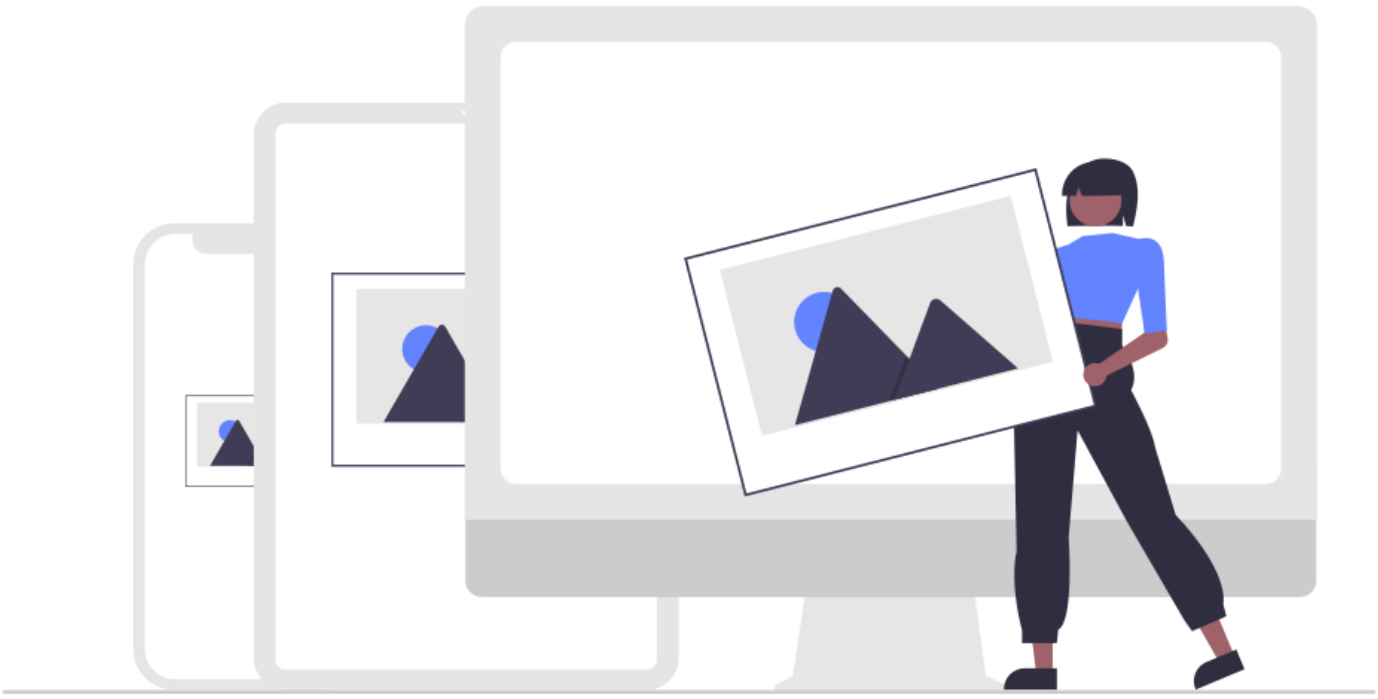


KUVA-ANNOTAATTORI

Projektidokumentti, 04.05.2022

Rajala Joni
778549, Kolmas vuosi kauppatieteet

1. **Yleiskuvaus ja vaikeustaso**

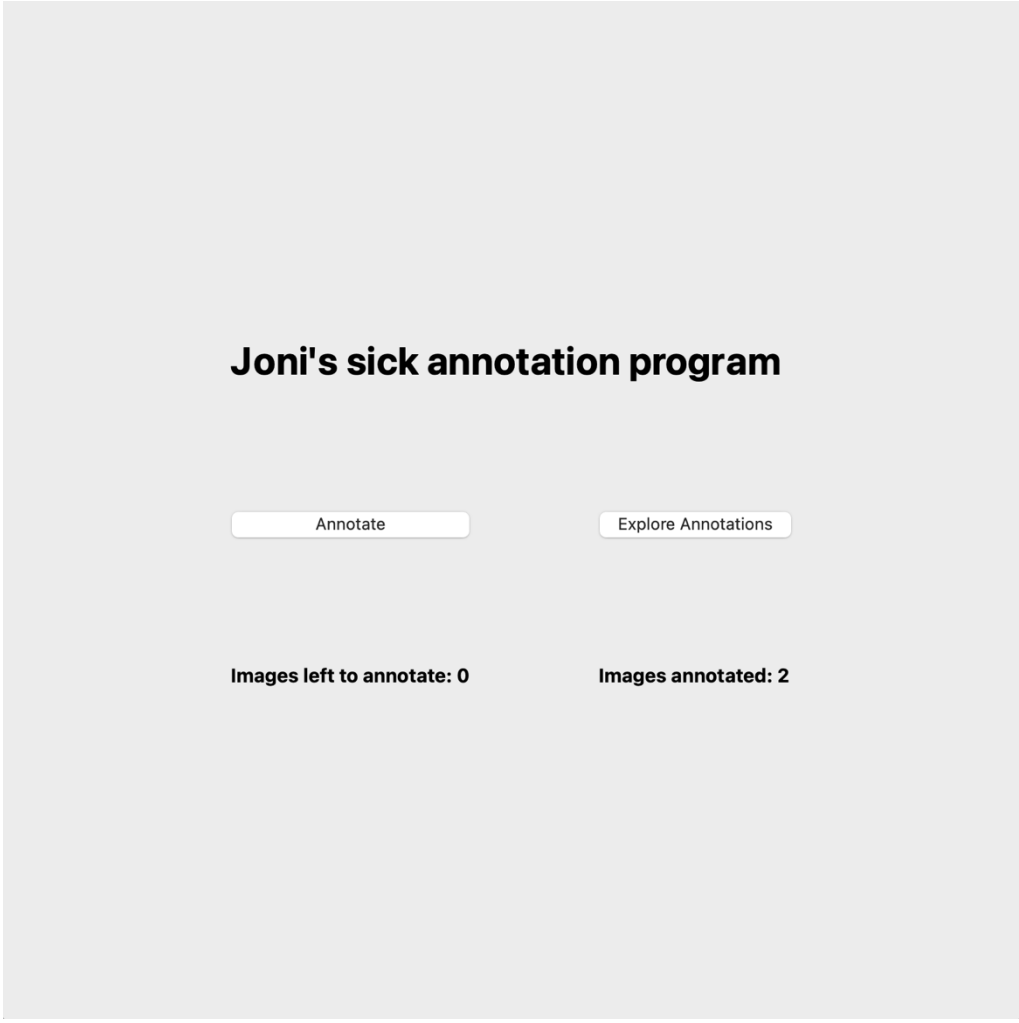
   Ohjelmisto, jonka avulla voidaan rakentaa konvoluutioverkkoa kouluttava tietojoukko. Ohjelmiston avulla voit annotoida kuvia, selata jo annotoituja kuvia, sekä ohjelmisto tarjoaa valmiiksi luodun PyTorch datasetin annotoiduista kuvista.

   Yhdestä kuvasta voidaan annotoida monia eri esineitä omiin luokkiinsa. Annotaatiot SQL tiedostoon kuvan nimen sekä annotaation luokan kanssa. Yksi rivi SQL tablelissa sisältää siis: id (primary key), image, annotation (muodossa begin_x, begin_y, end_x, end_y), object.

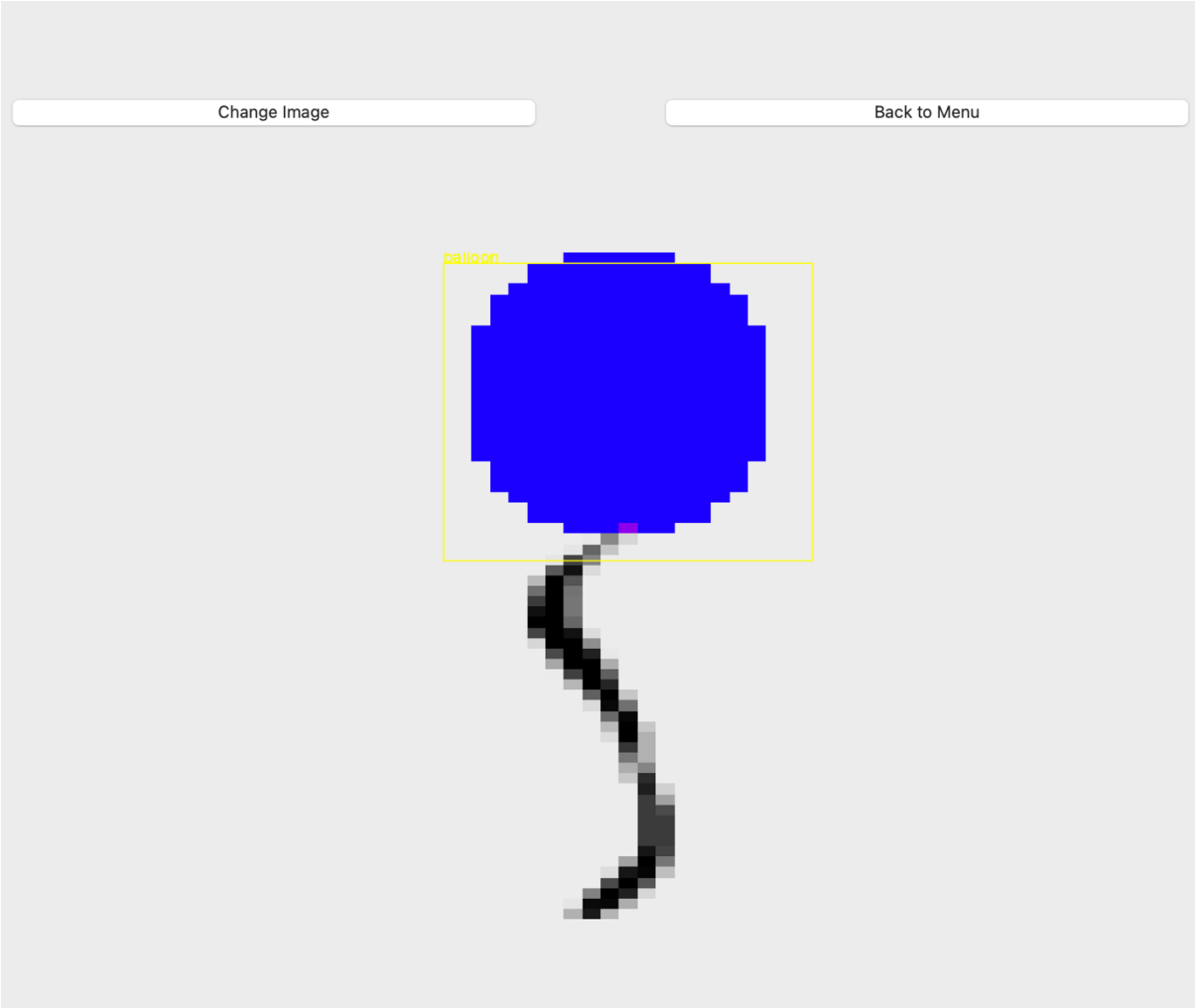
   Tästä SQL tiedostosta ohjelma myös lukee annotaatiot sekä rakentaa datasetin tarvittaessa. Vaikeustaso vaikea. Yksilöllisen ohjelmasta tekee laskurit, jotka kertovat kuinka monta kuvaa käyttäjä on annotoinut ja monta kuvaa on vielä annotoimatta.

2. **Käyttötapauskuvaus ja käyttöliittymän luonnos**

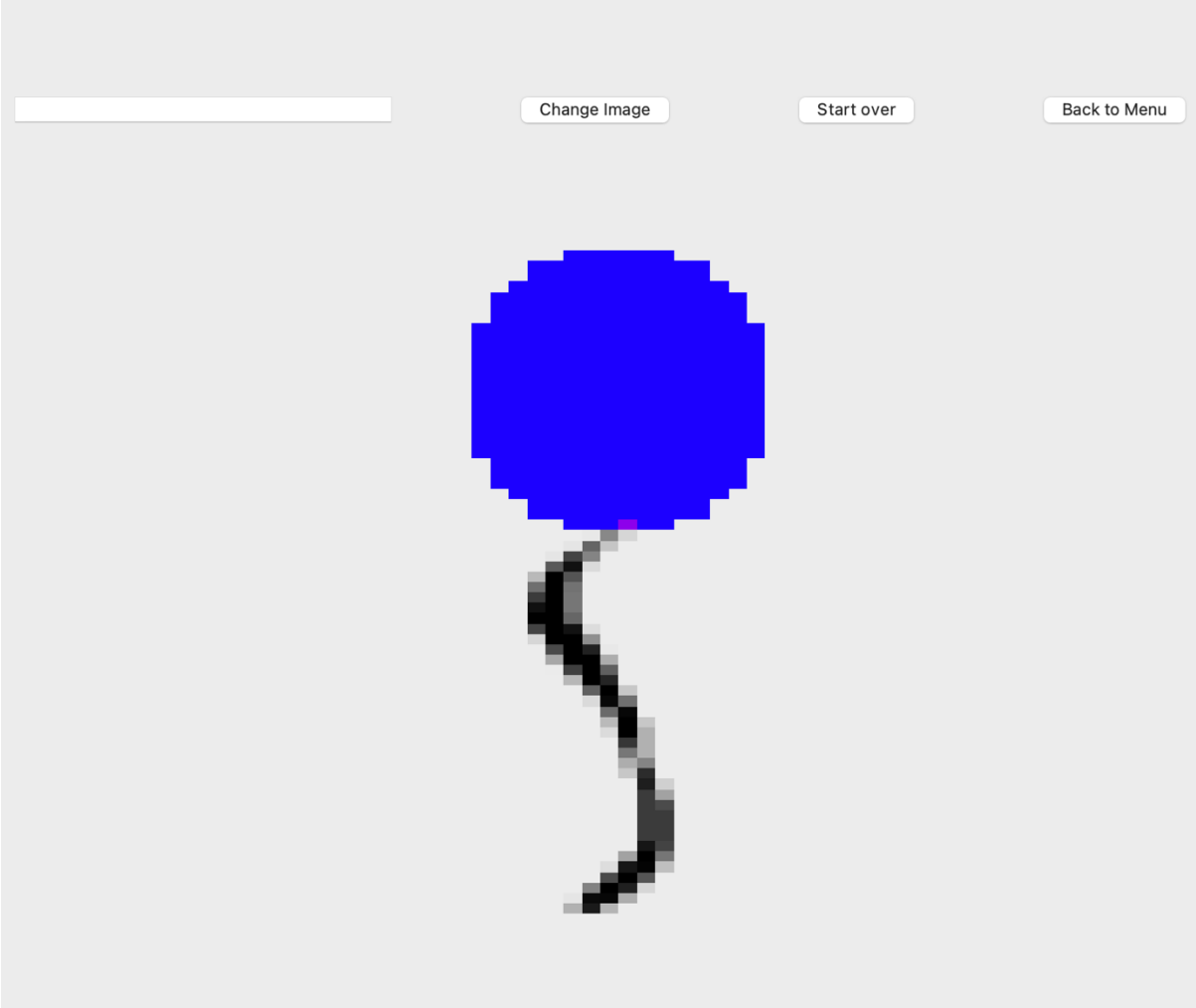
   Ohjelman avautuessa käyttäjä voi valita haluaako hän annotoida kuvia vai selata valmiiksi annotoituja kuvia.



Annotoituja kuvia selatessa kuvat vain avautuvat ruudulle annotaatiot piirrettynä. Nuolesta painaessa tulee ruudulle uusi kuva.



Jos haluaa annotoida kuvia, tulee kuvat ensin siirtää images/not_annotated kansioon, josta ohjelma osaa lukea ne. Kuvat avautuvat ruudulle, jossa on tekstikenttä, johon käyttäjä kirjoittaa ensin annotoitavan objektin luokan ja sen jälkeen hiirellä piirtää neliön objektin ympärille. Kun kaikki annotaatiot ovat piirrettynä, käyttäjä painaa ”Change Image” painiketta, jonka jälkeen annotaatiot tallentuvat ja ruudulle ilmestyy uusi kuva.



3. **Ulkoiset kirjastot**

PyQt5

- Graafisen käyttöliittymän luonti

SQLite3

- SQL tietokannan luonti ja sen kanssa kommunikointi

Torchvision, PyTorch ja Pandas

- Datasetin luonnissa käytettävät kirjastot

4. **Ohjelman rakenne**

   GUI luokka toimii ohjelman MainWindowina, johon eri widgetit lisätään. Se pitää sisällään muuttujat ja funktiot, joita kaikki widgetit käyttävät. Tallettaa itseensä aina näytettävän widgetin, jota esimerkiksi Menu luokka muuttaa, kun käyttäjä valitsee, haluaako hän selata vai annotoida kuvia.

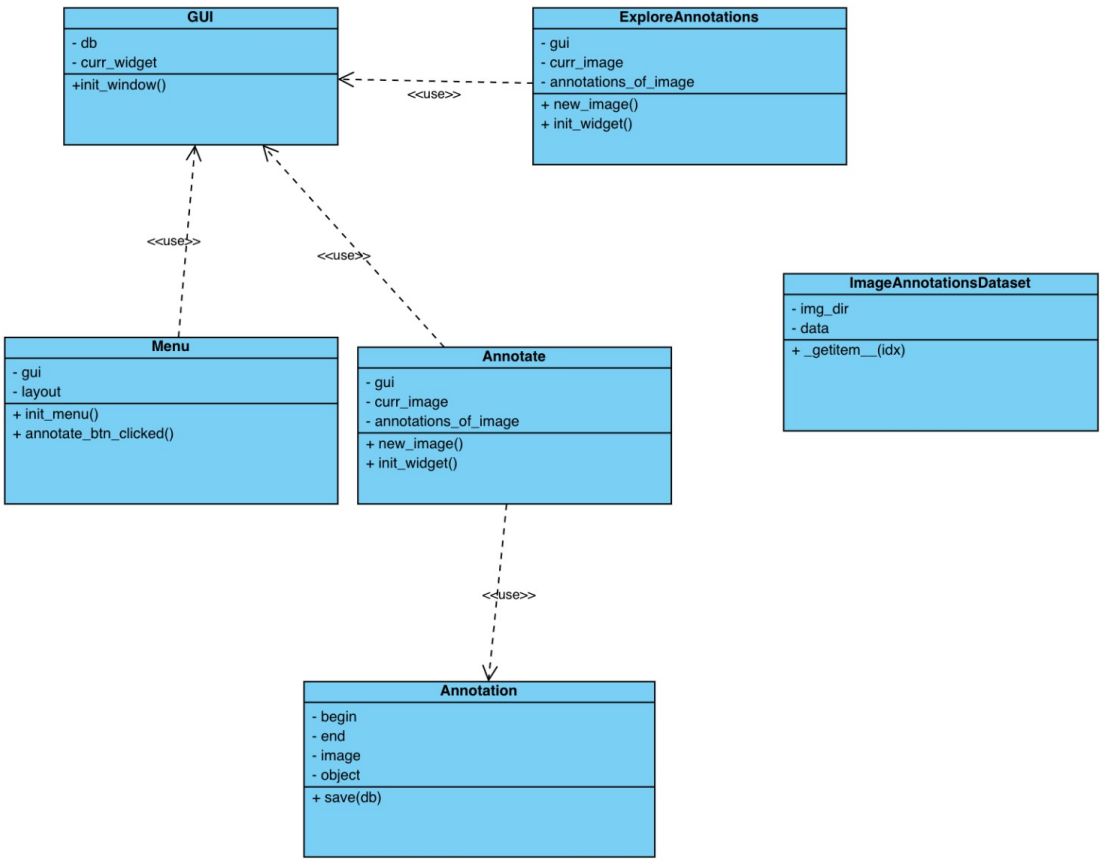
   Menu luokka on widget, joka avautuu, kun ohjelma käynnistetään. Sen avulla käyttäjä voi suunnistaa joko tutkimaan annotoituja kuvia tai annotoimaan lisää kuvia. Sisältää funktiot napeille, joilla käyttäjä suunnistaa ohjelmiston sisällä ja funktion, jolla widget alustetaan.

   Annotate luokka on widget, jonka avulla käyttäjä voi annotoida kuvia. Annotaatioista se luo Annotation luokan olioita. Sisältää funktiot, joilla luokka hoitaa annotaatioiden piirtämisen sekä funktiot napeille, joilla vaihdetaan kuvaa, aloitetaan alusta ja palataan menuun.

   ExploreAnnotations luokka on widget, jonka avulla käyttäjä voi selata jo annotoituja kuvia. Sisältää funktion, jolla kuva ja annotaatiot piirretään ja funktioita napeille.

   Annotation luokan avulla voidaan mallintaa ja tallentaa annotaatioita. Sisältää vain funktion, jonka avulla kyseinen annotaatio voidaan tallentaa databaseen.

   ImageAnnotationsDataset luokka tarjoaa käyttäjälle mahdollisuuden importata luomansa annotoidut kuvat datasettinä omaan käyttötarkoitukseensa. Se ei siis sisälly graafiseen käyttöliittymään. Sen rakenne noudattaa PyTorchin Custom Datasetin rakennetta.



5. **Tietorakenteet**

   Annotaatioiden tallennus toteutettiin SQL tietokantaan. Tietokanta sisältää vain yhden taulukon ”Annotations”. Tallettaminen voitaisiin hoitaa myös csv tiedoston avulla, mutta SQL tietokannan käyttö on tuo mukavaa harjoitusta sekä se antaa mahdollisuuden tulevaisuudessa tehdä tietokannasta monimutkaisemman. Esimerkiksi kuville voitaisiin tehdä oma taulukko, jos haluisimme tallentaa enemmän tietoa kuvista ja sitten annotations taulukon image sarake voisi olla foreign key.

   Ennen tallennusta tietokantaan, annotaatiot tallennetaan listaan. Listaa käytetään myös tallettamaan kaikki kuvat, kun halutaan selata niiden annotaatioita.

6. **Tiedostot ja tiedostoformaatit**

   Kuten jo mainittu annotaatiot tallennetaan SQL tietokantaan. Tämän lisäksi ohjelma käyttää kuvatiedostoja. Ohjelma tukee yleisimpiä kuvatiedostomuotoja, mutta niiden on käytettävä värimuotona RGB:tä, jotta PyTorch pystyy lukemaan ne.

7. **Algoritmit**

   Koska ohjelma on teknisesti melko simppeli ei se sisällä erikoisia algoritmeja tai matemaattisia laskutoimituksia. Ainoastaan Annotaation-luokan save

methodi käyttää sorttaus algoritmiä, selvittääkseen annotaation vasemman yläkulman ja oikean alakulman.

8. **Testaus**

   Testeissä keskityin siihen, että minään hetkenä vääränlaisia annotaatioita ei tallenneta tietokantaan tai vääränlaista dataa ei lueta tietokannasta. Tätä testaan kuvia importatessa sekä annotaatioiden tallennuksessa. Koska annotaatio rakentuu kuvasta, annotaation koordinaateista ja annotoiduista objektista, testaan erikseen kaikkia näitä. Esimerkiksi koordinaattien tulee olla nollan ja kuvan koon välissä, kuvan tulee olla oikeassa tiedostomuodossa ja objektin tulee tekstiä eli se ei voi olla tyhjä.

9. **Ohjelman tunnetut puutteet ja viat**

   Ohjelman käyttö ei ensikerralla tunnu kovin intuitiiviselta ja käyttäjä saattaa joutua hieman harjoittelemaan ennen kuin sen käyttö alkaa tuntumaan luonnolliselta. Tämän ongelman korjaaminen vaatisi sitä, että testaisin erilaisia vaihtoehtoja nappien sijainneille ja toiminnoille. Lisäksi ohjelmaan olisi voinut vielä lisätä ohjeistavia tekstejä, esimerkiksi kun annoitoitavat kuvat loppuvat.

10. **3 parasta ja 3 heikointa kohtaa**

    Parhaat:

    - Ohjelman yleinen toimivuus
        1. Vaikka ohjelma on aluksi hieman epäintuitiivinen, toimii se kuitenkin juuri niin kuin pitääkin.
    - Ohjelman tarjoama PyTorch datasetti annotoiduille kuville

    Heikoimmat:

    - Aluksi ohjelman käyttö saattaa tuntua hieman vaikealta
        1. Ohjelma saattaisi olla intuitiivisempi, jos objektin maalaus tapahtuisi ennen kuin se nimetään.

11. **Poikkeamat suunnitelmasta**

    Ajankäyttö arvioni osui juuri oikeaan ja tein lähes kaiken täysin kuin olin suunnitellut. Pieniä poikkeuksia kuten nappien sijainteja lukuun ottamatta. Lisäksi asia joka yksilöllisti ohjelmani muuttui alkuperäisestä suunnitelmasta.

12. **Aikataulu**

    Ensin loin GUI ja Annotate luokat. Lähes valmiin Annotate luokan pohjalta pystyin sitten helposti luomaan ExploreAnnotations ja Annotation luokat. Lopuksi loin ImageAnnotationDataset luokan.

    Kun luokat olivat lähes valmiit aloin rakentamaan testejä.

15.2 – 21.2 Välisenä aikana sain luokat lähes valmiiksi

- Aikaa kuluin noin 7 tuntia

21.2 - 28.2 välisenä aikana loin ohjelmalle testit ja varmistin, että ohjelma toimii halutulla tavalla.

- Kesto noin 10 tuntia

28.2 - 7.3 välisenä aikana viimeistelin ohjelmiston lisäämällä kommentteja ja parantamalla koodin luettavuutta.

- Kesto noin 5 tuntia

04.05 vielä tein muutamia pieniä korjauksia ja viimeistelin ohjelman.

13. **Arvio lopputuloksesta**

Ohjelman toteutus oli melko helppoa ja nopeaa. Tästä johtuen sain ohjelmistosta toimivan ja ohjeiden mukaisen. Ajoittain ohjelmankäyttöliittymä saattaa olla hieman tönkkö ja epäintuitiivinen. Jos ohjelmasta haluaisi paremman, tulisi käyttöliittymä suunnitella uusiksi, testaten erilaisia vaihtoehtoja. Koska eri ohjelman osat ovat rakennettu omiin luokkiin, on sen osien muuttaminen helppoa tulevaisuudessa.

14. **Kirjallisuusviitteet ja linkit**

Tärkeimpänä lähteenä Qt:n oma dokumentaatio.

- https://doc.qt.io/qt-5.15/
- https://doc.qt.io/qtforpython/

SQLite3 käyttöön

- https://www.sqlitetutorial.net/sqlite-python/creating-database/

Datasetin luomiseen

- https://pytorch.org/docs/stable/data.html#torch.utils.data.Dataset
- https://pytorch.org/tutorials/beginner/basics/data_tutorial.html

Lisäksi StackOverFlow artikkeleita eri toteutusteknisiin asioihin