
* Nepieciešams saglabāt oriģinālo autora paziņojumu
* Programmatūra tiek piedāvāta "kā ir", bez jebkādām garantijām

NASA API dati ir pakļauti NASA datu izmantošanas noteikumiem, kas pieejami: <https://api.nasa.gov/>

# **Lietotāja ceļvedis**

Šī NASA attēlu meklēšanas lietotne ir vienkārši lietojama programma, kas ļauj lietotājam meklēt NASA publiski pieejamos attēlus, izmantojot atslēgvārdus. Programmas palaišanai nepieciešams *Visual Studio Code*, *Python 3* un dažas bibliotēkas, ko var uzstādīt ar “*pip install requests Pillow”*.

**Programmas palaišana:**

* Pārliecinies, ka datorā ir ieinstalēts un uzstādīts **Visual Studio Code** un **Python 3**.
* Atver main.py failu un terminālī instalē vajadzīgās bibliotēkas ar komandu:

*pip install requests Pillow*

* Palaid main.py failu, uzsākot kodu.

**Attēlu meklēšana:**

* Ievadi meklējamo vārdu (piem. “Mars”) teksta laukā.
* Spied pogu **“Meklēt”**.
* Attēli tiks ielādēti no NASA datubāzes un uzklikšķinot pogu **“Atvērt attēlu pārlūkā”** tiks atvērts attēls jaunā interneta logā.

**Attēla lejupielāde:**

* Spied pogu **“Lejupielādēt attēlu”**.
* Attēls tiks saglabāts uz datora mapē, kur atrodas tavs *main.py* fails.

**Favorītu saglabāšana:**

* Spied pogu **“Pievieniot favorītiem”** zem izvēlētā attēla.
* Attēla dati tiks ierakstīti datubāzē, kur atrodas tavs *main.py* fails.

**Favorītu apskate:**

* Spied pogu **“Skatīt favorītus”**.
* Tiks atvērts jauns logs ar visiem saglabātajiem attēliem.

**Padomi:**

* Izmanto vienkāršus atslēgvārdus (angļu valodā), piemēram: “moon”, “earth”, “apollo”.
* Favorītu sarakstu var tīrīt, izdzēšot attiecīgus ierakstus no datubāzes.

# **Secinājumi**

Projekta izstrādes gaitā tika veiksmīgi sasniegti izvirzītie mērķi un uzdevumi – izveidota funkcionāla Python programmatūra, kas izmanto NASA publisko API, lai lietotājam sniegtu iespēju meklēt un skatīt NASA attēlus pēc izvēlēta atslēgvārda. Programmatūras izstrāde tika veikta atbilstoši klasiskajam programmatūras dzīves cikla modelim, sākot ar problēmas izpēti un mērķauditorijas vajadzību noteikšanu, turpinot ar prasību specifikāciju, risinājuma projektēšanu, realizāciju, testēšanu un noslēdzot ar produkta demonstrēšanu.

Projekts parāda skolēna spējas praktiski pielietot programmēšanas II kursā iegūtās zināšanas – tika izmantotas vairākas Python bibliotēkas (requests, sqlite3, cryptography), objektorientētā programmēšana ar klasēm, kā arī izstrādāta datubāze ar saistītām tabulām, kurā tiek saglabāti lietotāju dati un viņu favorītattēli.

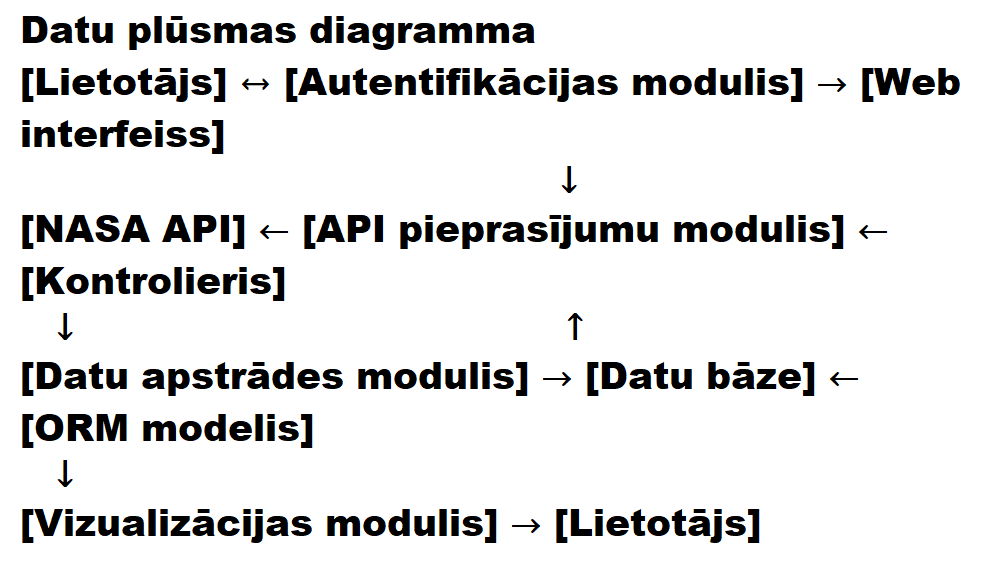
Projekta izstrādes gaitā tika veiksmīgi īstenota ideja izveidot lietotni, kas izmanto NASA publiski pieejamo attēlu meklēšanas API, ļaujot lietotājam interaktīvā un saprotamā veidā piekļūt kosmosa attēliem. Projekts apliecina prasmes darbā ar ārējiem tīmekļa servisiem (API), datu struktūrām, datubāzu integrēšanu un objektorientētu programmēšanu, vienlaikus attīstot arī lietotāja interfeisa veidošanu ar Tkinter bibliotēku.

Lietotnē izmantotā datubāze parāda prasmi saglabāt un strukturēt datus ilgtermiņā, bet iespēja lejupielādēt attēlus pierāda failu sistēmas darbības izpratni. Tika veikti arī datu aizsardzības pasākumi un ievēroti programmatūras izstrādes labās prakses principi.

# **Izmantotie avoti**

1. Robert Nemiroff & Jerry Bonnell. *Astronomy Picture of the Day* [tiešsaiste]. [Skatīts: 03.03.2025.] Pieejams:<https://apod.nasa.gov/apod/astropix.html>
2. Daniel Crendon [04.05.2020.]. *Apod-api* [tiešsaiste]. Pieejams: <https://github.com/nasa/apod-api>
3. NASA. *NASA APIs* [tiešsaiste]. [Skatīts: 03.03.2025.] Pieejams: <https://api.nasa.gov/index.html#browseAPI>
4. ALTA3 Research [10.12.2018.]. *Python Tutorial - Python with APIs: Getting a Nasa Picture of the Day* [video]. YouTube.com. Pieejams: <https://www.youtube.com/watch?v=UetUm0q0sBE&t=274s>
5. Pedro Pregueiro [01.02.2025.]. *Python & APIs: A Winning Combo for Reading Public Data* [tiešsaiste]. Pieejams: <https://realpython.com/python-api/>
6. Corey Schafer [18.04.2017.]. *Python SQLite Tutorial: Complete Overview - Creating a Database, Table, and Running Queries* [video]. YouTube.com. Pieejams: <https://www.youtube.com/watch?v=pd-0G0MigUA>
7. ASA Learning [29.11.2020.]. *Data Encryption & Decryption | Cryptography Python | Password with Fernet | Explained in Minutes* [video]. YouTube.com. Pieejams: <https://www.youtube.com/watch?v=-3qXJZEe7Lo>
8. Python Documentation. *Tkinter — Python interface to Tcl/Tk* [tiešsaiste]. Pieejams: <https://docs.python.org/3/library/tkinter.html>
9. Wilsjame. *How To Use NASA APIs* [tiešsaiste]. Github.com. Pieejams: <https://wilsjame.github.io/how-to-nasa/>
10. Andrea L. [11.12.2023.]. *Coding the Universe: Building a Space-themed Website Using NASA’s API* [tiešsaiste]. Pieejams: <https://medium.com/@n01597125/coding-the-universe-building-a-space-themed-website-using-nasas-api-case-study-235f6dc0cf4f>
11. Educative [01.10.2020.]. *How to use an API: fetch daily images with NASA's Open API* [tiešsaiste]. Pieejams: <https://www.educative.io/blog/how-to-use-api-nasa-daily-image>
12. SimeonOnSecurity [01.08.2023.]. *Python Security Best Practices: Protecting Your Code and Data* [tiešsaiste]. Pieejams: <https://simeononsecurity.com/articles/python-security-best-practices-protecting-code-data/>

# **Pielikumi**



*1. pielikums*

1. *Educative [01.10.2020.]. How to use an API: fetch daily images with NASA's Open API [tiešsaiste]. Pieejams:* [*https://www.educative.io/blog/how-to-use-api-nasa-daily-image*](https://www.educative.io/blog/how-to-use-api-nasa-daily-image) [↑](#footnote-ref-1)
2. *SimeonOnSecurity [01.08.2023.]. Python Security Best Practices: Protecting Your Code and Data [tiešsaiste]. Pieejams: https://simeononsecurity.com/articles/python-security-best-practices-protecting-code-data/* [↑](#footnote-ref-2)
3. *Python Documentation. Tkinter — Python interface to Tcl/Tk [tiešsaiste]. Pieejams:* [*https://docs.python.org/3/library/tkinter.html*](https://docs.python.org/3/library/tkinter.html) [↑](#footnote-ref-3)
4. *OpenAI (02.05.2025). [Large Language Model]. Pieejams:* [*https://chat.openai.com/*](https://chat.openai.com/) [↑](#footnote-ref-4)
5. *Python Documentation. Tkinter — Python interface to Tcl/Tk [tiešsaiste]. Pieejams:* [*https://docs.python.org/3/library/tkinter.html*](https://docs.python.org/3/library/tkinter.html) [↑](#footnote-ref-5)
6. *Pedro Pregueiro [01.02.2025.]. Python & APIs: A Winning Combo for Reading Public Data [tiešsaiste]. Pieejams:* [*https://realpython.com/python-api/*](https://realpython.com/python-api/) [↑](#footnote-ref-6)
7. [↑](#footnote-ref-7)