
Käyttöohje

Skripti Ry Telegram Bot ja Tenttiarkisto

SISÄLLYSLUETTELO

1 JÄRJESTELMÄN KÄYTTÖÖNOTTO	4
1.1 TELEGRAM BOTIN KÄYTTÖÖNOTTO	4
1.1.1 Käyttöympäristö	4
1.1.2 Sovelluksen vaatimukset	4
1.1.3 Palvelimelle asennus	4
1.2 FLASK API:N KÄYTTÖÖNOTTO	5
1.2.1 Käyttöympäristö	5
1.2.2 Sovelluksen vaatimukset	5
1.2.3 Palvelimelle asennus	5
2 TUOTTEEN KÄYTTÖ	6
2.1 TELEGRAM BOT	6
2.1.1 Esivaatimukset	6
2.1.2 Käytön aloittaminen	6
2.1.3 Botin komennot	7
2.2 TENTTIARKISTO FLASK API	13
2.2.1 Esivaatimukset	13
2.2.2 Resurssirakenne	13
2.2.3 Uusien tiedostojen tallennus	14
2.2.4 Palvelimen vastauksen rakenne	15
3 ONGELMATILANTEET	17
3.1 VIRHEILMOITUKSET	17
3.1.1 Järjestelmän virheilmoitukset	17
3.1.2 Käyttäjälle näkyvät virheilmoitukset	18
3.1.3 Virheistä palautuminen	18
4 YLLÄPITO	19
4.1 BOTIN YLLÄPITO JA PÄIVITTÄMINEN	19
4.1.1 Ylläpito	19
4.1.2 Päivittäminen	19
4.1.3 Virheiden raportointi	19
4.2 FLASK API:N YLLÄPITO JA PÄIVITTÄMINEN	20
4.2.1 Ylläpito	20
4.2.2 Päivittäminen	20
4.2.3 Virheiden raportointi	20

5 LISÄTIETOJA.....	21
5.1 YHTEYSTIEDOT.....	21
5.2 KIRJASTOJEN DOKUMENTAATIO.....	21

1 JÄRJESTELMÄN KÄYTTÖÖNOTTO

- *Yksityiskohtaiset, vaiheittaiset ohjeet, kuinka järjestelmän asennetaan*
- *Vaadittu laitteistoympäristö ja muut ohjelmat/sovellukset asennuksessa*

1.1 Telegram botin käyttöönotto

1.1.1 Käyttöympäristö

Telegram bot on kehitetty ja testattu Linux ympäristössä. Botti kääntyy myös Windows ympäristössä, mutta sen toimintaa ei ole testattu yhtä laajasti.

1.1.2 Sovelluksen vaatimukset

Sovellus on kehitetty Python 3 kielellä ja vaatii toimivan Python 3 asennuksen toimiakseen. Botti käyttää seuraavia kirjastoja, jotka pitää asentaa ennen sovelluksen käynnistämistä:

- python-telegram-bot
- python-dotenv
- requests
- urllib3

1.1.3 Palvelimelle asennus

Botti on asennettu valmiiksi Skripti Ry:n käyttöä varten, mutta jos haluat luoda oman bottisi, joka käyttää samaa sovelluslogiikkaa niin seuraa näitä ohjeita:

1. Kloonaa botin sovelluskoodin sisältävä repositorio palvelimelle.
2. Luo Telegramin BotFatherilla itsellesi botti ja ota sen Token ylös.
3. Kysy Skripti Ry:ltä tenttien tallennustunnukset, jos haluat käyttää tenttien tallennusominaisuutta.
4. Vie äskeiset tiedot .env tiedostoon samaan kansioon botin kanssa, muodossa:
token = *botti-token*
user = *tallennuskäyttäjätunnus*
password = *tallennussalasana*
5. Käynnistä botti update.sh shell-skriptillä. Skripti saattaa kysyä sinulta kloonavan tunnuksen salasanaa, riippuen siitä mistä yrität kloonata repositoriota.

1.2 Flask API:n käyttöönotto

1.2.1 Käyttöympäristö

Tenttiarkiston Flask API on kehitetty ja testattu Linux ympäristössä, mutta se saattaa toimia myös Windowsilla. Kaikki testaus on suoritettu Linux-isännöidyllä versiolla.

1.2.2 Sovelluksen vaatimukset

API on kirjoitettu Python 3 ohjelmointikielellä Flask kirjastoa käyttäen. Botti käyttää seuraavia kirjastoja, jotka pitää asentaa ennen sovelluksen käynnistämistä:

- Flask
- Flask-HTTPAuth
- GitPython
- python-dotenv

1.2.3 Palvelimelle asennus

API on asennettu valmiiksi Skripti Ry:n käyttöä varten osoitteeseen api.skripti.org, mutta jos haluat luoda oman kopiosi proxy-palvelimesta, joka käyttää samaa sovelluslogiikkaa niin seuraa näitä ohjeita:

1. Luo itsellesi GitHub repositorio
2. Luo/Hae GitHub tunnuksesi OAuth autentikaatio-Token repositorioon.
3. Käännä cloneGit.py tiedostossa oleva osoite-muuttuja luomaasi arkistoon.

Esim. `'https://' + token + ':x-oauth-basic@github.com/käyttäjä/repositorio'`

4. Luo .env tiedosto API:n juurikansioon, joka sisältää seuraavat tiedot:

```
cloneToken = *OAuth token*
actorEmail = *Määritelty lataajan SäPo*
actorName = *Lataajan nimi*
apiUser = *Tallennustunnuksen käyttäjätunnus (vapaavalinta)*
apiPassword = *Tunnuksen vapaavalintainen salasana*
```

5. Lataa jokin tuotannon WSGI palvelin, joka pystyy suorittamaan Flask-sovelluksia. Flaskin sisäänrakennetun palvelimen käyttäminen tuotantoympäristössä ei ole suositeltavaa.

2 TUOTTEEN KÄYTTÖ

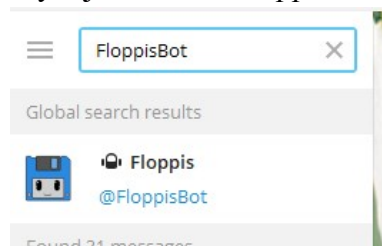
2.1 Telegram bot

2.1.1 Esivaatimukset

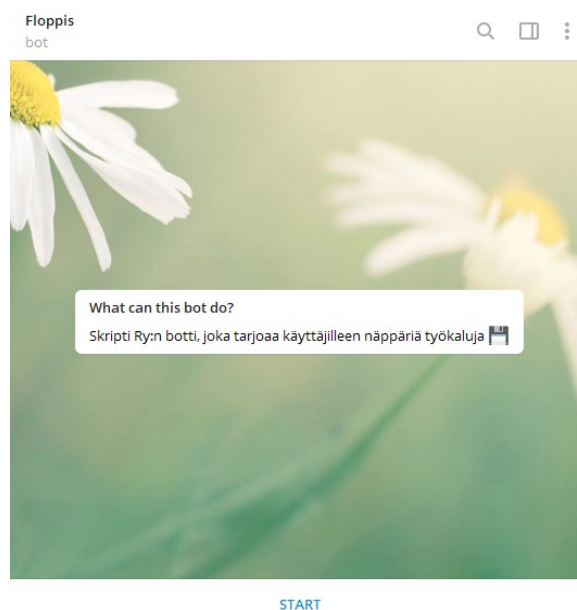
Vaativuksena Telegram Botin käytölle on asennettu Telegram sovellus. Telegram sovelluksen saa asennettua mobiililaitteille ja pöytäkoneille.

2.1.2 Käytön aloittaminen

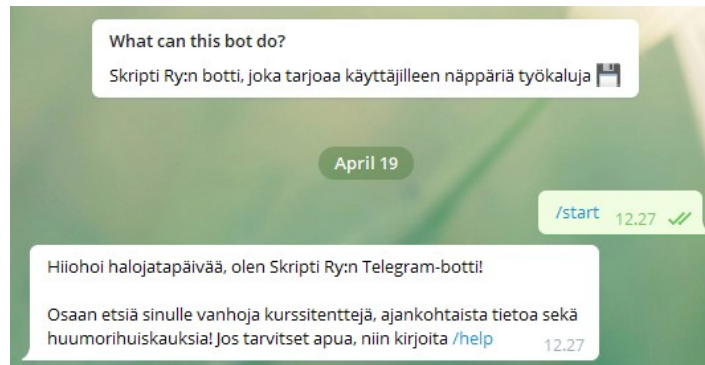
1. Avaa telegram sovellus, valitse käyttäjätunnusten haku ja etsi käyttäjää nimeltä *FloppisBot*.



2. Valitse käyttäjä. Ruudulle aukeaa botista lyhyt kuvaus.

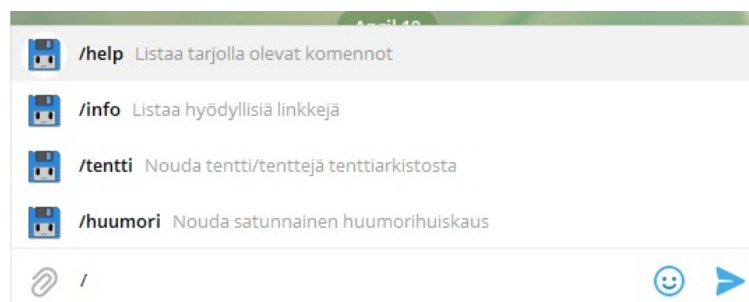


3. Paina *START* näppäintä. Botti vastaa lyhyellä esittelyllä, minkä jälkeen botti on käyttövalmiina.



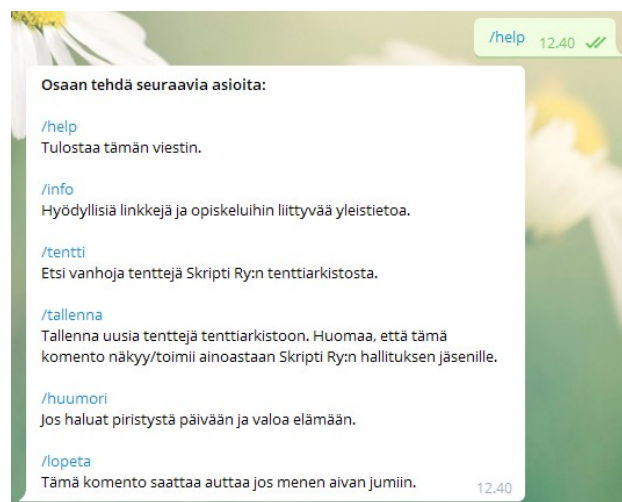
2.1.3 Botin komennot

Bottia käytetään komennoilla, joista tärkeimpiä botti itse ehdottaa käyttäjälle. Nämä ehdotukset näkyvät sen jälkeen kun käyttäjä on kirjoittanut "/" tai painanut Telegramin kenoviiva-nappia.



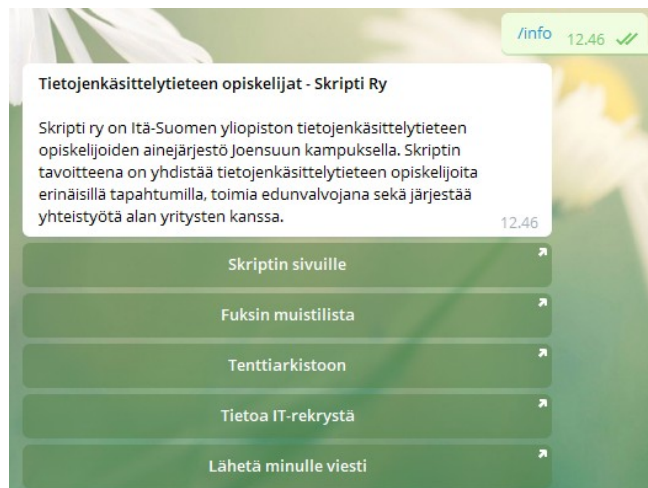
/help

Lähetää käyttäjälle viestin, jossa listataan botin julkiset komennot sekä niiden lyhyet kuvaukset.



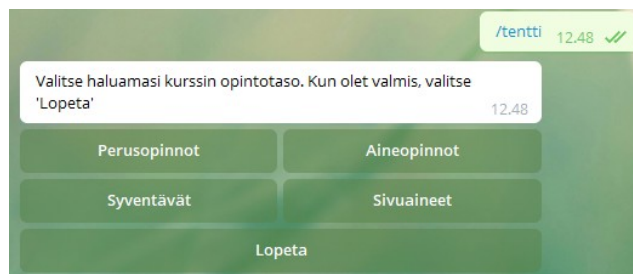
/info

Lähetää käyttäjälle viestin, joka sisältää lyhyen tietoisuuden Skripti Ry:stä sekä listaa alle hyödyllisiä linkkejä.



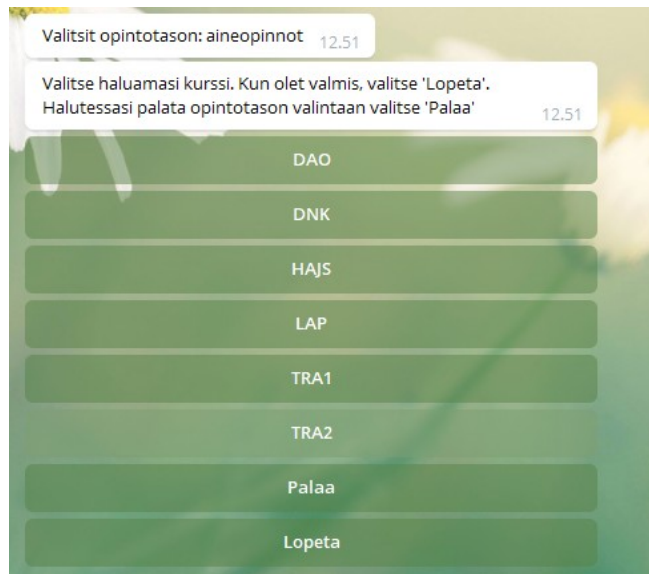
/tentti

Tarjoaa käyttäjälle yksinkertaisen käyttöliittymän, jonka kautta pystyy selaamaan ja hakemaan vanhoja tenttejä arkistosta. /tentti komennon jälkeen aukeaa opintotason valinta.



Käyttäjä voi lopettaa tenttien haun ja sulkea käyttöliittymän missä tahansa vaiheessa painamalla "Lopeta" nappia.

Käyttäjä valitsee haluamansa opintotason, jolloin sovellus listaa kaikki opintotason kurssit, jotka löytyvät arkistosta.

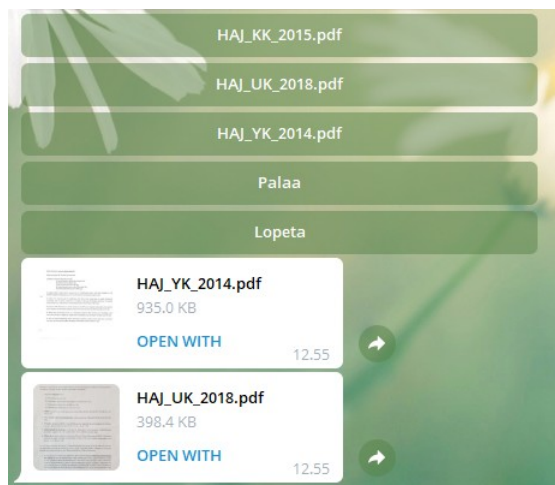


Käyttäjä voi palata takaisin hakemistorakenteessa missä tahansa vaiheessa painamalla "Palaa" nappia.

Käyttäjä valitsee haluamansa kurssin, jolloin sovellus listaa kaikki kyseisen kurssin tentit.



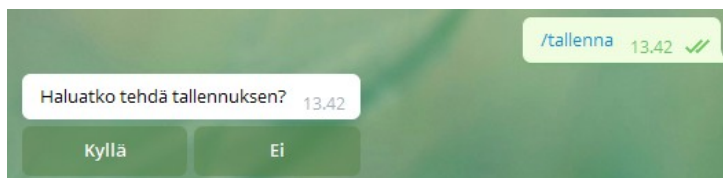
Tämän jälkeen käyttäjä voi ladata tenttejä. Nappia painaessa botti lähettää käyttäjälle tiedoston viestillä.



Jos käyttöliittymää ei katkaista “Lopeta” näppäimellä ja sitä ei käytetä viiteen minuuttiin, niin sovellus aikakatkaisee keskustelun.

/tallenna

Tämä komento toimii ja näkyy ainoastaan Skripti Ry:n hallituksen jäsenille, eli niille jotka ovat Skripti – Yleinen Telegram keskusteluryhmän Admin rooleissa.



Alussa botti varmistaa, että käyttäjä todella haluaa tehdä tallennuksen. Jos käyttäjällä ei ole tallennusoikeuksia niin botti ilmoittaa siitä.



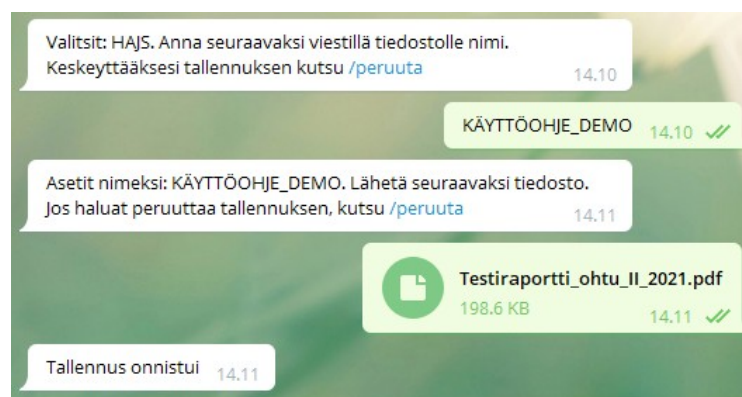
Tämän jälkeen botti kysyy tallennettavan tentin opintotasoa. Tallennuksen voi keskeyttää painamalla “Keskeytä” nappia.

Opintotason valinnan jälkeen botti näyttää käyttäjälle kaikki opintotason alle lisätyt kurssit.



Tässä vaiheessa käyttäjä voi valita jo olemassaolevista kursseista, lisätä uuden kurssin, palata opintotason valintaan tai keskeyttää tallennuksen kokonaan.

Jos käyttäjä päättää lisätä uuden kurssin, niin botti kysyy uuden kurssin nimen ennen tiedoston nimeämistä.



Kun käyttäjä on valinnut kurssin, niin botti kysyy käyttäjältä tallennettavan tiedoston nimen. Nimen määrittämisen jälkeen botti pyytää itse tiedostoa. Kun käyttäjä on lähettänyt tiedoston keskusteluun, botti ilmoittaa onnistuiko tallennus.

Tallennuksen voi peruuttaa tässäkin vaiheessa kutsumalla komentoa /peruuta

/huumori

Lähetää käyttäjälle satunnaisesti arvotun humoristisen kuvan.



2.2 Tenttiarkisto Flask API

Ohjeen esimerkit käyttävät Skripti Ry:n isännöimää API versiota osoitteessa api.skripti.org.

2.2.1 Esivaatimukset

Tenttiarkiston API:n käytölle ei ole esivaatimuksia, muuta kuin toimiva verkkoyhteys ja selain/työkalu jolla voi tehdä http pyyntöjä.

2.2.2 Resurssirakenne

API tarjoilee kurssimateriaalia eri kokonaisuuksina. Haettavan materiaalin laajuutta rajataan seuraavan URL rakenteen avulla:

api.skripti.org/

- Indeksi, listaa kaiken palvelimella olevan materiaalin.
- JSON rakenne: Opintotaso – Kurssi – Tiedostot

api.skripti.org/perusopinnot/

- Listaa kaikki palvelimella olevat perusopintotason tentit.
- Muita opintotasoja ovat aineopinnot, syventavat_opinnot ja sivuaineet.
- JSON rakenne: Kurssi - Tiedostot

api.skripti.org/perusopinnot/DNK/

- Listaa kaikki tietyn kurssin tentit.
- Opintotason tarjolla olevat kurssit löytyvät osoitteesta [url/opintotaso/](https://api.skripti.org/perusopinnot/DNK/)
- JSON rakenne: Tiedostot

api.skripti.org/perusopinnot/DNK/DNK_KK_2021.pdf

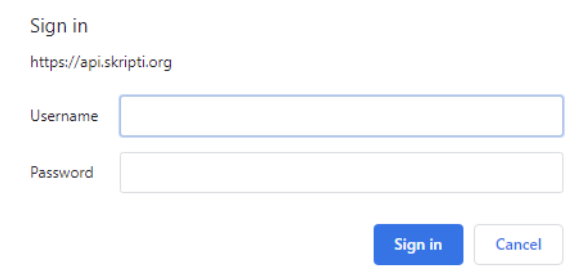
- Tarjoilee käyttäjälle haetun tentin, jos se löytyy palvelimelta.
- Tentti renderöityy selaimen tai siirtyy pyynnön mukana pyytäjälle.

2.2.3 Uusien tiedostojen tallennus

Uusien tiedostojen tallennus tapahtuu joko GitHubin repositorion kautta tai osoitteen *api.skripti.org/upload* kautta. Hyväksytyjä tiedostomuotoja ovat pdf, png, jpg ja jpeg.

Tallennus selaimella (huom, tämä tallennustapa ei ole suositeltava ja poistuu mahdollisesti tulevissa versioissa):

Alussa selain kysyy tallentajaa tunnistautumaan



Sign in
https://api.skripti.org

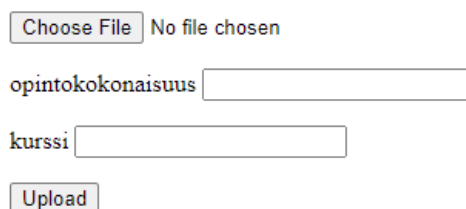
Username

Password

Kirjaudu API:n asennusvaiheessa määrittelemilläsi tallennustunnuksilla.

Tämän jälkeen aukeaa yksinkertainen HTML-formi.

Upload new File



No file chosen

opintokokonaisuus

kurssi

Valitse ladattava tiedosto, kirjoita opintokokonaisuuteen joko perusopinnot, aineopinnot, syventavat opinnot tai sivuaineet. Kirjoita Kurssi kenttään tentin kurssin lyhenne. Jos kurssia ei ole ollut aikaisemmin palvelimella, niin se luo sen.

Tämän jälkeen paina upload, jolloin tentti tallentuu palvelimelle ja synkronoituu GitHub repositorioon.

Tallennus HTTP POST metodilla:

POST metodi käyttää myös ennalta määriteltyä formia. Pyynnössä täytyy siis olla parametrit file, opintokokonaisuus ja kurssi.

file = pdf, png, jpg tai jpeg tiedosto

opintokokonaisuus = perusopinnot, aineopinnot, syventavat_opinnot tai sivuaineet

kurssi = Tekstimuotoinen lyhenne

Tiedoston lataaminen vaatii HTTP autentikoinnin ennen POST pyyntöä.

2.2.4

Palvelimen vastauksen rakenne

Kaikki palvelimen vastaukset tulevat application/JSON muodossa, paitsi itse tentit, jotka tulevat application/pdf muodossa. JSON-vastauksien rakenne on määritelty kappaleessa 2.2.2.

Huomattavat http Header parametrit:

access-control-allow-origin: * - Mikä tahansa koodi mistä tahansa osoitteesta saa pyytää palvelimen resursseja.

X-content-type-options: nosniff – Estää selaimia päättelemästä MIME-tyyppiä ja pakottaa ne käyttämään content-type headerin mukaista tyyppiä. Tämä estää esimerkiksi vihamielisen käyttäjän

```
    "from": {
        "id": 1580463617,
        "first_name": "Floppis",
        "is_bot": true,
        "username": "FloppisBot"
    },
    "data": "sivuaineet/DTP/dtp1_2020_kk.pdf",
    "from": {
        "id": 735587713,
        "first_name": "Armas",
        "is_bot": false,
        "last_name": "Ahlholm",
        "username": "nhola",
        "language_code": "fi"
    }
}

context.chat_data = {}

context.user_data = {}

Traceback (most recent call last):
  File "/home/ohu/.local/lib/python3.7/site-packages/telegram/ext/dispatcher.py", line 442, in process_update
    handler.handle_update(update, self, check, context)
  File "/home/ohu/.local/lib/python3.7/site-packages/telegram/ext/conversationhandler.py", line 549, in handle_update
    new_state = handler.handle_update(update, dispatcher, check_result, context)
  File "/home/ohu/.local/lib/python3.7/site-packages/telegram/ext/handler.py", line 160, in handle_update
    return self.callback(update, context)
  File "main.py", line 282, in button5
    query.message.reply_document(fileurl)
  File "/home/ohu/.local/lib/python3.7/site-packages/telegram/message.py", line 995, in reply_document
    caption_entities=caption_entities,
  File "/home/ohu/.local/lib/python3.7/site-packages/telegram/bot.py", line 117, in decorator
    result = func(*args, **kwargs)
  File "/home/ohu/.local/lib/python3.7/site-packages/telegram/bot.py", line 872, in send_document
    api_kwargs=api_kwargs,
  File "/home/ohu/.local/lib/python3.7/site-packages/telegram/bot.py", line 271, in _message
    result = self._post(endpoint, data, timeout=timeout, api_kwargs=api_kwargs)
  File "/home/ohu/.local/lib/python3.7/site-packages/telegram/bot.py", line 235, in _post
    return self.request.post(f'{self.base_url}/{endpoint}', data=data, timeout=timeout)
  File "/home/ohu/.local/lib/python3.7/site-packages/telegram/utils/request.py", line 354, in post
    **urlopen_kwargs,
  File "/home/ohu/.local/lib/python3.7/site-packages/telegram/utils/request.py", line 272, in _request_wrapper
    raise BadRequest(message)
telegram.error.BadRequest: Wrong file identifier/http url specified
```

11:54

lataaman virheellisen tiedoston tulkitsemisen suoritettavana tiedostona ohi content-type:ssä määritellyn MIME-tyypin.

3 ONGELMATILANTEET

3.1 Virheilmoitukset

3.1.1 Järjestelmän virheilmoitukset

Jos botin Python koodin suorituksessa tapahtuu virhe, niin se lähettää siitä automaattisesti viestin skripti-dev Telegram keskusteluryhmään. Tämä viesti on hyvin laaja, johon sisältyy komennon kutsujan tiedot, telegramin muuttujien tila virheen sattuessa sekä Pythonin Traceback.

```
Floppis Reply
An exception was raised while handling an update
update = {
  "update_id": 7684276,
  "callback_query": {
    "id": "3159325174297728921",
    "chat_instance": "991287169472563824",
    "message": {
      "message_id": 1943,
      "date": 1616493288,
      "chat": {
        "id": 735587713,
        "type": "private",
        "username": "nholm",
        "first_name": "Armas",
        "last_name": "Ahlholm"
      },
      "text": "Valitse haluamasi tiedosto(t). Kun olet valmis valitse 'Lopeta'",
      "entities": [],
      "caption_entities": [],
      "photo": [],
      "new_chat_members": [],
      "new_chat_photo": [],
      "delete_chat_photo": false,
      "group_chat_created": false,
      "supergroup_chat_created": false,
      "channel_chat_created": false,
      "reply_markup": {
        "inline_keyboard": [
          [
            {
              "text": "dtp1_2020_kk.pdf",
              "callback_data": "sivuaineet/DTP/dtp1_2020_kk.pdf"
            }
          ],
          [
            {
              "text": "Lopeta",
              "callback_data": "end"
            }
          ]
        ]
      }
    }
  }
},
```

3.1.2 Käyttäjälle näkyvät virheilmoitukset

Käyttäjälle ei näy muita virheilmoituksia, kuin /tallenna komennossa tentin tallennuksen epäonnistuminen.

Jos käyttäjä kutsuu /help, /tallenna tai /tentti komentoa ryhmäkeskusteluissa, botti ilmoittaa, että komentoa voi kutsua ainoastaan yksityisviestillä.

3.1.3 Virheistä palautuminen

Lähtökohtaisesti botti suorittaa jokaisessa yhteydessä uuden instanssin koodipohjasta, eli jos yhden käyttäjän yhteydessä tapahtuu virhe, niin se ei vaikuta toiseen käyttäjään.

Jos käyttäjällä menee avattu keskustelu jumiin, niin */lopeta* komennolla voi katkaista avatun yhteyden. Tämä saattaa auttaa virhetilanteissa/jumeissa.

Jos koko botti jumittuu/kaatuu, täytyy se uudelleenkäynnistää `update.sh` shell-skriptillä, joka sijaitsee botin juurikansiossa.

Jos `api.skripti.org` ei toimi/kaatuu, niin kokeile ensin kohdassa 4.2.2 esiteltyä komentoa `touch tmp/restart.txt`. Tämä uudelleenkäynnistää sovelluslogiikan.

Jos botin tenttien tallennus ilmoittaa 500 error, mutta tentti ilmestyy palvelimelle, niin käy tarkistamassa GitHubin Oauth token.

4 YLLÄPITO

- *Vaatiiko ohjelmisto joitain säännöllisiä ylläpitotoimenpiteitä?*
- *Miten ohjelmiston voi päivittää?*
- *Miten virheistä voi raportoida?*
- *Miten ohjelmistoon voi tehdä muutoksia tai mukautuksia?*

Tämä osio käsittelee Skripti Ry:n palvelinympäristön ja sovellusten ylläpitoa ja päivittämistä.

4.1 Botin ylläpito ja päivittäminen

4.1.1 Ylläpito

Bottia isännöi Itä-Suomen Yliopiston Tietojenkäsittelytieteen laitoksen omistama palvelin. Botti ei vaadi säännöllisiä ylläpidollisia toimenpiteitä. Palvelimen ongelma/virhetilanteissa apua saa Juha Hakkaraiselta juhakka@cs.uef.fi

4.1.2 Päivittäminen

Päivittäminen tapahtuu asennuskansion juuressa olevalla update.sh shell-skriptillä. Tämä skripti käy hakemassa määritellystä repositoriosta uusimmat päivitykset ja käynnistää botin Python skriptit uudelleen. Shell skripti saattaa kysyä salasanaa riippuen repositorion isännöintikohteesta. Jos repositorio siirretään Azure Devopsista, niin update.sh skripti pitää todennäköisesti ajantasaistaa uudelle repositoriolle.

4.1.3 Virheiden raportointi

Jos sovelluksessa havaitaan virheitä, niin niistä tulee ilmoittaa Skripti Ry:n hallitukselle, joka ohjaa virheet eteenpäin skripti-dev ryhmälle. Kehitysryhmällä ei ole sitoumuksia korjata virheitä, sillä sovelluksen jatkokehitys perustuu vapaaehtoisuuteen.

4.2 Flask API:n ylläpito ja päivittäminen

4.2.1 Ylläpito

Flask API:a isännöi OVH Cloud Skripti Ry:n tilauksella. OVH tarjoaa tällä hetkellä beta-testauksessa olevaa Deployed Python isännöintimallia. Testivaiheen kesto ei ole tarkasti tiedossa, joten OVH:n uutisointia tulee seurata tarkasti. Palvelu muuttuu maksulliseksi testivaiheen jälkeen, jolloin Skripti Ry:n tulee tehdä tilaus isännöinnistä.

4.2.2 Päivittäminen

API:n päivittäminen tapahtuu SSH yhteyden välityksellä. Kirjaudu Skripti Ry:n OVH Cloud tunnuksilla OVH Dashboardiin ja siirry siellä Python isännöintipalvelun alle. Sieltä löydät välilehden FTP-SSH. Ota SSH yhteys välilehdeltä löytyvään osoitteeseen.

Komentoriville aukeaa OVH Cloudin SSH liittymä. Juurikansiota löytyy kaksi kansiota, *www* ja *R05*. Siirry kansioon *R05* ja tee Git Pull. Siirry *server/flask* kansioon ja siirrä päivitettyt tiedostot *juuri/www* kansioon. Jos sovelluksen päätiedostoa, eli *testAPI.py* tiedostoa on päivitetty, niin se tulee uudelleennimetä muotoon *app.py*. Tämän jälkeen siirry kansioon *www* ja suorita komento *touch tmp/restart.txt*. Tämä kertoo palvelimelle, että sen tulee kääntää tiedostot uudelleen.

Tiivistettynä:

1. Git Pull repositorion muutokset
2. Siirrä päivitettyt API:n tiedostot *www* kansioon ja uudelleennimeä *testAPI.py* → *app.py*
3. Suorita *www* kansiossa komento *touch tmp/restart.txt*
4. Varmista palvelimen toiminta selaimella siirtymällä osoitteeseen *api.skripti.org*

Jos sovelluksen kääntymisessä tulee virheitä, niin verkko-osoitteessa *api.skripti.org* Phusion Passenger middleware tulostaa virheilmoitukset. Tämän jälkeen nano tekstieditorilla tekemään korjauksia...

4.2.3 Virheiden raportointi

Jos sovelluksessa havaitaan virheitä, niin niistä tulee ilmoittaa Skripti Ry:n hallitukselle, joka ohjaa virheet eteenpäin skripti-dev ryhmälle. Kehitysryhmällä ei ole sitoumuksia korjata virheitä, sillä sovelluksen jatkokehitys perustuu vapaaehtoisuuteen.

5 LISÄTIETOJA

5.1 Yhteystiedot

Sovellukseen liittyvät kyselyt voi lähettää Skriptin hallitukselle osoitteeseen hallitus@skripti.org tai Telegramilla hallituksen jäsenille.

5.2 Kirjastojen dokumentaatio

Flask API: <https://flask.palletsprojects.com/en/1.1.x/>

Python-Telegram-Bot:

<https://github.com/python-telegram-bot/python-telegram-bot>