Katalóg požiadaviek

GUI rozhranie na riadenie astronomického ďalekohľadu na pozorovanie kozmického odpadu

**1.**          **Úvod**

**1.1.**       **Účel katalógu požiadaviek**

Tento dokument, katalóg požiadaviek slúži na jednoznačné zaznamenanie všetkých požiadaviek zadávateľa projektu grafického rozhrania   na riadenie astronomického ďalekohľadu na pozorovanie vesmírneho odpadu. Dokument bol vytvorený na základe stretnutia so zadávateľom a následnej komunikácie. Je určený všetkým osobám, ktoré sa zapájajú do riešenia  tohto projektu alebo budú používať výsledný systém. Katalóg obsahuje úplný zoznam požiadaviek i všeobecný opis, z ktorého je jasné ako bude systém fungovať a preto tento katalóg tiež slúži aj ako záväzná dohoda medzi zadávateľom a vývojármi.

**1.2.**       **Rozsah využitia systému**

Cieľom je vytvoriť grafické rozhranie pre už existujúci systém. Toto rozhranie má umožňovať plne kontrolovať existujúci systém. To znamená, má umožniť užívateľovi manipulovať s ďalekohľadom, jeho kupolou, kamerou a filtrovým kolesom. Grafické rozhranie sa ale nestará o hardvérovú realizáciu a všetky používateľom zadané údaje iba posúva existujúcej aplikácii.  Okrem toho, rozhranie má zobrazovať dostupné informácie o polohe ďalekohľadu, stave vyhotovenej  snímky či o polohe plánovaného objektu na snímke.

**1.3.**       **Slovník pojmov a skratiek**

**Azimut -**Horizontálna súradnica meraná v stupňoch (0 – 360) v smere pohybu hodinových ručičiek od južného bodu.

**Elevation -**Vertikálna súradnica meraná v stupňoch (0 – 90) vyjadrujúca výšku nad horizontom.

**RA -**Right ascension alebo rektascenzia, vzdialenosť od jarného bodu pozdĺž rovníka smerom na východ. Meria sa v uhloch a spolu s deklináciou tvorí rovníkovú sústavu súradníc.

**Declination -**Súradnica vyjadrujúca uhlovú vzdialenosť od svetového rovníka.

**Filterwheel -**Filtrové koleso nastavujúce filter pre kameru. Obsahuje 5 rôznych nastaviteľných polôh, resp. filtrov.

**FITS -**Flexible image transport system - štandardný astronomický formát súboru.

**Zenit -**alebo nadhlavník je poloha hviezdy alebo iného nebeského telesa priamo nad pozorovateľom

**1.4.**       **Referencie**

Implementácia klávesových skratiek bude spravená podľa zadávateľom vypracovaného dokumentu [PopisBackend](https://drive.google.com/file/d/1eL3ljwdQ4YPdasL6I0C6yksuLwgg_b-M/view?usp=sharing" \t "_blank). Pri návrhu grafického rozhrania sa využije okomentovaný screenshot pôvodného konzolového rozhrania. Zadávateľ tiež dodá presný popis formátu JSON objektu obsahujúceho aktuálny stav ďalekohľadu a kód emulujúci správanie backendu, bez ktorých nedokážeme vyvinúť funkčné GUI.

**1.5.**       **Prehľad nasledujúcich kapitol**

Druhá kapitola ponúka všeobecný opis systému, keď najprv objasňuje všeobecnú perspektívu systému. Celkový prehľad funkcionality čitateľ nájde v časti Funkcie systému. Okrem toho druhá kapitola ponúka charakteristiku používateľa, všeobecné obmedzenia systému a nakoniec ozrejmuje dohodnuté predpoklady a závislosti. Tretia kapitola obsahuje zoznam požiadaviek rozdelený do podkapitol podľa jednotlivých blokov, ktorých sa dané požiadavky týkajú.

**2.**          **Všeobecný opis**

**2.1.**       **Perspektíva systému**

Cieľom projektu je implementácia grafického rozhrania pre ovládanie astronomického ďalekohľadu, ktorá má primárne zefektívniť a zjednodušiť samotnú prácu s ním a tým zároveň zredukovať počet chýb vznikajúcich v dôsledku zlyhania ľudského faktoru. Výsledná aplikácia má zároveň upozorňovať užívateľa pri nevhodne zvolenej konfigurácii ďalekohľadu, ktorá môže mať nepriaznivé účinky na jeho chod.

**2.2.     Funkcie systému**

Systém bude komunikovať ako klient so serverovým modulom, ktorého konkrétna implementácia pre nás nie je známa, prostredníctvom stanoveného sieťového protokolu a výhradne na lokálnej sieti.

Komunikácia bude spočívať v obojsmernej výmene dát. Dáta posielané zo serveru ku klientovi budú obsahovať:

* informácie o aktuálnom stave konfigurácie ďalekohľadu, ktoré klient — náš systém — spracuje a vhodnou formou zobrazí užívateľovi v grafickom rozhraní
* poslednú vykonanú snímku ďalekohľadom, ktorej náhľad sa má zobraziť taktiež v grafickom rozhraní

Klient bude odosielať serveru údaje ohľadom zmien v konfigurácii ďalekohľadu zadaných užívateľom, ktoré pred odoslaním budú validované.

V grafickom rozhraní sa budú taktiež nachádzať jednoduché grafiky, ktoré budú predstavovať:

* aktuálny smer ďalekohľadu, ktorý sa zobrazí pomocou dvoch grafík, teda sklonom nad horizontom ďalekohľadu a azimutálnym zobrazením
* smerovanie štrbiny kupoly

Do grafického rozhrania sa zapracujú zároveň upozornenia pre užívateľa pri nevhodne zvolenej konfigurácii ďalekohľadu.

**2.3      Charakteristika používateľa**

Aplikácia bude slúžiť iba Oddeleniu Astronómie a Astrofyziky KAFZM nakoľko je grafickým rozšírením už existujúcich neverejných systémov používaných výhradne týmto oddelením pričom všetci užívatelia budú mať rovnaké práva.

**2.4      Všeobecné obmedzenia**

Aplikácia vyžaduje, aby na lokálnej sieti bežal vyššie popísaný serverový modul, s ktorým bude musieť podľa stanoveného protokolu komunikovať. V jednom momente môže byť spustená iba jedna inštancia.

**2.5      Predpoklady a závislosti**

Dohodnutým programovacím jazykom na vývoj aplikácie je Java vo verzii 11, je teda taktiež nevyhnutné, aby bol na systéme prítomný príslušný java runtime environment. Aplikácia taktiež vyžaduje, aby na lokálnej sieti bežal vyššie popísaný serverový modul.

**3. Zoznam požiadaviek**

Jednotlivé nastavenia sú iba sprostredkúvane backend-u a preto ich význam nie je potrebné v tomto dokumente opisovať detailne.

**3.1 Všeobecné požiadavky** .

3.1.1 Spustenie

Systém po spustení zobrazí hlavnú stránku grafického rozhrania s aktuálnymi údajmi ak sa podarí pripojiť na server. Inak automaticky dokola opakuje pokus o pripojenie a informuje, že nie je pripojený.

3.1.2 Prihlasovanie

Systém nevyžaduje prihlasovanie užívateľov.

3.1.3 Rozmiestnenie jednotlivých blokov

Umiestnenie jednotlivých blokov v grafickom rozhraní má zohľadňovať ich umiestnenie v pôvodnej konzolovej aplikácii. Nemusí ale byť totožné.

3.1.4.1 Nastavenia parametrov

Všetky parametre, ktoré sú nastaviteľné klávesovou skratkou, bude možné nastaviť aj kliknutím na príslušné tlačidlo (pre parametre, ktoré nevyžadujú vstup) alebo vyplnenia príslušného textového bloku (pre parametre, ktoré vyžadujú aj vstupnú hodnotu) a následne kliknutie na tlačidlo odoslať.

3.1.4.2 Odosielanie nastavených hodnôt (Optional)

Bude možné zmeniť/nastaviť viac parametrov súčasne a poslať ako skupinu hodnôt.

3.1.5 Náhľad FITS snímky

Systém bude zobrazovať náhľad poslednej FITS snímky. Zároveň aj cestu k poslednej FITS snímke dodanú JSON štruktúrou. Táto snímka bude mať rozmer štvorca.

3.1.6 Zoznam a vysvetlenie klávesových skratiek

Systém poskytne zoznam všetkých klávesových skratiek aj s ich vysvetlením. Zoznam ale nebude zobrazovaný stále, ale iba po rozkliknutí.

3.1.7 Zobrazovanie času

Systém zobrazí dátum a čas podľa prijatej JSON štruktúry.

3.1.8 Výstražné správy

Systém upozorní vyskakovacím oknom ak sa niektorá časť (Camera, Dome) zmení status z “Ready” na “Not responding”. Jeho ukončenie nie je ničím podmienené, stačí kliknúť na tlačidlo podobné “OK”.

3.1.9 Status pripojenia na backend

Systém neustále zobrazuje stav pripojenia na backend. Stavy, ktoré môžu nastať, sú Connected a Disconnected.

3.1.10 Načítanie schedulera

Bude k dispozícii tlačidlo Load scheduling, ktorý spustí skript a progress bar vykonávanie skriptu. Progress bar buda nastavovaný na základe výstupu skriptu.

3.1.11 Spustenie a vypnutie schedulera

Scheduler (plánovač) spustíme klávesou “J” a vypneme klávesou “K”. Zároveň bude k dispozícii tlačidlo Start/Stop scheduler.

3.1.12 Časti používateľského rozhrania

Používateľské rozhranie bude rozdelené do častí, resp. blokov: Polar axis, Declination axis, Camera, Filterwheel, Dome a Target

3.1.13 Ukončenie programu

Program sa ukončí štandardným spôsobom. Pri ukončení sa neukladajú navyše žiadne dáta.

3.1.14 Zobrazenie celooblohových snímok

Program zobrazí náhľad snímky z celooblohových kamier. Táto snímka je uložená vo formáte JPG na tom istom servery ako FITS snímka. Cesta k nej je dodaná JSON štruktúrou.

3.1.15 Veľkosť GUI

Veľkosť GUI bude zafixované na rozmere 1200x700px.

**3.2 Zobrazenie a nastavenie parametrov bloku Camera**

3.2.1 Zobrazenie informácii o stave kamery

Z údajov v štruktúre JSON, ktorý backend posiela každých 60ms sa zobrazia nasledujúce údaje:

* RBI funkcia počet typu string v tvare “Ax”, kde ‘A’ je kladné celé číslo a ‘x’ je koncový charakter
* RBI funkcia čas typu string v tvare “A.Bs”, kde ‘A’ a ‘B’ sú kladné celé čísla, ‘.’ je oddeľovací charakter a ‘s’ je koncový charakter
* TDI mód typu string v tvare “on” alebo “off”
* Flush pozadia typu string v tvare “on” alebo “off”
* Binovanie typu string a v tvare „AxB“, kde ‘A’ a ‘B’ sú celé kladné čísla a ‘x’ je oddeľovací charakter.
* Veľkosť subframe typu string v tvare „A,B“, kde ‘A’ a ‘B’ sú kladné celé čísla a ‘,’ je oddeľovací charakter
* Súčasná teplota kamery typu string v tvare “A°”, kde ‘A’ je reálne číslo zaokrúhlené na 1 desatinné miesto.

3.2.2 Typ snímky

Systém zobrazí aktuálne nastavený typ snímky prečítaním údajov z poslednej JSON štruktúry. Typ bude možné nastaviť klávesovou skratkou “t”, ktorá premení typ na nasledujúci z troch možných typov, resp. na prvý. Používateľ nič ďalšie nezadáva. Typy snímky môžu byť : LIGHT, DARK a FLAT

3.2.3 Čas expozície

Systém zobrazí čas expozície v sekundách prečítaním údajov z poslednej JSON štruktúry. Čas bude možné nastaviť klávesovou skratkou “e”, kde používateľ zadá hodnoty väčšie ako 0 (double prompt)

3.2.4 Mód kamery

Systém zobrazí nastavený mód kamery prečítaním údajov z poslednej JSON štruktúry. Mód bude možné nastaviť klávesovou skratkou “m”, ktorá premení mód na nasledujúci z troch možných módov. Používateľ  ďalšie nič nezadáva. Módy kamery môžu byť: 1MHz low noise, 3.4MHz low noise, 1MHz high dynamic range

3.2.5 FITS Observer

Systém zobrazí informácie o pozorovateľoch v tvare „M.Priezvisko“. Pozorovateľov bude možné nastaviť klávesovou skratkou “O”, kde používateľ zadá mená pozorovateľov oddelených čiarkou (string prompt). Maximálna dĺžka údajov nie je limitovaná, ale v GUI bude zobrazených maximálne 50 charakterov. Celé údaje sa zobrazia až po rozkliknutí.

3.2.6 FITS Objekt

Systém zobrazí informácie o objekte. Objekt bude možné nastaviť klávesovou skratkou “o”, kde používateľ zadá názov pozorovaného objektu (string prompt). Maximálna dĺžka údajov nie je limitovaná, ale v GUI bude zobrazených maximálne 50 charakterov. Celé údaje sa zobrazia až po rozkliknutí.

3.2.7 FITS poznámky

Systém zobrazí vytvorené poznámky. Poznámky bude možné nastaviť klávesovou skratkou “n”, kde používateľ zadá ľubovoľné poznámky k pozorovanému objektu (string prompt). Maximálna dĺžka poznámok nie je limitovaná, ale v GUI bude zobrazených maximálne 50 charakterov. Celé poznámky sa zobrazia až po rozkliknutí poznámok.

3.2.8 Želaná teplota kamery

Systém zobrazí želanú teplotu kamery, ktorú chceme dosiahnuť v stupňoch Celzia, prečítaním údajov z poslednej JSON štruktúry. Želanú teplotu kamery v stupňoch Celzia bude možné nastaviť klávesovou skratkou “s”, ktorú bude možné zadať pre všetky R (reálne čísla), ako aj negatívne hodnoty (double prompt) 

3.2.9 Delay medzi snímkami

Systém zobrazí delay(meškanie) medzi jednotlivými snímkami prečítaním údajov z poslednej JSON štruktúry v sekundách. Meškanie bude možné nastaviť klávesovou skratkou “d”, kde používateľ zadá hodnotu väčšiu ako 0 (double prompt)

3.2.10 Počet ostávajúcich snímok a času v expozícii

Systém zobrazí zostávajúci čas v expozícii v sekundách, ktorý bude vyčítaný z poslednej JSON štruktúry a zároveň počet ostávajúcich snímok . Počet opakovaní snímkovania bude možné nastaviť klávesovou skratkou “r”, kde používateľ zadá kladné celé číslo väčšie ako 0 (int prompt)

3.2.11 Zapnutie kamery

Systém umožní zapnúť expozíciu klávesovou skratkou “E”. Klávesové skratky “X” a “A” sú analogické k “E”

**3.3 Zobrazenie a nastavenie parametrov bloku Filterwheel**

3.3.1 Filter

Systém zobrazí polohu filtrového kolesa.  Poloha filtrového kolesa sa bude môcť nastaviť na : „B V R I C“, čo je zoznam všetkým možných polôh filtrového kolesa. Klávesové skratky na nastavenie filtra budú “B” = filter B a ostatné “V”,”R”,”I”,”C” analogicky ako pre “B”.

3.3.2 Upozornenie pri menení filtra

Keďže výmena filtra chvíľu trvá, systém používateľa upozorní na to, že prebieha výmena filtra

3.3.3 Farebné rozlíšenie polôh filtra

Písmenká jednotlivých filtrov budú zobrazované v zodpovedajúcej farbe. B – blue, V – green, R – red, I – infrared (orange )a C – clear (white)

**3.4 Zobrazenie a nastavenie parametrov bloku Target**

V aktuálne používanej konzolovej aplikácii sa hodnoty z bloku Target nevyužívajú.

3.4.1 Zobrazenie informácii o Target

Z údajov v štruktúre JSON, ktorý backend posiela každých 60ms sa zobrazia nasledujúce údaje:

* Encoder typu string v tvare “A”, kde ‘A’ je reálne číslo
* DEnc typu string v tvare “A”, kde ‘A’ je reálne číslo
* HA apparent typu string v tvare “AhBmCs”, kde ‘A’ a ‘B’ sú celé čísla, ‘C’ je reálne číslo, ‘h’ a ‘m’ sú oddeľovacie charaktery a ‘s’ je koncový charakter
* RA J2000 typu string v tvare “AhBmCs”, kde ‘A’ a ‘B’ sú celé čísla, ‘C’ je reálne číslo, ‘h’ a ‘m’ sú oddeľovacie charaktery a ‘s’ je koncový charakter
* DE apparent typu string v tvare “ A°B’C’’ ”, kde ‘A’,’B’ a ‘C’ sú celé čísla a ’ ° ’ a ’ ‘ ’ sú oddeľovacie charaktery a ‘ ‘’ ‘ koncový charakter
* DE J2000 typu string v tvare “ A°B’C’’ ”, kde ‘A’,’B’ a ‘C’ sú celé čísla a ’ ° ’ a ’ ‘ ’ sú oddeľovacie charaktery a ‘ ‘’ ‘ koncový charakter
* Azimut typu string v tvare “A°”, kde ‘A’ je reálne číslo a ‘°’ je koncový charakter
* Elevation typu string v tvare “A°”, kde ‘A’ je reálne číslo a ‘°’ je koncový charakter
* Pole crossing string “on” alebo “off” podľa toho či je pole crossing zapnutý
* Status typu string, kde bude informácia v sekundách s približným časom, kedy sa budeme nachádzať na poli

3.4.2 Pole crossing

Systém zobrazí, či prechádzame cez pole v hodnotách on/off (áno/nie) z údajov v JSON štruktúre. Klávesovou skratkou “p” vieme nastaviť pole crossing na hodnoty “on” alebo “off”

3.4.3 Nastavenie RA a DE J2000

Systém umožní nastaviť hodnoty RA a DE klávesovou skratkou “L”, kde používateľ zadá 2 hodnoty typu string, kde prvá nastaví RA a druhá DE. Obe hodnoty môžu byť vo formáte DD.DDD alebo HH:MM:SS.SSS, kde D,H,M a S sú celé kladné čísla.

3.4.4 Nastavenie a spustenie cieľa GoTo funkcia

Systém umožní klávesovou skratkou “G” spustiť automatický pohyb alebo zastaviť pohyb na v minulosti zadaný cieľ

**3.5 Zobrazenie a nastavenie parametrov bloku Polar axis**

3.5.1 Zobrazenie informácií o stave ďalekohľadu, ktoré sa týkajú motorov polar axis

Z údajov v štruktúre JSON, ktorý backend posiela každých 60ms sa zobrazia nesledujúce údaje:

* Informácie z enkóderu, reálne číslo, napr. -0.0417
* zjavný hodinový uhol ďalekohľadu vo formáte J2000, napr. +00h05m45.7s
* rektascenzia ďalekohľadu vo formáte J2000
* azimut ďalekohľadu v stupňoch, napr. 234.3°
* status enkóderu - string v tvare “ready“ alebo “not responding“

3.5.2 Grafika

Systém graficky (na kompase) zobrazí azimutálne zobrazenie ďalekohľadu. Ak kupola a ďalekohľad nie sú zosynchronizované, okolo grafiky bude červený rámček.

3.5.3 Nastavenie a spustenie motoru v RA smere

* Left (šípka vľavo) - po stlačení klávesy sa dá zadať hodnota od 1 - 250 000 a po potvrdení (príslušný button) sa pohne motor v RA smere.
* Right (šípka vpravo) - po stlačení klávesy sa dá zadať hodnota od 1 - 250 000 a po potvrdení (príslušný button) sa pohne motor v RA smere, ale opačným smerom.
* END - po stlačení klávesy sa zastavia RA aj DE motory.
* PageDown - po stlačení klávesy sa zastaví RA motor.
* [ - enable motorov RA a DE, t.j. motory budú pripravené sa hýbať.
* ] - disable motorov RA a DE
* c- kalibrácia polohy ďalekohľadu podľa poslednej redukovanej snímky.
* z- kalibrácia polohy ďalekohľadu na zenit

**3.6 Zobrazenie a nastavene parametrov bloku Declination axis**

3.6.1 Zobrazenie informácií o stave ďalekohľadu, ktoré sa týkajú motorov declination axis

Z údajov v štruktúre JSON, ktorý backend posiela každých 60ms sa zobrazia nesledujúce údaje:

* informácie z enkóderu - reálne číslo, napr. -0.0417
* zjavná deklináciu ďalekohľadu vo formáte J2000, napr. +00h05m45.7s
* deklináciu ďalekohľadu vo formáte J2000
* sklon ďalekohľadu v stupňoch, napr. 74.3°
* status enkóderu - string v tvare “ready“ alebo “not responding“

3.6.2 Grafika

Systém graficky zobrazí sklon nad horizontom ďalekohľadu. Ak je sklon ďalekohľadu menší ako 25 stupňov, okolo grafiky bude červený rámček.

3.6.3 Nastavenie a spustenie motoru smere osi DE

* Up (šípka hore) - po stlačení klávesy sa dá zadať hodnota od 1 - 250 000 a po potvrdení (Enter) sa pohne motor v smere osi DE.
* Down (šípka dole) - po stlačení klávesy sa dá zadať hodnota od 1 - 250 000 a po potvrdení (príslušne tlačidlo) sa pohne motor v smere osi DE, ale v opačnom smere.
* END - po stlačení klávesy sa zastavia RA aj DE motory.
* Delete - po stlačení klávesy sa zastaví DE motor.
* [ - enable motorov RA a DE, t.j. motory budú pripravené sa hýbať.
* ] - disable motorov RA a DE
* c- kalibrácia polohy ďalekohľadu podľa poslednej redukovanej snímky.
* z- kalibrácia polohy ďalekohľadu na zenit.
* T - automatická korekcia ďalekohľadu, ak sa dostal do nebezpečnej výšky.

**3.7 Zobrazenie a nastavenie parametrov bloku Dome**

3.7.1 Zobrazenie informácií o stave ďalekohľadu, ktoré sa týkajú kupoly

Z údajov v štruktúre JSON, ktorý backend posiela každých 60ms sa zobrazia nesledujúce údaje:

* informácie z enkóderu - reálne číslo, napr. -0.0417
* azimut štrbiny kupoly v stupňoch, napr. 0.0°
* želaný azimut štrbiny kupoly v stupňoch
* informáciu o zosynchronizovaní kupoly s kamerou - string v tvare “off  alebo “on“
* status pripojenia kupoly - string v tvare “ready“ alebo “not responding“

3.7.2 Grafika

Systém graficky (na kompase) zobrazí azimutálne zobrazenie štrbiny kupoly. Ak kupola a ďalekohľad nie sú zosynchronizované, okolo grafiky bude červený rámček.

3.7.3 Nastavenie a spustenie pohybu kupoly

* Insert - po stlačení klávesy sa kupola začne hýbať na východ,  okrem toho používateľ nezadáva žiadny číselný vstup.
* PageUp - po stlačení klávesy sa kupola začne  hýbať na západ,  okrem toho používateľ nezadáva žiadny číselný vstup.
* Home - po stlačení klávesy sa kupola prestane hýbať.
* f- po stlačení klávesy používateľ zadá reálne číslo v rozmedzí (0, 60), podľa ktorého sa nastaví rýchlosť kupoly.
* y- po stlačení klávesy sa spustí synchronizácia kupoly.
* a- po stlačení klávesy používateľ zadá reálne číslo v rozmedzí (0, 360), podľa ktorého sa nastaví želaný azimut kupoly.

  Návrh softvéru

GUI rozhranie na riadenie astronomického ďalekohľadu na pozorovanie kozmického odpadu

1. Úvod
2. Účel dokumentu

Tento dokument poskytuje úplný návrh systému. Je Určený vývojárom systému, ktorí vyvíjajú systém presne podľa tohto návrhu. Dokument má zaznamenať všetky dôležité rozhodnutia ohľadom výberu technológii i architektúry.

1. Zameranie a rozsah dokumentu

Predpokladom práce s týmto dokumentom je znalosť Katalógu požiadaviek. V tomto dokumente je popísaný podrobný návrh implementácie všetkých bodov z požiadaviek špecifikovaných v katalógu. V dokumente sú uvedené všetky použité technológie. Popisuje tiež rozhrania s externým serverom a komunikáciu s nim. Okrem toho, dokument obsahuje kompletný návrh grafického rozhrania. Na lepšie zobrazenie návrhu systému používa UML diagramy – triedny diagram, sekvenčný, stavový.

1. Používané technológie
2. Java

Programovacím jazykom projektu ju Java vo verzii 11

1. JavaFX – FXML

Na vytvorenie GUI sa využíva knižnica JavaFX, ktorá využíva značkovací jazyk FXML na definovanie používateľského rozhrania. FXML na začiatku vygeneruje jednotlivé *Pane*s, *Button*s, *Label*s a iné objekty GUI aj s ich vzhľadom. Na samotné vyhotovenie FXML súboru je vhodné použiť Scene Builder.

Príklad:

<Label layoutY="-25.0" text="Polar Axis:" textFill="WHITE">

    <font><Font size="18.0" /></font></Label>

1. JSON

V projekte sa využíva JavaScript Object Notation – JSON na komunikáciu medzi backendom a frontendom.

1. Gson

Na spracovávanie JSON štruktúr sa využíva knižnica Google Gson.

1. Eap.fits

V projekte treba čítať súbory typu FITS, načo sa využíva knižnica eap.fits. Táto knižnica poskytuje triedu *RandomAccesFitsFile* ktorou sa otvorí FITS súbor tak, že sa použije ako argument daný FITS súbor štandardnou triedou *RandomAccesFile.* Z *RandomAccesFitsFile* sa dá metódou *getHDU(index)* dostať Header Data Unit  -  *FitsHDU*. V prípade, že chceme dostať z FITS súboru obrázok, treba použiť index 0, prípadne v niektorých prípadoch 1. Na zobrazenie sa používa trieda *FitsImageViewer*ktorá rozširuje triedu knižnice swing *JPanel.*

Ukážka kódu:

RandomAccessFitsFile fitsFile = new RandomAccessFitsFile(openedFile);

FitsHDU hdu = fitsFile.getHDU(0);

FitsImageData imageData = (FitsImageData) hdu.getData();

FitsImageViewer image = new FitsImageViewer(imageData);

1. Prehľad nasledujúcich kapitol

V 2. kapitole sú špecifikované vonkajšie interface-y a to najprv z pohľadu backendu a potom z pohľadu frontendu. V 3. kapitole je predstavený podrobný grafický návrh používateľského rozhrania. Táto kapitola obsahuje viacero obrázkov GUI aj s vymenovaním, čo sa bude na jednotlivých obrazovkách zobrazovať. 4. kapitola približuje implementáciu projektu predstavením rozdelenia modulov aj s opisom ich činností a zodpovedností. Okrem toho sa v nej nachádza viacero UML diagramov aj s opisom.

1. Špecifikácia vonkajších interface-ov

Vyvíjaná aplikácia, teda frontend, bude komunikovať na localhost-e s ďalším komponentom, backend-om, ku ktorého celej implementácii sme nedostali prístup, ale bol nám poskytnutý útržok jeho kódu vo forme [emulátoru](https://drive.google.com/file/d/1DM-ZXVIQVmLwT5dfjfiFxOtVd_Y1r-WU/view?usp=drive_web). Podľa poskytnutého kódu bol vytvorený opis správania backendu a z neho odvodený aj návrh vonkajšieho interface-u frontend-u.

1. Backend

Vytvorí prúdový socket protokolu TCP/IPv4 počúvajúci na lokálnom porte 27017. Po nadviazaní prvého spojenia zavrie socket, ktorý počúval a ďalej komunikuje už iba so socket-om, ktorý bol pridelený prvému — práve-nadviazanému — spojeniu.

Spustí nový thread, ktorý v pravidelnom intervale odosiela aktualizovaný stav ďalekohľadu vo formáte JSON, každý element v JSON je naformátovaný presne ako má byť vykreslený frontend-om.

V hlavnom thread-e nasleduje slučka, ktorá prijíma a spracúva dáta. Po spracovaní prijatých dát vždy odosiela správu, ktorá je v prípade úspechu väčšinou prázdna, ale môže občas obsahovať oznam pre používateľa, a v prípade chyby obsahuje chybovú hlášku. Očakáva dáta, ktoré sú v tvare bežného ANSI reťazca "%command\_id%;%arg1%;%arg2%...", presný zoznam príkazov je vo vyššie spomenutom archíve a prehľad v tejto [tabuľke](https://drive.google.com/file/d/1RU5Zga4QIeIOJ-t_tB5ou7XrWKMb8kvC/view).

Je potrebné podotknúť, že akákoľvek chyba pri komunikácii vedie k ukončeniu spojenia a nie je nijak špeciálne ošetrená.

1. Frontend

Vytvorí prúdový socket protokolu TCP/IPv4 pripájajúci sa na lokálny port 27017. Cieľová adresa a port môžu byť upravené v konfiguračnom súbore config.json. Konfiguračný súbor je formátu JSON a obsahuje elementy address a port.

Po nadviazaní spojenia bude vytvorený nový thread, ktorý prijíma a spracúva dáta z periodických update-ov a správ obsahujúcich výsledky odosielaných príkazov v backend-e. V prípade, že prijaté dáta:

* obsahujú prázdnu správu, nič sa neudeje.
* začínajú znakom ‘{‘, predpokladá sa, že je odosielaný JSON s aktualizovaným stavom ďalekohľadu, bude v takom prípade prekonvertovaný na objekt JSONObject a následne posunutý všetkým modulom, ktoré boli úspešne zaregistrované a tie následne dáta spracujú podľa svojich potrieb.
* začínajú iným znakom ako ‘{‘, predpokladá sa, že správa je výsledok nejakého odoslaného príkazu a zobrazí sa v komponente Operation.

Eventy z používateľského rozhrania frontend-u sú spracúvané príslušnými modulmi, ktoré pomocou komponentu Communication môžu odoslať ľubovoľné dáta, mali by sa pri tom samostatne riadiť zoznamom akceptovaných príkazov, ktorý je vo vyššie spomenutom archíve. Jednotlivé moduly už nemajú možnosť reagovať na spätnú správu backend-u, ktorej spracovanie je popísané vyššie.

1. Návrh používateľského rozhrania
2. Hlavné okno



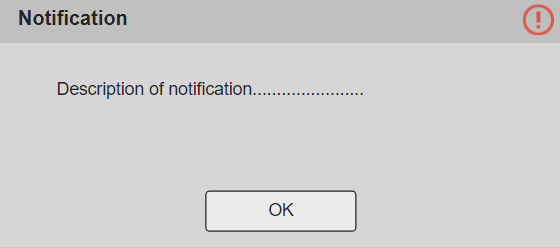
*Obrázok 1: Hlavné okno*

Systém zobrazí v hlavnom okne:

* Názov aplikácie
* Konkrétny čas
* Azimutálne grafické zobrazenie smerovania ďalekohľadu a štrbiny kupoly
* Grafické zobrazenie sklonu ďalekohľadu nad horizontom
* Tlačidlo “Shortcuts” na prechod do okna Shortcuts list (viď.nižšie)
* Informáciu o pripojení aplikácie k backendu
* Poslednú vytvorenú snímku
* Cestu k posledne vytvorenej snímke
* Tlačidlo “EXIT” na ukončenie aplikácie
* Blok Axis bude rozdelený na 2 časti: Polar a Declination Axis
* Blok Polar axis, kde zobrazí hodnoty:
* Encoder
* HA apparent
* RA J2000
* Azimut
* Status
* Blok Declination axis, kde zobrazí hodnoty:
* Encoder
* DE apparent
* DE J2000
* Elevation
* Status
* Podblok pre Polar axis a Declination axis s tlačidlami a vstupnými poľami:
* Enable motors/ Disable motors
* Stop DE, Stop RA
* Stop DE&RA
* Correction
* Calibrate
* Calibrate to zenith
* Slew RA, Slew DE
* Tlačidlo “→” na odosielanie zadaných údajov
* Blok Dome, kde zobrazí hodnoty:
* Encoder
* Azimuth
* Target azimuth
* Synchronize
* Status
* Podblok pre Dome s tlačidlami a vstupnými poľami:
* Go West
* Go East
* Stop move
* Synchronize
* Speed, Azimuth
* Blok FilterWheel, kde zobrazí hodnoty:
* Filter
* Podblok pre Filterwheel s tlačidlami a vstupnými poľami:
* Filter -> B,V,R,I,C
* Blok Operation, kde zobrazí posledne vykonanú operáciu
* Blok Camera, kde zobrazí hodnoty:
* Type
* Exposure
* Mode
* RBI flush count
* RBI flood time
* TDI mode
* Background flush
* Binning
* Subframe
* Observer
* Object
* Notes
* Setpoint
* Cooler
* Delay
* Remaining
* Status
* Podblok pre Camera s tlačidlami a vstupnými poľami:
* Abort imaging, Expose sequence, Expose from nearest.
* Type, Mode
* Exposure time, Cooler setpoint, Sequence repeats, Exposure delay
* Object, Observer, Notes
* Tlačidlo “→” na odosielanie zadaných údajov
* Blok Target, kde zobrazí hodnoty:
* Encoder
* DEnc
* HA apparent, DE apparent
* RA J2000, DE J2000
* Azimuth
* Elevation
* Pole crossing
* Status
* Podblok pre Target s tlačidlami a vstupnými poľami:
* GoTo/Cancel, Pole crossing
* Load Target RA, Load Target DE
* Tlačidlo “→” na odosielanie zadaných údajov
* Blok Schedulera, kde zobrazí a umožní:
* Progress bar vykonávania externého skriptu (v percentách)
* Start scheduling, Load scheduling

1. Okná notifikácii

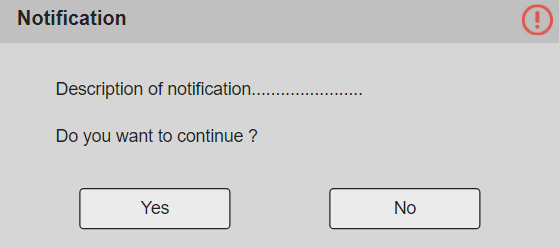
Systém bude zobrazovať notifikácie prostredníctvom dvoch rozdielnych okien (viď. nižšie)



*Obrázok 2: Notifikácia č.1*

V prvom okne notifikácie systém zobrazí:

* Názov notifikácia (“Notification”)
* Popis notifikácie, resp. prečo vznikla
* Tlačidlo “OK”, ktorým používateľ zatvorí notifikáciu

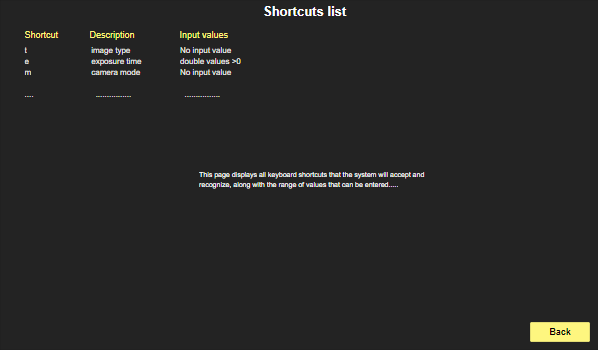


*Obrázok 3: Notifikácia č.2*

V druhom okne notifikácie systém zobrazí:

* Názov notifikácia (“Notification”)
* Popis notifikácie, resp. prečo vznikla
* Otázku, či chce používateľ pokračovať ďalej a akceptovať vzniknutú notifikáciu
* Tlačilo “Yes”, ktorým používateľ akceptuje problém a zatvorí notifikáciu
* Tlačidlo “No”, ktorým používateľ neakceptuje problém a posledná zmena sa neuloží a následne zatvorí notifikáciu

1. Okno klávesových skratiek

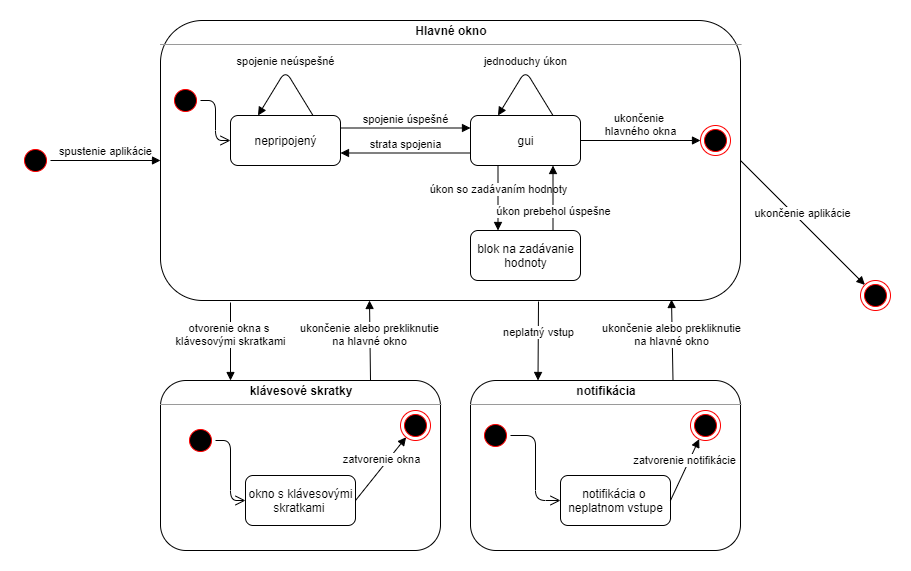


*Obrázok 4: Okno klávesových skratiek*

Systém zobrazí v okne Shortcuts list:

* Zoznam všetkých klávesových skratiek, ktoré systém pozná a akceptuje spolu s intervalom vstupných hodnôt v prípade, že sa vstupná hodnota očakáva
* Tlačidlo “Back”, ktorým používateľ zatvorí okno a vráti sa späť na Hlavné okno

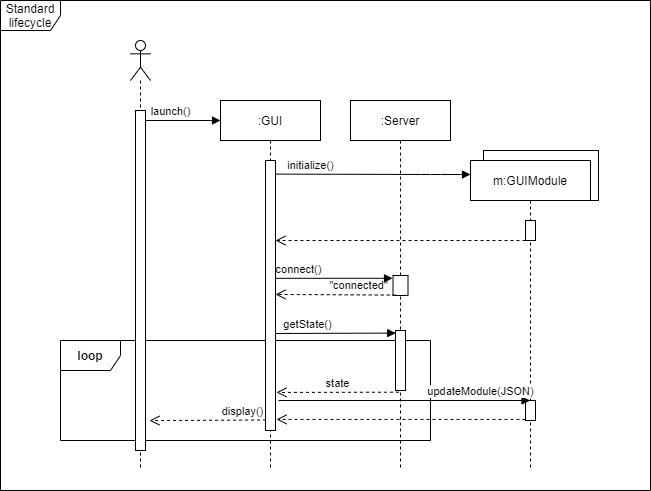
1. Návrh implementácie
2. Stavový diagram



*Obrázok 5: Stavový diagram aplikácie*

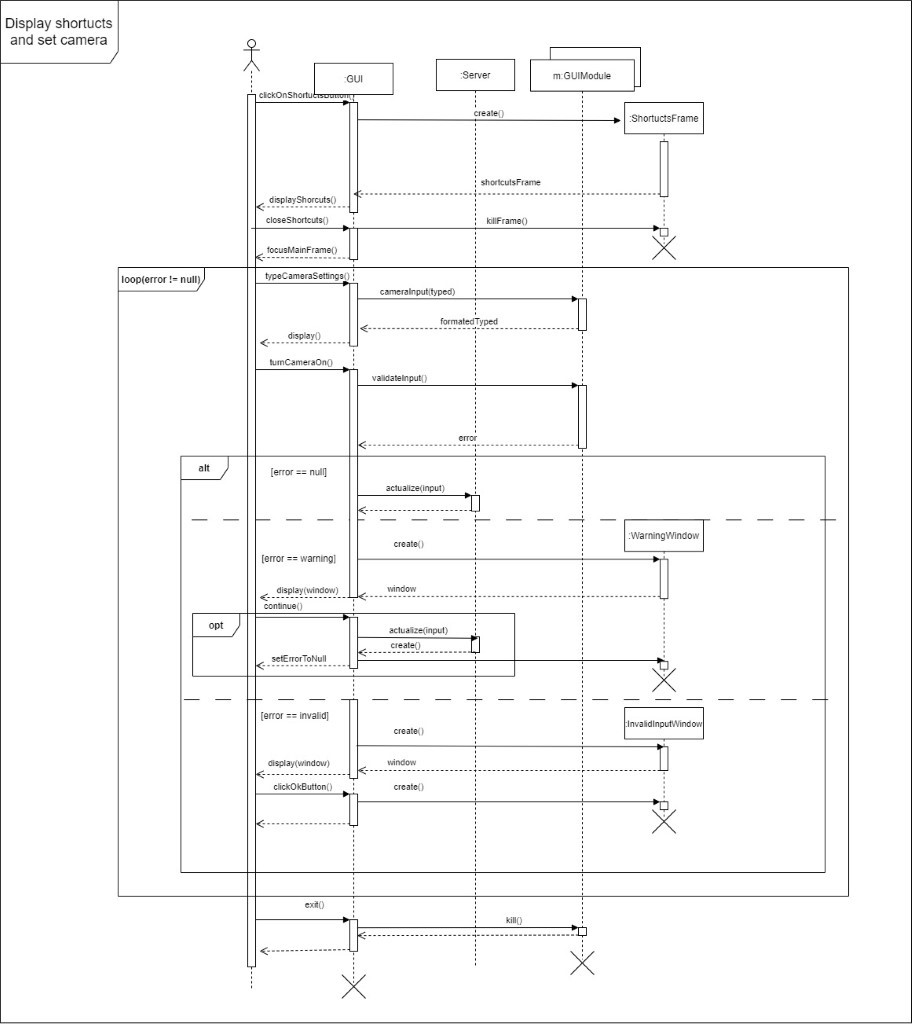
1. Sekvenčné diagramy

Prvý sekvenčný diagram ukazuje čo prebieha pri zapnutí aplikácie a následne po celý čas, kým má aplikácia spojenie so serverom. GUI po celý čas, bez ohľadu na akciu užívateľa prijíma JSON štruktúru, ktorá vyjadruje stav ďalekohľadu a následne tento JSON posúva jednotlivým GUI modulom.



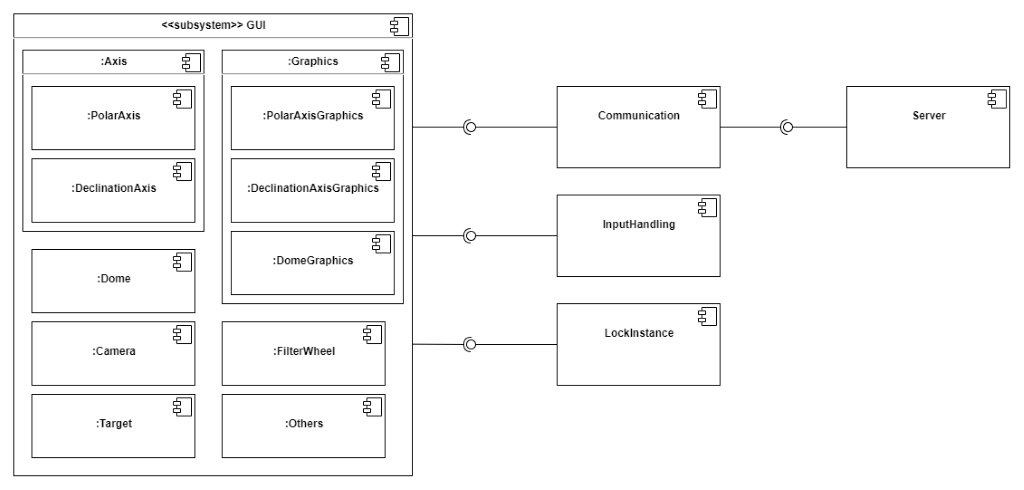
*Obrázok 6: Sekvenčný diagram zapnutia programu a štandardného pasívneho behu*

  Ďalší diagram je už obsiahlejší. Zachytáva pokus o nastavenie kamery s pomocou klávesových skratiek a odoslanie nastavení na server. Mimo cyklu je vyjadrené otvorenie helpera na skratky (Pozri Obrázok 4: Okno klávesových skratiek). Vo vnútri cyklu je samotné zadávanie nastavení a odoslanie na server, prípadne otvorenie varovných alebo chybových okien, v závislosti od vstupu. Koniec nastáva po úspešnom zadaní nastavení kamery.



*Obrázok 7: Sekvenčný diagram, nastavenie kamery*

1. Komponentný diagram



*Obrázok 8: Komponentný diagram*

1. Rozdelenie komponentov

Systém je členení do štyroch javovských balíkov: Communication, InputHandling, LockInstance a GUIModules.

1. Communication

Tento balík obsahuje obsahuje jedinú rovnomennú triedu, ktorá zodpovedá za komunikáciu aplikácie so serverom. Táto trieda musí byť inicializovaná metódou init(host, modules), v ktorom sa začne počúvať host a pridelia sa moduly, ktoré vystupujú ako pozorovatelia. Je zodpovedná za to, aby každá časť systému mala aktuálne dáta zo serveru. O to sa stará jej metóda periodicUpadete(), ktorá počúva server a po prijatí posiela JSON všetkým prihláseným pozorovateľom. Okrem toho, statickou metódou sendData() zabezpečuje, že sa na server odošlú dáta zadané jednotlivým častiam GUI. Túto metódu môže zavolať ktorýkoľvek modul a tak odošle potrebné dáta na server.

1. InputHandling

Tento balík spracováva vstup od používateľa. Zabezpečuje priradenie a spustenie správnej funkcie po zadaní klávesovej skratky. Nestará sa o odchytávanie klávesnice ani o implementáciu reakcie na vstup užívateľa, iba páruje klávesový vstup, ktorý môže byť v hociktorej časti GUI (s výnimkou oblastí na vpisovanie textu a výber možností) so správnou funkciou ktorú poskytuje niektorá trieda implementujúca GUIModel.

Okrem toho, súčasťou tohto balíka je trieda zabezpečujúca zobrazenie upozornení na nezvyčajný, prípadne chybný vstup.

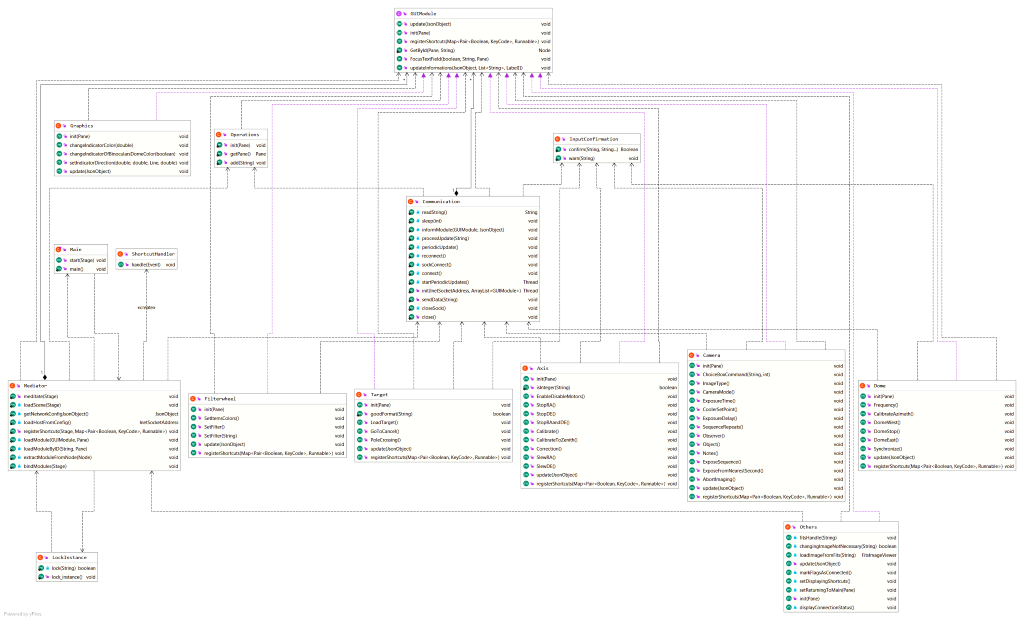
1. LockInstace

Balík LockInstance zabezpečuje, aby nebolo možné spustiť viacero inštancií aplikácie naraz pomocou jedinej triedy LockInstance. Tá poskytuje statickú metódu lock\_instance(), ktorú treba zavolať na začiatku programu a ktorá program ukončí, ak už beží iná inštancia aplikácie.

1. GUIModules

Najobsiahlejší balík GUIModels  je najvyššou vrstvou aplikácie, stará sa o zobrazenie všetkých potrebných údajov používateľovi. Obsahuje rozhranie GUIModel a triedy, ktoré toto rozhranie implementujú. Jednotlivé triedy predstavujú logické celky v GUI, napr. trieda Dome zabezpečuje vstup a zobrazenie informácii týkajúcich sa kupoly, pričom tieto informácie sú členené pohromade, ako ukazuje  Obrázok 1. Všetky triedy tohto balíka implementujú interface GUIModule. Ten predpisuje metódy update(JSON), init(Pane) a registerShortcuts(shortcut). V metóde init každá trieda inicializuje svoj pane. V update aktualizuje údaje na základe prijatého JSON-u.  Tieto triedy tiež vytvárajú funkcie zodpovedajúce jednotlivým skratkám, ktoré registrujú funkciou registerShortcuts().

1. Triedny diagram



Testovacie scenáre

1. Spustenie aplikácie

**Scenár:**Užívateľ spustí aplikáciu.

**Očakávaný výstup:**Aplikácia sa pokúsi pripojiť na server, ak sa jej to podarí, začne prijímať a zobrazovať dáta predstavujúce aktuálne nastavenia ďalekohľadu. Ak sa na server nedokáže pripojiť, opätovne sa ďalej pokúša nadviazať spojenie pokiaľ ju užívateľ nevypne. Stav spojenia aplikácie je nastavený na disconnected, ako v prípade straty pripojenia k backendu, kým sa nepripojí.

**Súvisiace požiadavky:**3.1.1, 3.1.2

1. Ukončenie aplikácie

**Scenár:**Užívateľ ukončí aplikáciu stlačením x

**Očakávaný výstup:**Aplikácia sa odpojí zo servera a štandardne sa ukončí. Nič sa neukladá.

**Súvisiace požiadavky:**3.1.12

1. Používanie klávesových skratiek

**Scenár:** Užívateľ klikne na tlačidlo skratky, následne okno skratiek vypne.

**Očakávaný výstup:**Po kliknutí na tlačidlo skratiek sa otvorí nové okno na ktorom budú zobrazené všetky klávesové skratky použiteľné v aplikácii aj s krátkym vysvetlením, čo jednotlivé skratky robia. Po zatvorení tohto okna sa focus vráti na hlavnú stránku GUI.

**Súvisiace požiadavky:**3.1.4.1, 3.1.6

**Upozornenie: všetky nasledujúce scenáre, v ktorých sa nastavujú parametre komponentov možno vykonať nielen pomocou myši, ale aj pomocou týchto klávesových skratiek.**

1. Strata spojenia s komponentom

**Scenár:**Pri pasívnom prezeraní GUI niektorá časť ďalekohľadu prestane odpovedať.

**Očakávaný výstup:**Aplikácia zmení zobrazovaný status komponentu z „ready“ na „not responding“. Okrem toho používateľa upozorní vyskakovacím oknom na stratu spojenia.

**Súvisiace požiadavky:**3.1.8

1. Strata spojenia s backendom

**Scenár:**Pri pasívnom prezeraní GUI backend prestane odpovedať.

**Očakávaný výstup:**Stav pripojenia sa zmení z connected na disconnected.

**Súvisiace požiadavky:**3.1.9

1. Nastavenie a zapnutie kamery

**Scenár:**Užívateľ správne nastaví postupne typ snímky, čas expozície, mód kamery, želanú teplotu kamery, omeškanie medzi snímkami, počet ostávajúcich snímkov, meno pozorovateľa, meno pozorovaného objektu a poznámky. Po nastavení zapne kameru.

**Očakávaný výstup:**Aplikácia postupne odosiela nastavené parametre na server. Preto začne v zobrazovacej časti zobrazovať nové parametre, ktoré boli práve nastavené. Po kliknutí na tlačidlo „Turn camera ON“ sa pošle na backend príkaz na zapnutie kamery a začne sa meniť zobrazovaný last frame a zobrazovaná cesta k nemu.

**Súvisiace požiadavky:**3.2.1, 3.2.2, 3.2.3, 3.2.4, 3.2.5, 3.2.6, 3.2.7, 3.2.8, 3.2.9, 3.2.10, 3.2.11

1. Chybné nastavenie času expozície kamery

**Scenár:**Používateľ zadá exposure v chybnom formáte a takto sa ho pokúsi odoslať.

**Očakávaný výstup:**Aplikácia zobrazí výstražné okno s upozornením na chybne zadaný formát.

**Súvisiace požiadavky:**3.2.3, 3.1.8

1. Nastavenie filtra

**Scenár:**Používateľ zmení filter.

**Očakávaný výstup:**Systém odošle zmenu na server, ten zmenu zaregistruje a posiela naspäť frontendu. Potom sa zmena zobrazí aj v zobrazovacej časti. Kým sa mení filter – čo nejaký čas trvá – v bloku Operation sa o tom vypisuje správa. Keď sa výmena dokončí, vypíše sa o tom v tomto bloku správa.

**Súvisiace požiadavky:**3.3.1, 3.3.2, 3.3.3.

1. Nastavenie bloku target

**Scenár:**Používateľ nastaví target RA a DE, zapne pole crossing a zapne funkciu Go To.

**Očakávaný výstup:**Predpokladáme, že pole crossing bol vypnutý. Systém odošle zadaný target RA a DE, po kliknutí na tlačidlo pole crossing odošlo hodnotu „on“. Tieto zmeny začne zobrazovať. Zapnutie funkcie Go To ovplyvní pohyb ďalekohľadu a teda obe grafiky a zobrazované údaje v bloku Axis.

**Súvisiace požiadavky:**3.4.1, 3.4.2, 3.4.2, 3.4.3, 3.4.4, 3.5.1, 3.5.2, 3.6.1, 3.6.2

1. Nastavenie smeru ďalekohľadu

**Scenár:**Užívateľ kliknutím zapne motory a pomocou GoTo nastaví RA aj DE. Po dokončení pohybu spustí kalibráciu a nakoniec vypne motory.

**Očakávaný výstup:**Tlači „Enable“ sa kliknutím zmení na „Disable“ a odoslaním príkazu na backend sa sa zapnú motory. Po zadaní RA a DE a spoločnom odoslaní sa tieto údaje pošlú na backend a ďalekohľad sa začne hýbať. To sa zobrazí okrem iného aj na grafike. Kliknutie na tlačidlo „Calibrate“ spôsobí odoslanie požiadavky na kalibráciu na backend. Kliknutie na „Disable“ vypne motory a zmení tlačidlo späť na „Enable“.

**Súvisiace požiadavky:**3.5.1, 3.5.2, 3.5.3, 3.6.1, 3.6.2, 3.6.3

1. Nastavenie RA a neskôr DE ďalekohľadu s automatickou korekciou

**Scenár:** Užívateľ zapne motory. Potom zadá Slew RA, pred dokončením pohybu ale stlačí „Stop RA“. Potom zadá a odošle DE, no ešte pred dokončením pohybu stlačí „Stop DE“. Nakoniec stlačí tlačidlo „Correction“.

**Očakávaný výstup:**Pri stlačení „Enable“ sa text tlačidla zmení na „Disable“ a na backend sa odošle príkaz ktorým sa zapnú motory. Odoslaním Slew RA sa začne hýbať, no iba v RA smere. Stlačením „Stop RA“ pohyb prestane. Pri odoslaní „Slew DE“ sa začne ďalekohľad opäť hýbať, tentokrát v DE smere. Podobne „Stop DE“ tento pohyb zastaví. Nakoniec sa stlačením „Correction“ odošle príkaz na automatickú korekciu.

**Súvisiace požiadavky:**3.5.1, 3.5.2, 3.5.3, 3.6.1, 3.6.2, 3.6.3

1. Nastavenie želaného smeru kupoly

**Scenár:**Používateľ zadá želaný azimut a odošle ho.

**Očakávaný výstup:**Aplikácia odošle želaný azimut kupoly na backend, následkom čoho sa kupola začne hýbať v požadovanom smere, čo sa zobrazí v grafike aj texte v bloku Dome. Okolo grafiky sa bude zobrazovať červený rámček, signalizujúci vypnutú synchronizáciu kupoly a ďalekohľadu.

**Súvisiace požiadavky:**3.7.1, 3.7.2, 3.7.3

1. Pohyb kupoly a následná synchronizácia kupoly

**Scenár:**Užívateľ najprv nastaví rýchlosť pohybu, potom klikne na „Go West“, po chvíli na „Stop move“, „Go East“ a po chvíli opäť na „Stop move“. Nakoniec klikne na „Synchronize“.

**Očakávaný výstup:**Pri nastavení rýchlosti sa odošle na backend požadovaná rýchlosť. Pri stlačení Go west sa začne kupola hýbať, čo bude vidno aj na grafike. Pri stlačení „Stop move“ sa zastaví a pri stlačení „Go East“ sa začne hýbať smerom na východ a po opätovnom stlačení “Stop move” sa zastaví. Pri stlačení „Synchronize“ sa začne hýbať smerom tak, aby bola synchronizovaná s ďalekohľadom. To, že je kupola synchronizovaná sa prejaví v grafike tým, že sa už nebude zobrazovať červený štvorec okolo grafiky kupoly.

**Súvisiace požiadavky:**3.7.1, .3.7.2, 3.7.3